IJDCST @November Issue- V-1, I-7, SW-20
ISSN-2320-7884 (Online)
ISSN-2321-0257 (Print)

Fuzzy Driven Range Queries over Extended Boolean Retrievals

[1]M VVNS Narayana, [2]IMV Krishna.

[1]Student, Dept. of CSE, PVPSIT, Kanuru, Vijayawada, AP, INDIA

[2]Assistant Professor, Dept. of CSE, PVPSIT, Kanuru, Vijayawada, AP, INDIA

ABSTRACT: Efficient Extended Boolean Retrieval (EBR) models based on stemming methods such as Query Tree Scorings ensured significant quantitative and qualitative query results compared to plain keyword, ranked keyword or pure Boolean retrievals. Although this query model allows the representation of complex concepts in an and / or format combined with p-norm approaches, screening process, term independent bounds, their practical implementation is a complicated process. Stemming techniques to implement screening and scoring methods to support Efficient EBR's are irrespective of term lengths. Therefore, the computations and processing involved for a single keyword query will be similar to that of multiple search term queries. Searching and retrieving a stemmer reduces the query to its word root form and matches results containing this stem, which has its side effects. We propose to implement Iterative Range-Search-Based Algorithm to support Fuzzy Search in String Collections to reduce number of disk seeks to support multiple terms. Experimental query models based on proposed system suggests Fuzzy EBR's results are better than prior EBR's. A practical implementation of the proposed system validates the claim.

Keywords: EBR, IRSBA, Plain Keyword, Ranked Keyword, Range Queries.

I. INTRODUCTION

Search service providers have an interest in delivering competitive effectiveness levels within the smallest possible resource cost. Information retrieval was one of the first areas of natural language processing in which statistics were successfully applied.

Auto completion is a widely used mechanism to get to a desired piece of information quickly and with as little knowledge and effort as possible. Unix Shell is the early usage of pressing the tabulator key gives a list of all file names that start with whatever has been typed on the command line after the last space. We find a similar feature in most text editors and in a large variety of browsing GUIs. Auto completion has been integrated into a number of search engines like Google Suggest or Apple's Spotlight. The list of completions is simply a range from a list of words. This is the list of all words entered into the file so far for the text editors. For these kinds of applications, we can easily achieve fast response times by two binary or B-tree searches in the sorted list of candidate strings.

Ranked retrieval has been successfully deployed in a wide range of applications. The advantage of the ranking is the simplicity of querying and that results are ordered by estimated relevance. Having the answers returned as a ranked list also gives users the ability to consciously choose the amount of effort they are willing (or able) to invest in inspecting result documents. Advantages of Boolean retrieval include:

Complex information need descriptions: To express the complex concepts Boolean queries are used.

Composability and Reuse: The concept of Boolean filters itself can be recombined into larger query tree structures.

Reproducibility: Document scoring only depends on the document itself and can be reproduced with knowledge of the query.

Scrutability: Properties of resultant documents can be understood simply by examination of the query.

Strictness: Exclusion criteria and strict inclusion are inherently supported based on metadata.

Boolean retrieval and the extended Boolean variant remains a critically important retrieval mechanism. Particularly when there are exclusion criteria as well as inclusion criteria and ranking over bags of words is not appropriate. More advanced forms of auto completion take into account the context in which the to-be-completed word has been typed. Imagine a user of a search engine typing a query, then with every letter being typed. The best hits for any of these completions should be displayed at the same time.

II. RELATED WORK

The auto completion feature as described so far is reminiscent of stemming too prefixes instead of full words are considered. Our auto completion feature gives the user feedback on which completions of the prefix typed so far would lead to highly ranked documents. The user can then assess the relevance of these completions to his or her search desire. While our auto completion feature is for finding information, auto completion has also been employed for predicting user input.

Contextual information has been used to select promising extensions for a query. Our auto completion problem is related to but distinctly different from multidimensional range searching problems. These data structures could be used for our auto completion problem if we were willing to limit the number of query words. We can achieve fast query processing times and space efficiency at the same time, we have the set of documents matching the part of the query before the last word already computed. Our auto completion problem is therefore a 1 1/2 - dimensional range searching problem. We have considered those we are aware of with regard to their applicability to our auto completion problem. However, we found them either unsuitable or inferior to the inverted index in that respect.

Authors of systematic reviews seek to identify as much as possible of the relevant literature in connection with some aspect of medical observations, typically a highly specific clinical inquiry. To provide a "best currently known" summary of knowledge and practice in that field, synthesize the evidence contained in a set of identified documents. A variety of organizations provides central points of call for systematic reviews the largest of these efforts and the Agency for Healthcare Research and Quality.

