A Review of Full-Text Search in Different Perspective

Chattar Singh
International School of Informatics and Management
Jaipur, Rajasthan
Rajasthan Technical University,
Kota, Rajasthan
chattarsinghr@gmail.com

Dr.Vijay Gupta
International School of Informatics and Management
Jaipur, Rajasthan
Rajasthan Technical University,
Kota, Rajasthan
vijaygupta1@gmail.coms

***Abstract* -** Searching is the most important concept in case of the information systems. In order to perform the search, it requires an effective index formation and quick and effective retrieval of desired data. Full-Text search is advance concept of searching documents and the database related records. Full-Text search is responsible for the better search management by forming index, based on the words presents in documents. This paper reviews the concept of the Full-Text search, types and also Lucene Full Text Search.

Keywords - Full-Text Search, Lucene Search, Unstructured Data.

1. INTRODUCTION

Full text search is a method, which permits directing search through records and databases by a title, yet additionally by content. Dissimilar to metadata search techniques, which investigate just the portrayal of the archive, full text search experiences all the words in the report, demonstrating data that is more important or some more specific data. The procedure picked up its ubiquity in 1990's. Around then the way toward filtering was exceptionally long and tedious, so it was upgraded.

Full text search motors are utilized broadly. For instance, Google permits clients to discover the needed question on pages especially with the assistance of this strategy. In the event that their website with a ton of data, applying full text search may be extremely valuable since it facilitates connection for a client.

Utilizing a text index can be quicker than utilizing a customary index to discover lines containing a given worth. Full text search capacity in SQL Anywhere contrasts from searching utilizing predicates, for example, LIKE, REGEXP, and SIMILAR TO, on the grounds that the coordinating is term-based, not example based.

String examinations in full text search utilize all the ordinary grouping settings for the database. For instance, in the event that the database is designed to be case unfeeling, at that point full text searches will be case uncaring (Aruleba, Aremu, Oriogun, & Agbele, 2015).

2. REVIEW OF LITERATURE

(Wang Long and Wan Zhenkai, 2009) Designed the secure P2P File Sharing System based on Full-Text Retrieval. The proposed system is able to find the keywords as per user requirement and apart for the features of full-text search, the proposed system also integrated with the ID based authentication system. The experimental analysis of system is performed on large scale network and results are quite efficient as compared to earlier approaches.

(Shengdong Li et.al, 2009) In this author developed the full-text search based information accessing system, which is based on the lucene concept. Authors have a modified approach by increasing the index buffer size and reducing the frequency of write operations for index. The system is simulated for the 70000 documents and the search efficiency found is 15.6 % that the previous approaches and similarly the index creation efficiency also improved by 55.1 %.

(A. Arslan and O. Yilmazel, 2010) Authors proposed and compared the full text search approach inspired by Lucene full search concept. In this author make use to two relational database management systems and also by making use of the Stemming algorithms like Porter algorithm, Lovins etc. In this study, authors performsthe TREC-4 adhoc by performing the query of forty nine topics over the 567,529 of the document for the purpose of the comparison for the search effectiveness and also for the search efficiency.
(L. Qian and L. Wang, 2010) Authors compared the performance evaluation for the internet applications like instant messaging, twitter, Google buzz and so on. For this purpose, authors used and studies the Lucene based indexing and searching for the short-text. For the purpose of the performance evaluation authors compared the Lucene approach with the Oracle text and with the result analysis proves the better performance aspects of the Lucene approach for the short text based search.
(Y. Ding, K. Yi and R. Xiang, 2010) Authors created the full-text search based on the lucene approach in which two main segments are of importance, index creation and index search. Here, the author created the two query modes, one is C/S for the index repository of Lucene and databases which is created in Java and the other one is B/S which is query method for users. The system is empowered with the duplicate paper detection system which isbased on Lucene. Apart from this,the implementation is performed for the highlighting of keywords and also the Chinese information index is also created.
(S. Lincheng, 2011) In order to address the search engines cost issues for the small and medium sized organizations, authors employ the DotLucene based techniques in order to implement the full-text search engine. From the results which are obtained from the experimental analysis of search space has almost 530,000 records, the full-text search engine which is using the concept of Lucene is much more effective in performing search as compared to the traditional concept of fuzzy search based system.
(K. N. Aye and N. L. Thein, 2011) In this authors suggests new concept for information retrieval, in case of the unstructured data. For this purpose, authors proposed an efficient indexing and searching relating framework which will be used for the retrieval of the unstructured data. They used the text-based as well as the content-based approaches for the purpose of the unstructured data retrieval.
(M. Li et. al, 2012) Developed a web crawler optimization algorithm inspired by features of Pango and Lucene.Net to create efficient full-text search functions. The authors analyzed various characteristics of news-based sites to integrate these features into traditional vertical search engines. This integration of Pango helped build content for news and improved efficiency for full-text searches.
(A. Latreche and L. Guezouli, 2012) Here authors worked on the XML documents, these are semi-structured in nature and are difficult to work with. Authors present the new concept of the similarity measure, which is being inspired by CASIT model.
(J. Chen and R.Huang, 2013) Authors developed the multithreaded crawler in order to implement the web information based crawling, and this system used the lucene, search concept, thus implemented the data indexing and used that index to retrieve information.
(J. Huang et. Al, 2013) This paper introduces clinical IR systems utilizing query domain ontology closely tied to the query context.
(K. Ma et.al 2013) The authors proposed a structure-independent hierarchical document model to represent hierarchical documents. They suggested a transformation engine for translating rich files into the models. The authors also employed a core log event listener to capture document changes and maintain the index for storage purposes.

(M. Li and S. Cao, 2014) In this paper, authors proposed the cloud storage based on the HDFS features for the purpose of the massive information management and also for the application. The model is based on the dis-tributed information based retrieval system was proposed.
(X. Shi and Z. Wang, 2014) This paper proposes an optimized full-text search system that combines the Lucene text-based search engine with an Oracle database. Efficient performance was observed in both data indexing and retrieval. The proposed solution improves performance for index creation and reduces the time consumed in the process. The recall results also show improvement.
(A. Tamrakar and S. K. Vishwakarma, 2015) The authors suggested a probabilistic model for an IR solution. The main approach is to determine the probability of a document being relevant to a user's query. Each query is associated with an ideal answer set, which is selected based on the user's query. This approach was analyzed using the FIRE 2011 dataset.
(F. Ramli et al., 2016) The authors present an ontology-based approach for designing and developing a new representation for IR systems to overcome the challenges of conventional keyword-based approaches. The dataset comprises historical documents related to the Vietnam War.
(B. I. Ismail et al., 2017) This paper presents an informatics description of search engines, detailing how data is maintained and flows within them. It also explains the processes of data acquisition and retrieval using the search engine.
(I. Safder and S. Hassan, 2018) The authors proposed a deep learning-based system that enhances search mechanisms by classifying algorithm-specific metadata, such as accuracy, precision, and recall. This approach uses frequent terms as in 'bag of words' models.

3. TYPES OF FULL-TEXT SEARCH

A. Term and phrase search

When playing out a full text search for a rundown of terms, the request for terms isn't significant except if they are inside a phrase. On the off chance that you put the terms inside a phrase, the database worker searches for those terms in the very same request, and same relative situations, in which you determined them (Moskovitch, R., Martins, S. B., Behiri, E., Weiss, A., & Shahar, Y. 2007).

B. Prefix search

The full text search highlight permits to search for the starting segment of a term. This is known as a prefix search. To play out a prefix search, indicates the prefix that is needed to search for, trailed by a mark. This is known as a prefix term (Moskovitch, R., Martins, S. B., Behiri, E., Weiss, A., & Shahar, Y. 2007).

C. Proximity search

The full text search include permits to search for terms that are close to one another in a solitary segment. This is known as a proximity search. To play out a proximity search, which indicates two terms with either the watchword NEAR between them, or the tilde (~)? (Lin, J. 2009).
It can utilize a number contention with the NEAR catchphrase to indicate the greatest separation. For instance, term1 NEAR[5] term2 discovers occasions of term1 that are inside five terms of term2. The request for terms isn't critical; 'term1 NEAR term2' is equal to 'term2 NEAR term1'.

D. Boolean search

The numerous terms are isolated by Boolean administrators when performing full text searches. SQL Anywhere underpins the accompanying Boolean administrators when playing out a full text search: AND, OR, AND NOT (Lin, J. 2009).

E. Fuzzy search

Fuzzy searching can be utilized to search for incorrect spellings or varieties of a word. To do as such, utilize the FUZZY administrator followed by a string in two fold statements to locate a surmised coordinate for the string. For instance, CONTAINS (Products, Description, 'FUZZY "cotton"') returns cotton and incorrect spellings, for example, coton or cotton (Lin, J. 2009).

F. View search

To utilize a full text search on a view or determined table, you should manufacture a text index on the segments in the base table that you need to play out a full text search on. The accompanying assertions make a view on the Marketing Information table in the example database, which as of now has a text index name, and afterward play out a full text search on that view.

4. POPULAR FULL-TEXT SEARCHES

- Apache Solr: Built on Lucene, Solr is an open-source search platform designed for scalability, fault tolerance, and distributed indexing. It provides advanced full-text search, faceted search, real-time indexing, and dynamic clustering.
- ArangoDB: ArangoDB is a multi-model database that supports full-text search capabilities. It allows users to run full-text queries across its integrated graph, document, and key-value data models.
- BaseX: BaseX is a native XML database that supports full-text indexing and search. It is optimized for querying XML data, providing efficient search capabilities within XML documents.
- Lucene is a very high-performance, full-featured text search engine library that serves as the base for many applications developed with the purpose of indexing and searching text.
- mnoGoSearch is a tool of groupware and web server query. This is full-text search software used in the web for indexing and searching documents and web pages by application.
- Searchdaimon: It is an enterprise full-text search appliance, focused on the indexing of documents, emails, databases, and other sources of information to support full-text searches.
- Sphinx: Sphinx is a free, open-source, full-text search application server that is majorly developed to provide easy and fast searching procedures for the retrieval of information. It is, therefore, often installed to offer search possibilities on MySQL, PostgreSQL, or other databanks.
- Elasticsearch: It is a distributed, RESTful search and analytics solution, an open-source mechanism with Apache Lucene. In use-cases, it is proved to be highly scalable and fast, hence can sustain heavy loads of data. It can complete full-text search capability in relation to analytics.
- KinoSearch: An open-source search engine library modeling its API after Apache Lucene. It aims to offer full-text search facilities employing ranked searching, field-specific queries, sorting, and filtering. Although the KinoSearch project is now largely succeeded by the Apache Lucy project, it has served as a Perl-based search engine alternative in its prime.
- Lemur/Indri: The Lemur Project is a language modeling and information retrieval research tool. Indri is one of the search systems being developed as part of the Lemur project. It is specifically aimed at supporting much more powerful search and analysis applications, including passage retrieval and cross-language information retrieval. Indri uses language modeling techniques and has available more advanced features at the top levels — for instance, field-based retrieval, structured queries, and distributed retrieval.

5. LUCENE SEARCH

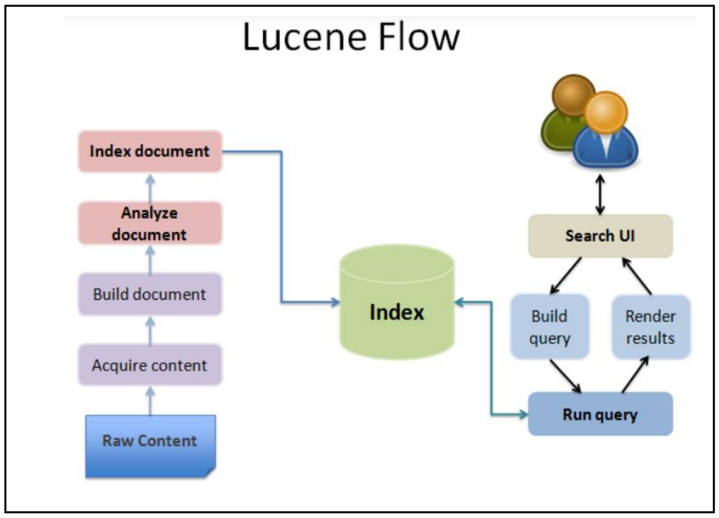
Lucene is an open-source search library, written in Java and maintained by the Apache Software Foundation. This library provides a very flexible and efficient way to implement full-text search capabilities within different types of applications. Lucene gives an advanced feature set that will aid developers in building sophisticated search functionality. This will provide a framework with features in indexing, searching, and analysis that facilitate the processing of vast reams of data and the retrieving of specific information. It indexes documents in many formats: text, HTML, and PDF. So, it's quite versatile for different scenarios.

Lucene, by default, uses the document-based indexing model. Basically, any document consists of several fields, so a developer can structure the data in the way needed. In Lucene, the indexing process entails tokenization of the text and filters applied to it before the storage of resulting tokens in the inverted index, which maps terms to documents in which they occur. Such a structure backs up very fast document retrieval by search queries. It also sustains all types of queries—term queries, phrase queries, and wildcard queries—which might be combined with Boolean operators to execute complex searches.

One of the central strengths of Lucene is the result of relevant searches through scoring and ranking algorithms. This means that documents are scored for relevance against the query, considering field boosts, inverse document frequency, and word frequency. More relevant documents will always be to the top based on some form of grading system. In addition, Lucene makes it easy to have several scoring systems plugged in and further allows the developer to tune their search results.

A. Figures

Fig. 1. Lucene Search



Lucene provides a number of sophisticated functions, including spell checking and highlighting, along with faceted search. Faceted search makes it easier for the user to narrow down the results by specifying price ranges or categories. Highlighting allows the user to see exactly what the search terms matched in the search result. This is good for evaluating if the document is relevant and needs to be opened. It corrects misspellings in the user's query, improving the search experience. Lucene is extremely popular for building search engines and is used as a base for bigger search platforms due to the flexibility and extensibility it offers, such as Elasticsearch and Apache Solr..

6. CHALLENGES OF FULL TEXT SEARCH

To comprehend and like the systems and terms engaged with full-text-searching comprehend the difficulties engaged with the equivalent. A couple of the basic difficulties are.

- Scale - How accomplishes it work for gigantic measures of data?
- Proximity - How to make the search results significant?
- Taking care of Synonyms, Abbreviations, homonyms, Phonetics
- Taking care of incorrect spellings
- Searching through pictures
- Re-indexing - Handling new or continually evolving data?
- FTS is appropriated Systems.

7. CONCLUSION & FUTURE SCOPE OF THE STUDY

With the growth of data, it is required to have an efficient information retrieval system, which will retrieve the proper data and in the efficient way. In this paper the concept of full-text search is discussed as well as its features and drawbacks/challanges and in the later research we will like of work on these drawbacks and will try to propose a new full-text search based on lucene, with overcoming more of these challenges and issues.

REFERENCES

1. Aruleba, K., Aremu, R. D., Oriogun, P., & Agbele, K. K. (2015). Evaluation of full text search retrieval system. *Nigeria Computer Society*, 154–159.
2. Moskovitch, R., Martins, S. B., Behiri, E., Weiss, A., & Shahar, Y. (2007). A comparative evaluation of full-text, concept-based, and context-sensitive search. *Journal of the American Medical Informatics Association, 14*(2), 164–174. https://doi.org/10.1197/jamia.M2141
3. Lin, J. (2009). Is searching full text more effective than searching abstracts? *BMC Bioinformatics*. https://doi.org/10.1186/1471-2105-10-86
4. Wang, L., & Wan, Z. (2009). Secure P2P file sharing system based on full-text retrieval. In *Proceedings of the 1st International Conference on Information Science and Engineering (ICISE)*.
5. Li, S., Lv, X., Ling, F., & Shi, S. (2009). Study on efficiency of full-text retrieval based on Lucene. In *Proceedings of the 2009 International Conference on Information Engineering and Computer Science* (pp. 1–4). Wuhan.
6. Arslan, A., & Yilmazel, O. (2010). Quality benchmarking relational databases and Lucene in the TREC4 adhoc task environment. In *Proceedings of the International Multiconference on Computer Science and Information Technology* (pp. 365–372). Wisla.
7. Qian, L., & Wang, L. (2010). An evaluation of Lucene for keywords search in large-scale short text storage. In *Proceedings of the 2010 International Conference on Computer Design and Applications* (pp. V2-206–V2-209). Qinhuangdao.
8. Ding, Y., Yi, K., & Xiang, R. (2010). Design of paper duplicate detection system based on Lucene. In *Proceedings of the 2010 Asia-Pacific Conference on Wearable Computing Systems* (pp. 36–39). Shenzhen.
9. Lincheng, S. (2011). A large-scale full-text search engine using DotLuence. In *Proceedings of the 2011 IEEE 3rd International Conference on Communication Software and Networks* (pp. 793–795). Xi'an.
10. Aye, K. N., & Thein, N. L. (2011). Efficient indexing and searching framework for unstructured data. *Proceedings of SPIE - The International Society for Optical Engineering*.
11. Li, M., Gu, X. J., & Yang, Z. X. (2012). Research of vertical search engine in news industry. In *Proceedings of the 2012 International Symposium on Management of Technology (ISMOT)* (pp. 253–256). Hangzhou.
12. Latreche, A., & Guezouli, L. (2012). Similarity measure for semi-structured information retrieval based on the path and neighborhood. In *Proceedings of the 2012 International Conference on Information Technology and e-Services* (pp. 1–5). Sousse.
13. Chen, J., & Huang, R. (2013). A price comparison system based on Lucene. In *Proceedings of the 2013 8th International Conference on Computer Science & Education* (pp. 117–120). Colombo.
14. Huang, J., Daoud, M., & Ye, Z. (2013). Exploiting semantics for improving clinical information retrieval. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

15. Ma, K., Yang, B., & Abraham, A. (2013). Toward full-text searching middleware over hierarchical documents. In *Proceedings of the 2013 13th International Conference on Intelligent Systems Design and Applications* (pp. 194–198). Bangi.
16. Meng, L., & Cao, S. (2014). A series method of massive information storage, retrieval and sharing. In *Proceedings of the 2014 IEEE International Conference on Mechatronics and Automation* (pp. 1171–1175). Tianjin, China.
17. Shi, X., & Wang, Z. (2014). An optimized full-text retrieval system based on Lucene in Oracle Database. In *Proceedings of the 2014 Enterprise Systems Conference* (pp. 61–65). Shanghai.
18. Tamrakar, A., & Vishwakarma, S. K. (2015). Analysis of probabilistic model for document retrieval in information retrieval. In *Proceedings of the 2015 International Conference on Computational Intelligence and Communication Networks (CICN)* (pp. 760–765). Jabalpur.
19. Chen, X., & Xu, L. (2016). An educational resource retrieval mechanism based on Lucene and topic index. In *Proceedings of the 2016 13th Web Information Systems and Applications Conference (WISA)* (pp. 125–128). Wuhan.
20. Ramli, F., Noah, S. A., & Kurniawan, T. B. (2016). Ontology-based information retrieval for historical documents. In *Proceedings of the 2016 Third International Conference on Information Retrieval and Knowledge Management (CAMP)* (pp. 55–59). Bandar Hilir.
21. Ismail, B. I., Kandan, R., Goortani, E. M., Mydin, M. N. M., Khalid, M. F., & Hoe, O. H. (2017). Reference architecture for search infrastructure. In *Proceedings of the 2017 7th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)* (pp. 115–120). Penang.
22. Ismail, B. I., et al. (2018). A proposed framework for search as a service on private cloud. In *Proceedings of the 2018 IEEE Conference on Open Systems (ICOS)* (pp. 7–12). Langkawi Island, Malaysia.
23. Lakhara, S., & Mishra, N. (2017). Design and implementation of desktop full-text searching system. In *Proceedings of the 2017 International Conference on Intelligent Sustainable Systems (ICISS)* (pp. 480–485). Palladam.
24. Safder, I., & Hassan, S. (2018). DS4A: Deep search system for algorithms from full-text scholarly big data. In *Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW)* (pp. 1308–1315).
25. Jabbar, J., JunSheng, W., Weigang, L., & Urooj, I. (2019). Implementation of search engine with Lucene in the document management system. In *Proceedings of the 2019 IEEE 2nd International Conference on Electronics and Communication Engineering (ICECE)* (pp. 1–4).
26. Yu, J., Su, A., Liu, W., Cheng, X., & Yang, J. (2019). Thematic learning-based full-text retrieval research on British and American journalistic reading. In *Proceedings of the 2019 14th International Conference on Computer Science & Education (ICCSE)* (pp. 611–615).
27. Youzhuo, Z., Yu, F., Ruifeng, Z., Shuqing, H., & Yi, W. (2020). Research on Lucene based full-text query search service for smart distribution system. In *Proceedings of the 3rd International Conference on Artificial Intelligence and Big Data (ICAIBD)* (pp. 338–341).