International Journal of Soft Computing and Engineering (IJSCE)
ISSN: 2231-2307, Volume-3, Issue-6, January 2014

Use of Elastic Search for Intelligent Algorithms to Ease the Healthcare Industry

C. Bhadane, H. A. Mody, D. U. Shah, P. R. Sheth

Our project is optimizing and adding an intelligent algorithm to a cross-platform software that tabulates the medicines and provides for easy retrieval of a list of similar medicines based upon a keyword. Relational databases and schematic databases had several drawbacks. This lead to the use of Elastic Search as the indexing technique for the project. Elastic Search being an unconventional Database (in the sense it is schema less), has certain drawbacks while dealing with data. Inspite of its real-time efficiency while handling terabytes of data and auto cluster balancing, it could use some improvement while deploying on the client side. We plan to use elastic search's analytics tool to help improve the queried data. And also preprocess the query to make it faster before sending it over to the server so as to reduce the time latency while being used over slower data connections

I. INTRODUCTION

Elastic Search was chosen because it is convenient to set up nodes and clusters. The API makes it easy to use different languages like Java, Ruby, Perl, Python, and more. In runtime, elastic search manages distribution—adding a node is quite easy and data is redistributed automatically.

Elastic Search is a schema less indexing technique. Hence hashing or indexing is similar to graphs in it. Elastic Search is able to achieve fast search responses because, instead of searching the text directly, it searches an index instead. This is like retrieving pages in a book related to a keyword by scanning the index at the back of a book, as opposed to searching every word of every page of the book. This type of index is called an inverted index, because it inverts a page-centric data structure (page->words) to a keyword-centric data structure (word->pages).Elastic Search uses Apache Lucene to create and manage this inverted index. While working with Elastic Search, several shortcomings were also noticed. The need for an intelligent algorithm used to prioritize the retrieved result of search queries was missing. Additionally, Query Optimization must be carried out to increase the response speed over slow internet connections.

Work shall be done on Elastic search using an intelligent algorithm that will give as its result an insight into the popularity of the medicines. Thus, using this algorithm on elastic search, queries into the index will give an intelligent output in accordance with the popularity of the medicine. Along with this, query optimization shall be carried out into the index for faster responses that will give quick results at slow internet speeds as well. Currently, the problems faced by the existing system are slow response at slow internet connections and non intelligent response to searches.

Manuscript Received on January 2014.

C.Bhadane, Computer Engineering, D. J. Sanghvi College of Engineering, Mumbai, India.

P.R. Sheth, Computer Engineering, D. J. Sanghvi College of Engineering, Mumbai, India.

D. U. Shah, Computer Engineering, D. J. Sanghvi College of Engineering, Mumbai, India.

H. A. Mody, Computer Engineering, D. J. Sanghvi College of Engineering, Mumbai.

II. LITERATURE REVIEW

There are various real life projects that already use elastic search in the way that suits their business profile. Here, we have done a study on 4 different very famous projects to understand the best way to use elastic search and also find the real time problems faced by them.

1.STUMBLEUPON

StumbleUpon is the easiest way to discover new and interesting things from across the web. Every month millions of people turn to StumbleUpon to be informed, entertained and surprised by content and information recommended just for them. .As a customer "stumbles" through great web pages, they can tell StumbleUpon whether they like or dislike recommendations so they can be shown more of what they are looking for. StumbleUpon will show web pages based on that feedback as well as what similar Stumblers have liked or disliked. The Challenge involved was to create a scalable, ultra-fast Internet discovery platform for products based on user interest StumbleUpon previously used an implementation of Apache Solr for it search requirements. However, as StumbleUpon grew to millions of users, Solr had difficulty scaling and was difficult to administer and manage. ApacheSolr was not a good fit for StumbleUpon . It was very difficult to manage. Issues with servers going down due to overload were reported. It was simply not scalable. When StumbleUpon decided to upgrade its search infrastructure it began to look for alternative solutions. After attending an Elasticsearch training course and meeting Elasticsearch creator Shay Banon, StumbleUpon decided to implement Elasticsearch for all keyword search requirements due to its real-time capabilities and ease of use. Today, Elasticsearch is deeply integrated in the core StumbleUpon recommendation engine which handles 30 million "stumbles" or recommendations per day. The majority of Elasticserch queries come from the StumbleUpon recommendation engine. Elasticsearch can grab all categories and balance the categories and updates in the keyword search index to get real-time, high-speed queries. Specialty purpose indexes with different levels of search are also updated frequently.