To construct each systematic review a complex Boolean search query is used to retrieve a set of possibly relevant documents. The Boolean query might consist of as many as several dozen-query lines describing a concept with fielded keywords MESH heading, metadata, free-text term expansion. Reproducibility and Scrutability are key requirements for the document selection process and to this end; the queries used are usually included verbatim in the review document.

The complexity of queries is then further increased using wildcard expansion terms based on syntactic similarities. Queries might become as large as a thousand terms leading to the inescapable conclusion that efficient evaluation techniques are required. One of the reasons that key documents might be missed is that the queries have to be specific enough that they retrieve a set of documents that it is feasible to exhaustively review.

Extended Boolean retrieval has been shown to provide advantages over pure Boolean retrieval in these two regards. Compares the retrieval performance of complex structured Boolean queries with extended Boolean queries for which the queries have been transformed to more balanced query trees containing only the three basic Boolean operators.

III. EXISTING SYSTEM

Typical search query is a query that a user enters into search engine to satisfy his or her information needs. These queries are classified into plain keyword, ranked keyword, or pure Boolean retrievals, Extended Boolean Retrievals (EBR). Efficient retrieval systems are required to support complex queries by professional searchers, possibly with significant commercial or societal outcomes resting on the results of the search. Ranked retrieval has been

successfully deployed in a wide range of applications however the results are determined by access popularity and not because of their relevancy. Boolean retrievals have not been superseded, and are still the preferred method in domains such as legal and medical search. Advantages of Boolean retrieval include: Complex information need descriptions: To express the complex concepts, Boolean queries are used; Composability and Reuse: The concept of Boolean filters itself can be recombined into larger query tree structures; Reproducibility: Document scoring only depends on the document itself, not statistics of the whole collection, and can be reproduce with knowledge of the query; Scrutability: Properties of resultant documents can be understood simply by examination of the query; and Strictness: Exclusion criteria and strict inclusion are inherently supported based on metadata.

EBR queries have the disadvantage of being harder to formulate than ranked queries, and, regardless of the level of expertise of the user, have the drawback of generating answer lists of unpredictable length. Screening process, evaluation cost, computations cost involved in results filtering is too high, and obtained results although relevant are too low. A better system is required that can handle the above disadvantages and can generate a better number of relevant results. Search service providers have an interest in delivering competitive effectiveness levels within the smallest possible resource cost. To implement the p-norm approach to EBR and quantify the results based on max-score and wand optimization techniques p-norm - Given a number or a list, it computes the probability that a normally distributed random number will be less than that number. This function also goes by the rather ominous title of the "Cumulative Distribution Function (CDF)" Ranked keyword retrievals can be adapted using optimization techniques to allow selective bypass of irrelevant results via a low-cost screening process. Term independent bounds are able to further reduce the number of score calculations for short, simple queries thus lowering extensive computations for short terms. Involved a Query Tree Scoring algorithm to implement the above screening and scoring methods. This efficient EBR delivers a better number of relevant results and screening process (max-score, wand), evaluation cost (p-norm), computations cost (Term independent bounds) are reduced to manageable numbers.

IV. PROPOSED SYSTEM

The Query Tree Scoring construction is a stemming technique to implement screening and scoring methods to support Efficient EBR's irrespective of term lengths. Searching and retrieving a stemmer reduces the query to its word root form and matches results containing this stem. For example for query 'specially' a stemming algorithm will find the results "especially", "special", "specialize", "specializing", "specification" and other having the root "spec". However if in the query word will be casual mismatch like 'spesial' or 'spetial' the search engine based on a stemming algorithm will show zero results. Fuzzy technique uses approximate full text search and retrieval. This means it will match all possible results for a search query despite its form or spelling mistakes/mismatches presence no matter what part of word they will be in. This way it will retrieve "special" even either your query will be "spesial", "spetial" or "spizial". It will show all related results by relevancy and similarity degree. Typically, such fuzzy implementations include some form of distance and score computations between the specified word and the words in the corpus. We propose to implement Iterative Range-Search-Based Algorithm to support Fuzzy Search in String Collections to reduce number of disk seeks to support multiple terms. Experimental query models based on proposed system suggests Fuzzy EBR's better than prior EBR's.

Iterative-Range-Search (IRS) Based Algorithm:

We study how to answer a ranking query by answering (possibly multiple) range selection queries. In each selection query has a threshold on the similarity between the given string and a string in the collection. We can leverage existing approximate-string-selection techniques without modifying their implementations. We start with an initial similarity

threshold, which could be a fixed value or a value computed based on the query.

Step 1 (from line 6 to 10): The goal of this step is to compute at least f · k results, where the multiplication factor is $f \geq 1$. We call a function "ApproxRangeSearch" to run an approximate-string-range search algorithm of our choice. We decrease the similarity threshold, depending on the number of results we got.

Step 2 (from line 11 to 16): We compute the score for each element computed in step 1 and keep the first k elements ordered by their scores. We want to be certain that these k elements are indeed the best results. Assume the element e that was not seen before and it has the maximum possible weight in the data set. Compute how similar e needs to be to the query in order to have a better score than the current k^{th} element.

Advantages: The IRS algorithm has the advantage that it can utilize any of the existing algorithms for approximate-string range search. Hence, it is easy to implement as it uses the range-search algorithm as a black box function.

Algorithm:

Iterative-Range-Search (IRS) Based Algorithm:

```
 1:  Let k be the number of results requested;
 2:  Let W_max be the maximum weight of a string in the data set;
 3:  Let f ≥ 1 be a multiplication factor;
 4:  Let R ← φ be the range-search-result set;
 5:  Let τ be the initial similarity threshold;
     {Step 1: Computing initial candidates}
 6:  while size(R) < f · k do
 7:  R ← ApproxRangeSearch(τ );
 8:  if size(R) < f · k
 9:  then Decrease τ ;
10:  end while
     {Step 2: Finalizing results}
11:  Compute scores for elements in R and keep the first k;
12:  Let τ1 be the minimum similarity for which
     Score(τ1 , Wmax ) > Score(R[k]);
13:  if τ1 < τ then
14:  R ← ApproxRangeSearch(τ1);
15: Compute scores for elements in R and keep the first k;
16: end if
17:  Return R[1..k];
```

V. EXPERIMENTAL ANALYSIS

Each index is stored in a single file with the individual lists concatenated and an array of list offsets at the end. The vocabulary is stored in a separate file. We ensured that the index was not cached in main memory. The first collection is a several encyclopedias on Allopathic medicine (pubmed data set). This collection has been searchable via our engine by different key words over the past months by an audience of several people. Second collection consists of the complete dumps of the computer programming Wikipedia.

From each of these queries, the result of system has sequence of files and its paths of given data set based on input key words. Auto completion queries were generated in the same manner from a set of 100 randomly generated queries, with a distribution of the number of query words and of the term frequency similar to that of the real queries for the Allopathic collection. As we search the word "diabetic blood"

over the collected data set. The query has been generated and the resulted is tabulated as shown in the Table.1.

Testing Search Key word: diabetic blood	No. of Files Found		Time in seconds
Existing System	And	2	0.195
	Or	113	0.165
Proposed System	202		14.563

Table 1: The results are evaluated by performing of Iterative- Range-Search (IRS) Based Algorithm.

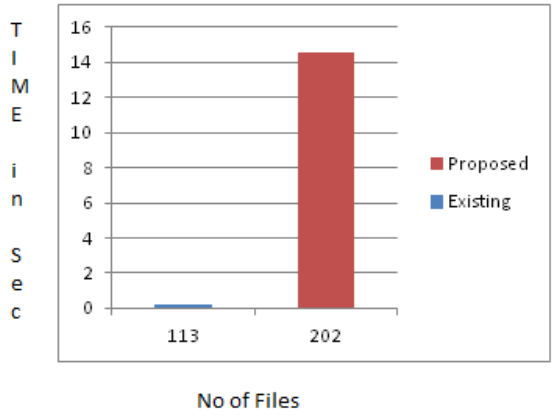

Fig 1: Average running time for Existing and Proposed Systems on PUBMED data set.

A number of observations can be made from the results. Although the CalcScore () scoring method requires partial sorting of tree node identifiers during document scoring. It has comparable execution times to a straightforward implementation that recursively iterates the query tree and computes scores for the next document in the OR set of that subtree (Tree Iteration), often being faster. The scoring method in the adaptation of maxscore yields significant reductions in the number of candidate documents scored and execution times for all query sets. The more documents are to be retrieved the less effective, the top-k optimizations are execution times are significantly below the baselines. The intuition behind the need for a second step is shown by our result. Suppose the first step returns the results with ids between 1 and 4. Current top-2 results are 1 and 3. Second step is needed in order to capture elements that have a high similarity but a low weight, like element with id 5. Finally, top-2 results are 1 and 5.

VI. CONCLUSION

Having noted that ranked keyword querying is not applicable in complex legal and medical domains because of their need for structured queries including negation and for repeatable and scrutable outputs. We have introduced a fuzzy implementation of distance and score computation feature for full text search, and presented a new compact indexing data structure for supporting this feature with very fast response times. The next logical step following this work would be to conduct a user study for verifying that belief, gives the interactivity of the engine. Answering such queries is important to many applications such as record linkage, where there is a mismatch between a user query and the representations of the entities the user is looking for. Our extensive experiments on real data sets showed that these algorithms could answer Fuzzy Driven Range Queries efficiently on large data sets.

VII. ACKNOWLEDGEMENT

I would like to thank M.V Rama Krishna, HOD CSE, PVPSIT, Kanuru, