EFFICIENTLY SEARCHING LARGE COLLECTION OF DOCUMENTS IN A DISTRIBUTED SYSTEM

P.Jency Rani[1], J.Rajalakshmi[2], A.Sahaya Nirmal[3], P.Yadunath[4], V.Vidhya[5]
[1,2,3,4] BE CSE , [5]Assistant Professor , Annai Vailankanni College of Engneering

ABSTRACT
The intention of this work is to describe our research and solution to the problem of searching large documents in the fast growing information era effectively and efficiently by designing an "EFFICIENTLY SEARCHING LARGE COLLECTION OF DOCUMENTS IN A DISTRIBUTED SYSTEM". To improve the performance and enhance the search the proposed work introduces the concept of Elasticsearch.
Elasticsearch is a NOSQL, distributed full text database. Which means that this database is document based instead of using tables or schema, documents can be used. Most software generates tons of data that is worth analyzing, Elasticsearch comes with Kibana to give a full analytics system. Kibana has the feature of visualization which helps the users of the system to view the search result in a user friendly manner. The sense plug-in of the elastic search provides a graphical user interface for the Elasticsearch system.
Index Terms: Data warehouse, Elastic search, Kibana, Sense, SQL,

1. INTRODUCTION

"EFFICIENT SEARCHING OF LARGE DOCUMENTS IN A DISTRIBUTED SYSTEM" is a search technique introduced to perform search in large amount of data.

Elasticsearch can be viewed as a Data ware house, where large documents with many different attributes and non-predictable schemas can be stored. Since Elastic search is schema less, it can store any kind of documents, with any kind of data stored elasticsearch can be capable to search the easily and quickly. With the help of Kibana and sense, the Elasticsearch system can be made more user friendly and the searching can be made easily and effectively.

1.1 Need for large data analysis

Analysis of data is a process of inspecting, cleaning, transforming, and modelling data with the goal of discovering useful information, suggesting conclusions, and supporting decision-making. Data analysis has multiple facets and approaches, encompassing diverse techniques under a variety of names, in different business, science, and social science domains.

Efficient Searching of Large Documents in a Distributed System is the retrieval of results for any given input (e.g. websites in social media like fb, twitter, etc). Here we precede this work with the help of a search engine known as elasticsearch. This elasticsearch uses a plug-in called kibana to visualize data in real time.

2. LITERATURE SURVEY

2.1. Information Retrieval (IR) System Using Keyword Search Technique

Keyword search is the most famous information discovery technique because the user does not need to know either a query language or the underlying structure of the data. Large number of techniques is used in Information Retrieval (IR) system. Keyword search is the technique used for the retrieving data or information. Keyword search can be implemented on both structured and semi-structured databases; also it is possible on graph structure which combines relational, HTML and XML data. Keyword search uses a number of techniques and algorithms for storing and retrieving data but the method is less accuracy and does not give a correct answer, require large time for searching and large amount of storage space for data storage.

Data mining or information retrieval is the process to retrieve data from dataset and transform it to understandable form, so users can easily get their desired information. Keyword search on relational databases find the answer of the tuples which are connected to database keys like primary key and foreign keys. So this system also present comparative search techniques using keyword search like DISCOVER, BANKS, BLINKS, EASE, and SPARK.

Here in this system, initially the admin should login in to the system. The admin has to upload the information and keyword which are needed by the user. Registered candidate can upload the keyword and the file length. Then ranking is created based on the file uploaded. Since the search is based on the ranking the time required for execution is very less. The File length and Execution time can be seen by using chart. The registered users will finally receive the required information to their mail.

2.2. Efficient Query Optimizing System for Searching Using Data Mining Technique

With the tremendous growth of information available to end users through the Web, search engines come to play a more critical role. Nevertheless, because of their general purpose approach, it is always less uncommon that obtained result sets provide a burden of useless pages. Next generation Web architecture, represented by Semantic Web, provides the layered architecture possibly allowing to overcome this limitation. Several search engines have been proposed, which allow increasing information retrieval accuracy by exploiting a key content of Semantic Web resources that is relations. However, in order to rank results, most of the existing solutions need to work on the whole annotated knowledge base.

2.3. DRAWBACK OF EXISTING SYSTEM

Both of the existing system, has big disadvantage while handling large amount of data. They both lag extremely at speed during the processing of large data. The process time duration required for searching in such large data using these systems is high. This kind of large data processing usually makes the server gets overloaded and can end up in crashing.

2.2. Proposed System

Here we propose the system to efficiently searching large collection of documents in a distributed system. This has been developed to solve the issues of processing and searching of large data in the existing systems. Main application area of this system are social Medias, marketing agencies, hospitals, search engines, survey etc.

This system is developed using the frameworks Elasticsearch and Kibana with the help of java language. Elasticsearch is a search server. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. Kibana is open source data visualization plug-in for Elastic search. It provides visualization capabilities on top of the content indexed on an Elastic search cluster. Users can create bar, line and scatter plots, or pie charts and maps on top of large volumes of data.

Searching system that we propose is an information retrieval method which provides optimal searching and highlighting the certain search based on the query and represents them in a visualised format through Kibana. The visualisation usually uses graphical format.

3. MODULE DESCRIPTION

The project "Efficiently Searching Large Collection of Documents in a Distributed System" has the following module

1. Elasticsearch Configuration
2. Kibana Configuration
3. Sense
4. Index Creation & Search
5. Elasticsearch in a distributed environment

3.1. Elastic search Configuration

Elasticsearch is a real-time distributed search and analytics engine. It allows to explore the data at a high speed and scale. It is used for full-text search, structured search, analytics, and combination of all the three. Elasticsearch can run on your laptop, or scale out to hundreds of servers and petabytes of data.

The only requirement for installing Elasticsearch is a recent version of Java. The latest version of Elasticsearch can be downloaded from *elastic.co/downloads/elasticsearch*. The downloaded package includes various formats

such as ZIP and TAR.GZ. After downloading it is necessary to extract the package.
First the downloaded package is unzipped using the following comments

```
gunzip elasticsearch<version>.tar.gz
```

Then the unzipped package is untared using the following comments

```
tar -xvf elasticsearch<version>.tar
```

Now the formats such as ZIP and TAR.GZ in the downloaded package are removed and the package is ready for configuration.
To ensure the availability of the elasticsearch in the system the following comment is executed in the terminal

```
cd elasticsearch-<version>
```

The comment displays the current version of the elasticsearch installed. It shows that elasticsearch is installed successfully.
The configuration file (elasticsearch.yml) in the elasticsearch is updated for its cluster name and host name.
The elasticsearch is executed using the comment ./elasticsearch.
Now the elasticsearch is ready for execution and starts as an instance in the system. Default port used by the elasticsearch is 9200.
The configuration of the elasticsearch can be checked in the browser by typing IP address: port number (192.168.1.14:9200). Once the elasticsearch successfully starts the following code is displayed in the browser

```
{
 "name": "Tom Foster",
 "cluster_name": "elasticsearch",
 "version":
{
  "number": "2.1.0",
  "build_hash":
"72cd1f1a3eee09505e036106146dc1949dc5dc87",
  "build_timestamp": "2015-11-18T22:40:03Z",
  "build_snapshot": false,
  "lucene_version": "5.3.1"
 },
 "tagline": "You Know, for Search"
}
```

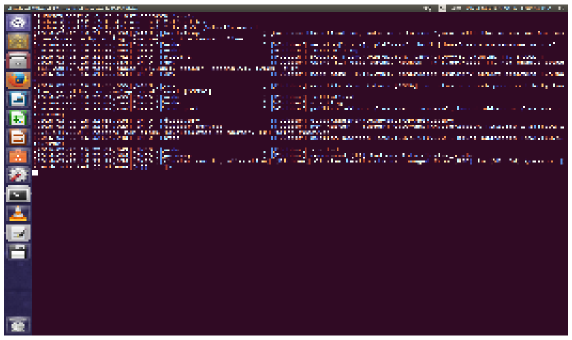

Fig: 3.1. Elasticsearch – Start-up Window - Once the Elasticsearch is configured successfully in the system, it starts up and run as an instance in the system. As Elasticsearch starts running the following window appear in the terminal.

3.2. Kibana Configuration

After configuring the elasticsearch its time to configure the Kibana for providing the visualization of the search string and the search results. As in elasticsearch the downloaded kibana package also includes various formats such as ZIP and TAR.GZ. It has to be unzipped and untared for usage.
First the downloaded package is unzipped using the following comments

```
gunzip kibana-<version>.tar.gz
```

Then the unzipped package is untared using the following comments

```
tar -xvf kibana<version>.tar
```

Now the formats such as ZIP and TAR.GZ in the downloaded package are removed and the package is ready for configuration.
The configuration file (kibana.yml) in the kibana is updated for its host address. The Kibana runs in the default port 5601.
Now the kibana can be executed using the comment ./kibana.

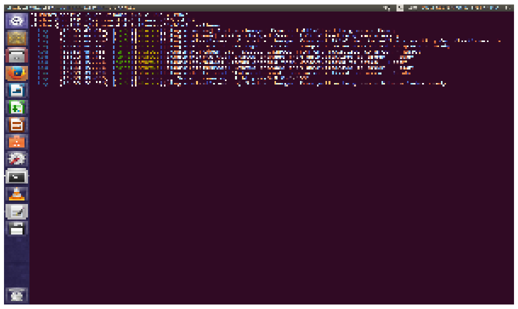

Fig:3.2. Kibana – Start-up -
After configuring the Elasticsearch, the next comes the Kibana. It helps in visualization of the search given by the user. Once Kibana is configured and successfully launched the

following window opens in the terminal as soon as ./kibana comment is executed.

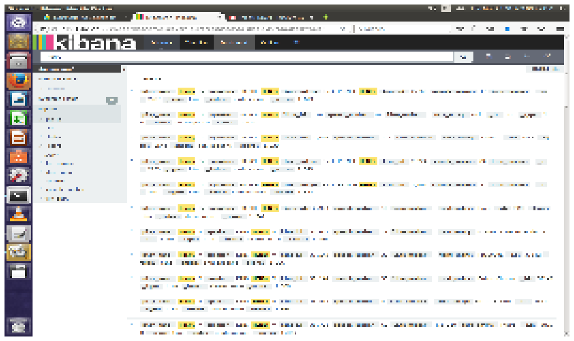

Fig: 3.3. Kibana – Search using the graphical interface - The Kibana provides easy search mechanism for the users. The users have to just type the search string in the search bar in the window shown below. The search results are highlighted in the document and the number of hits is displayed in the screen. The interface also provides many options like save and share so that the search performed can be saved for the futures reference and shared. It provides a very user friendly so that search can be performed very easily.

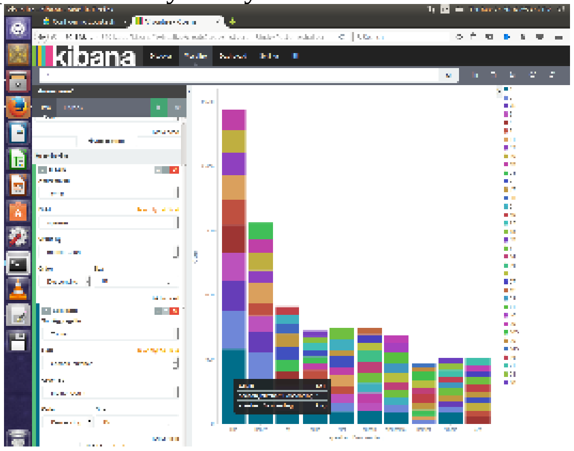

Fig: 3.4. Kibana – Visualization of bar chart - Kibana bar char visualization of data fields is shown in the below screen shot. In which the search results are visualized in a bar chart.

3.3. Sense

Next comes the sense, downloaded sense package also includes various formats such as ZIP and TAR.GZ. It has to be unzipped and untared for usage.

First the downloaded package is unzipped using the following comment

gunzip sense-<version>.tar.gz

Then the unzipped package is untared using the following comments

tar -xvf sense<version>.tar

Now the browser window is opened to check the kibana using the comment IP address: port number (192.168.1.41:5601). The graphical user interface of the Kibana and sense gets opened in the browser window. The system is configured successfully for the next level of search.

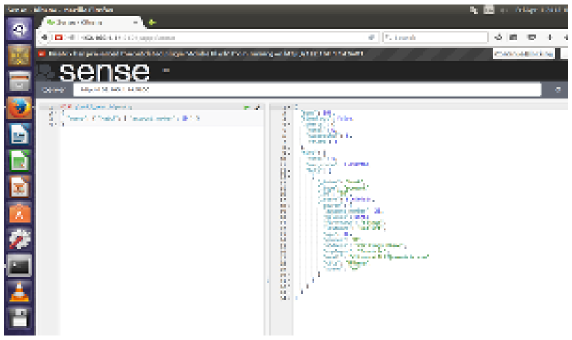

Fig: 3.5. Sense Interface The below screen shot shows the sense interface to create index for the document to be searched. The window is split into two portions. In the left side the query is types and the execution output is displayed in the right side. The same can be executed in the terminal but sense provides a graphical interface for the execution of the query.

3.4. Index Creation & Search

Index helps in efficient search of the large document. In elasticsearch environment, index can be either created in the terminal window or the graphical user interface of the sense.

The first step after successful configuration is to create a index file. The search operation is executed based on the index created by the users. The index is created using 'PUT' comment in CURL.

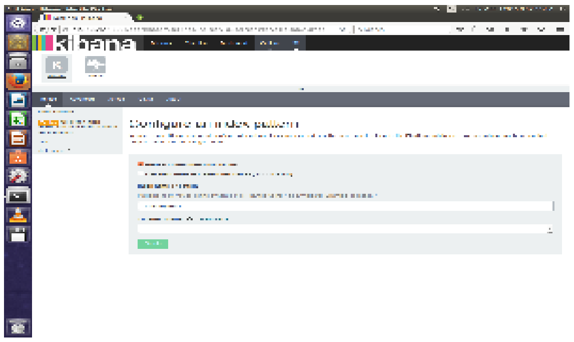

Fig: 3.6. Configuring an Index in Kibana - The first task to be done in the kibana interface is to configure an index. The interface to configure an index is given below.

3.5. Elasticsearch in distributed environment:

Elasticsearch can be implemented in distributed environment by connecting more than one system.

The elasticsearch package is downloaded and configured in the system as in Elasticsearch configuration module. Then the cluster name in config file is changed to user defined name. Then the nodes are added to the cluster. The IP address of the other nodes in the distributed system is added to the configuration file using the following comments

$ vim config/elasticsearch.yml // opens the configuration file

cluster.name: User defined name of the cluster

node.name: IP address of the connected node

Install Elasticsearch-Head Plug-in (All Nodes)

Elasticsearch-head is a web front end for browsing and interacting with an Elastic Search cluster. Use the following command to install this plug-in on all node systems.

$ Bin/plug $ in --install mobz/elasticsearch-head

Starting Elasticsearch Cluster (All Nodes)

As the Elasticsearch cluster setup has been completed. Let start Elasticsearch cluster using following command on all nodes.

$ ./bin/elasticsearch &

By default elasticsearch listen on port 9200 and 9300. So connect to the nodes to the port 9200 like following url,

http://IP address (connected node):9200/_plug-in/head/

Verify Multi Node Cluster

To verify that cluster is working properly. Insert some data in one node and if the same data is available in other nodes, it means cluster is working properly.

1) *Insert Data on NODE_1*

To verify cluster create a bucket in Node_1 and add some data.

$ curl -XPUT http://NODE_1:9200/mybucket

$ curl -XPUT 'http://NODE_1:9200/mybucket/user/rahul' -d '{ "name" : "Rahul Kumar" }'

$ curl -XPUT 'http://NODE_1:9200/mybucket/post/1' -d '
{
  "user": "rahul",
  "post date": "01-16-2015",
  "body": "Adding Data in Elasticsearch Cluster" ,
  "title": "Elasticsearch Cluster Test"
}'

2) *Search Data on All Nodes*

Now search same data from Node_1 and check if same data is replicated to other nodes of cluster. As per above commands we have created a user named and added some data there. So use following commands to search data associated with user rahul.

$ curl 'http://NODE_1:9200/mybucket/post/_search?q=user:rahul&pretty=true'

3) *View Cluster Data on Web Browser*

To view data on Elasticsearch cluster access of elasticsearch-head plug-in using one of cluster ip at below url. Then click on browser tab.

http://NODE_1s:9200/_plugin/head/

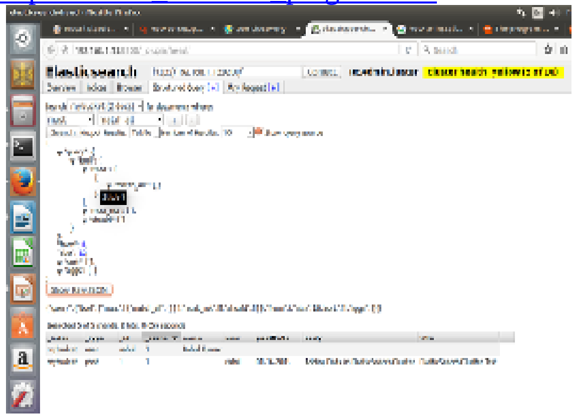

Fig: 3.6. Elasticsearch job submission in distributed environment - The following screenshot shows the job submission in distributed environment

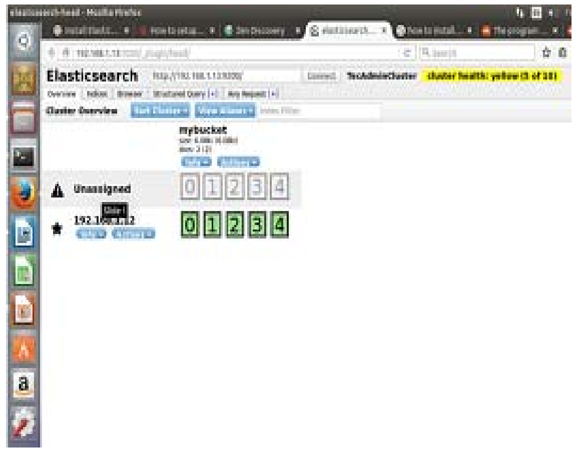

Fig: 3.7. Elasticsearch in distributed environment - The following figure depicts the elasticsearch in an distributed system. One node with IP 192.168.1.12 is attached with the server.

4. CONCLUSION

4.1 Conclusion

The need for a system to efficiently searching large collection of documents in a distributed system has become vital .As the size and collection of data is being increased day by day, the existing system is not able to handle such big collection of data efficiently. The existing system lags at speed and time for the retrieval of certain information.

The project "Efficiently searching large collection of documents in a distributed" introduces a system to tackle the drawbacks of the existing search technique. This system is based on the concept of Elasticsearch and Kibana. Elasticsearch plays the key role in retrieving the specific information from the huge collection of data. By making use of Kibana, we were able to convert the results into visualized format. The main objective of our system is to provide efficient searching for huge documents. The merits of our system includes enhanced accuracy of search results, more reliability than existing system, scalability and the most important one is it's high time efficiency. By attaining all these objectives, the ultimate goal of the system is achieved.

4.2 Future Enhancement

As future work we can upgrade the elasticsearch performance by integrating it with Hadoop. Now the world of information processing is moving to the concept of big data. Big data and related processing has become a recent advanced and useful technology in processing large collection of data in real world. While integrating the proposed system with Hadoop, the input data can be easily loaded from hadoop HDFS (Hadoop Distribution File System) and get the concerned input data.

7. REFERENCES

[1] https://www.elastic.co/guide/

[2] http://joelabrahamsson.com/

[3] R. Baeza-Yates, C. Hurtado, and M. Mendoza, "Query Recommendation Using Query Logs in Search Engines," Proc. Int'l Conf. Current Trends in Database Technology (EDBT '04), pp. 588- 596, 2004.

[4] D. Beeferman and A. Berger, "Agglomerative Clustering of a Search Engine Query Log," Proc. Sixth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '00), pp. 407-416, 2000.

[5] S. Beitzel, E. Jensen, A. Chowdhury, and O. Frieder, "Varying Approaches to Topical Web Query Classification," Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development (SIGIR '07), pp.783-784, 2007.

[6] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li, "Context- Aware Query Suggestion by Mining Click-Through," Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD '08), pp. 875-883, 2008.

[7] H. Chen and S. Dumais, "Bringing Order to the Web: Automatically Categorizing Search Results," Proc. SIGCHI Conf. Human Factors in Computing Systems (SIGCHI '00), pp. 145-152, 2000.

[8]C.-K Huang, L.-F Chien, and Y.-J Oyang, "Relevant Term Suggestion in Interactive Web Search Based on Contextual Information in Query Session Logs," J. Am. Soc. for Information Science and Technology, vol. 54, no. 7, pp. 638-649, 2003.

[9] R. Jones and K.L. Klinkner, "Beyond the Session Timeout: Automatic Hierarchical Segmentation of Search Topics in Query Logs," Proc. 17th ACM Conf. Information and Knowledge Management (CIKM '08), pp. 699-708, 2008.