StumbleUpon used to hold data in MySQL but now stores data in Elasticsearch to handle more flexible queries and achieve very fast, horizontally scalable performance. StumbleUpon's main keyword search index's average latency equals 25 milliseconds of throughput and 3,000 queries per second. Elasticsearch is also very easy to administer and manage. Adding additional nodes to a cluster is greatly simplified and its very flexible API approach make integrating with additional indexes and data sources easy, such as Redis, MySQL and HBase.

2. SOUNDCLOUD

As the world's leading social sound platform, SoundCloud lets anyone create, record, promote and share their sounds on the web in a simple, accessible and feature-rich way. SoundCloud enables sound creators to instantly record or upload original audio content, embed sound across websites and blogs, share publicly and privately, receive detailed analytics, plus get feedback from the community directly onto the waveform. Recently, SoundCloud set out to redesign their search infrastructure to meet growing customer demands and to keep pace with the massive growth in their data.The Challenge Involved: Create a flexible, easy-to-use, real-time search engine for SoundCloud's 30,000,000 users. Powerful and Intelligent Search Is Key to Enhanced Customer Experience Providing the best experience for customers to view, search and explore relevant content in a timely and meaningful manner is paramount for SoundCloud. SoundCloud switched to Elastic Search because Faster time to search improves user experience. Feedback from on time-to-searchability has been very positive. Newly posted sounds are discoverable in about three seconds rather than several days with Solr. It was easier to add new functionality. Elasticsearch has greatly expedited the implementation and configuration of new features and functionality. One important issue was streamlined node maintenance. No onerous, manual work is involved in bootstrapping or maintaining nodes – everything is set up with Elasticsearch's REST API and index definitions are specified in flat JSON files. Also, improved insights help decision making. A single dashboard with all relevant metrics helps SoundCloud to know how search is performing, both in terms of load and search quality.

3. McGraw Hill

Learning is moving online, whether it is through stand-alone curriculum or a hybrid online/classroom environment. McGraw-Hill has been responding to the growing demand for technology in the classroom by moving from a textbook business model to a digital-based model. Their focus in recent years has been on developing adaptive learning systems that enable classroom teaching to come closer to a one-to-one student-teacher interaction. Gone are the days when one teacher delivers the same lesson plan to 30 students. Today, instructors are using technology to customize content, learning, and even tests to a student's preferences and learning style. In 2012, McGraw-Hill Education Labs began work on an adaptive learning platform called EdSense. The platform is intended to provide search, discovery, and indexing to the virtually unlimited content available on the Internet, and then offer content recommendations based on the student's preferences and popularity of the content. Textbooks are being replaced by the web. They needed to create an online learning environment that adapts itself to the student's needs and preferences using search, index, and ranking of this virtually unlimited data. The challenge involved was to create a self-adapting learning system that can scale to millions of users. Thus, McGraw switched to Elastic Search. Content discovery and recommendation for virtually unlimited content from the Internet was one reason. McGraw-Hill's role is changing from creating textbook content to delivering third party content in a usable format. Elasticsearch enables them to scale the amount of content they are able to index, search, and recommend, which in turn scales the value of their platform. Monitor and monetize third party content was another goal. EdSense searches and recommends content from a myriad of third party providers, much of which is paid content. Elasticsearch allows EdSense to monitor, bill, and
pay for usage of this material. Scale distribution to millions of learners was also to be achieved. As online learning becomes more a part of everyday life, the company sees the EdSense opportunity to deliver adaptive learning to millions of people. They can scale to this magnitude easily with Elasticsearch. Deliver content based on user preference and content popularity was done using Elastic Search. Content is delivered in real-time to students based on their preferences (print vs. video) or previous behavior (a low score on a test). Using Elasticsearch, Edsense can deliver personalized student learning by assessing each student's skill level and using data to determine how each can progress through lessons most effectively.

III. QUERY OPTIMIZATION

Along with our project, we plan to optimize it. We plan to use the Optimize API that ElasticSearch provides and solve the customers query in a faster and a better way.Elastic Search provides an Optimize API. The optimize API allows to optimize one or more indices through an API. The optimize process basically optimizes the index for faster search operations (and relates to the number of segments a Lucene index holds within each shard). The optimize operation allows to reduce the number of segments by merging them. As per our project, we discovered various ways to optimize query in elastic search.

If the amount of data to search increases, you will need more memory. When you run ElasticSearch, you will encounter many problems due to the use of memory. In an operating method recommended by an ElasticSearch community, when you run a server exclusively for ElasticSearch, you are advised to allocate only half of the memory capacity to ElasticSearch, and to allow the OS to use the other half for system cache. You can set the memory size by setting the ES_HEAP_SIZE environmental variable or by using -Xms and -Xmx of JVM.

When using ElasticSearch, you will see OutOfMemory errors frequently. This error occurs when the field cache exceeds the maximum heap size. If you change the setting for index.cache.field.type from **resident** (default) to **soft**, **soft** reference will be used and the cache area will be preferentially GC, and this problem can be resolved

If the amount of data increases, the number of index files also increases. This is because Lucene, which is used by ElasticSearch, manages indexes in the unit of *segments*. Sometimes the number will even exceed the number of

MAX_OPEN files. For this reason, you need to change the maximum open file limit by using the ulimit command. The recommended value is **32000-64000**, but sometimes you may need to set a larger value depending on the size of the system or data. The following are some methods to optimize the queries and we plan to use a combination of some of these.

1) 1.Index Optimization

NELO2 manages indexes by date. If indexes are managed by date, you can delete old logs that don't need to be managed easily and quickly. In this case, the overhead imposed on the system is smaller than when deleting logs by specifying the TTL value for each document.If index optimization is performed, *segments* are incorporated. Using this method, you can enhance search performance. As index optimization can impose a burden on the system, it is better to perform it when the system is being used less.

2) 2.Shards and Replicas

You can't change the number of shards after setting it. For this reason, you need to decide this value carefully by taking the current number of nodes in the system and the number of nodes expected to be added in the future into account. For example, if there are 5 nodes and the number is expected to reach 10 in the future, it is recommended to set the number of shards as 10 from the beginning. If you set it as 5 in the beginning, you can add 5 more nodes later, but you won't be able to use the added 5 nodes. If you set the number of replicas to 1, of course, you can utilize the added 5 nodes as nodes exclusively for replication.If the number of shards increases, it is more advantageous to process a large amount of data because queries are distributed as much as the number of shards. But you need to set this value appropriately, because the performance could be deteriorated due to increasing traffic if the value is too high.

B. 3.Configuring Cluster Topologies

The content of the configuration file of ElasticSearch is shown in **Code 23** below. There are three types of nodes:

data node
This does not act as the master, and only stores data. When it receives a request from a client, it searches data from shards or creates an index.
master node
It functions to maintain a cluster, and requests indexing or search to data nodes.
Search balancer node
If it receives a search request, it requests data, gathers data and delivers the result.

You can have one node which will function both like a master and a data node. But if you use the three types of node separately, you can reduce the burden of the data node. In addition, if you configure the master node separately, you can improve the stability of a cluster. Also, you can reduce operation costs by using low-spec. server equipment for the master and search node.**Figure 1** below shows the configuration of NELO2 topologies that use ElasticSearch. The efficiency of equipment use and the stability of the entire cluster has been improved as follows: only ElasticSearch runs on the 20 data nodes (server) so that they can achieve sufficient performance, while other daemon server processes in addition to ElasticSearch run on the 4 master nodes and 3 search nodes.

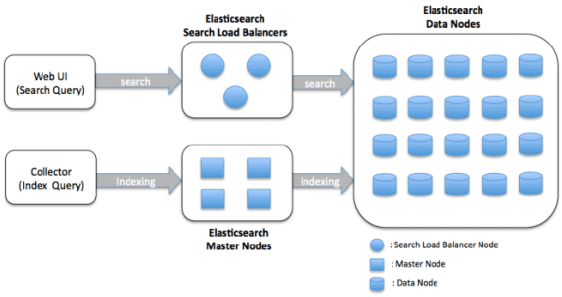

Figure 1… NELO2 ElasticSearch Topologies.

1) Configuring Routing

When a large amount of data needs to be indexed, increasing the number of shards will improve the overall performance. On the other hand, if the number of shards increases, the traffic among nodes will also go up. For example, when there are 100 shards, if it receives a single search request, it sends the request to all the 100 shards and aggregates data, and this imposes a burden on the entire cluster.If you use routing, data will be stored only in a specific shard. Even if the number of shards increases, the application will still send a request only to a single shard, and consequently the traffic can be reduced dramatically. **Figure 2,3 and 4** are excerpted from the slides RafalKuc presented at Berlin Buzzwords 2012. If you don't use routing, as shown in **Figure 2**, the application will send a request to all the shards. But if you use routing, it will send a request only to a specific shard, as shown in **Figure 3**.According to the material cited, in **Figure 4** when there are 200 shards, the response time is over 10 times faster with routing than without routing. If routing is applied, the number of threads will increase by 10 to 20 times compared to when it is not applied, but the CPU usage is much smaller. In some cases, however, the performance will be better when routing is not applied. For a search query whose result should be collected from multiple shards, it could be more advantageous in terms of performance to send the request to multiple shards. To complement this, NELO2 determines the use of routing depending on the log usage of the project.

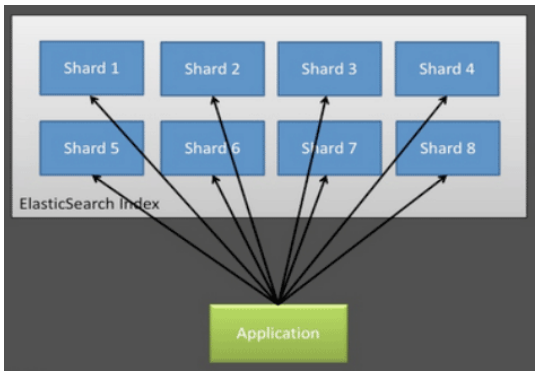

Figure 2….Before Using Routing.

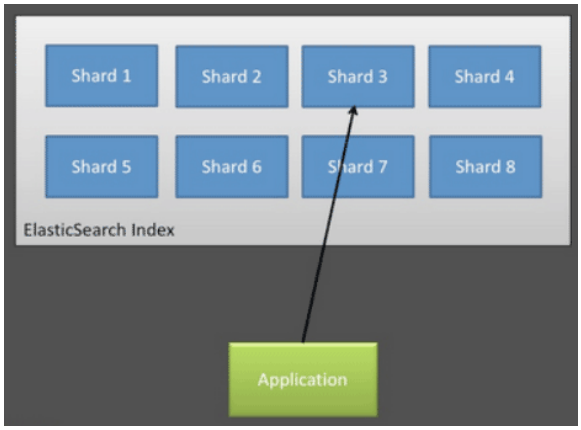

Figure 3… After Using Routing

Queries without routing (200 shards, 1 replica)					
#threads	Avg response time	Throughput	90% line	Median	CPU Utilization
1	3169ms	19,0/min	5214ms	2692ms	95 – 99%

Queries with routing (200 shards, 1 replica)					
#threads	Avg response time	Throughput	90% line	Median	CPU Utilization
10	196ms	50,6/sec	642ms	29ms	25 – 40%
20	218ms	91,2/sec	718ms	11ms	10 – 15%

Figure 4….Statistics before/after Routing

IV. CONCLUSION

After studying the way ElasticSearch works and the different Optimization API it provides, we can conclude that in comparison to all other open source search techniques available, this is the most reliable and the fastest to work with. We also no its clear advantages over Apache Solr. Thus, for this project, ElasticSearch is the most suitable search technique.

ACKNOWLEDGMENT

We would like to thank our guide, Prof. ChetashriBhadane for all her help and support. We would also like to thank the Head of Department, Computer Engineering, Dr. NarendraShekokar. Our sincere thanks to the entire teaching staff of our college.

REFERENCES

[1]All the documentation provided by www.elasticsearch.com
[2]Blogs by the owner of ElasticSearch, Mr. Shay Banon.
[3] http://proceedings.spiedigitallibrary.org/proceed
[4]http://www.elasticsearch.org/tutorials/javascript-web-applications-and-elasticsearch/
[5]http://euphonious-intuition.com/2012/07/an-introduction-to-mapping-in-elasticsearch/
[6]https://engineering.aweber.com/using-elasticsearchs-aliases/

C. Bhadane
1. M.Tech. (Computer Engg.) NMIMS Mumbai. NPSTME, Mumbai in 2012 with CGPA 3.74/4.
2. B.E. Computer Engg. NMU – Jalgaon with 1st Class, (64.00%), May' 2003.
3. 7 years teaching experience
4. International Paper Publication: Paper titled "An Efficient Parallel Approach for Frequent Itemset Mining of Incremental Data" is Published in International Journal of Scientific & Engineering Research (IJSER) in Feb 2012, Issue ., ISSN 2229-5518.
5. National Paper Publication at Conferences:
a. Got Merit certificate for the paper titled "The role of RTOS in 3G handsets".
b. Paper titled "An Efficient Approach for Incremental Data Mining for Frequent Itemsets" is presented I National Conference at Nashik, 2011.

H. A. Mody
Undergraduate student at Dwarkadas J. Sanghvi College of Engineering, University of Mumbai.

D. U. Shah
Undergraduate student at Dwarkadas J. Sanghvi College of Engineering, University of Mumbai.

P. R. Sheth
Undergraduate student at Dwarkadas J. Sanghvi College of Engineering, University of Mumbai.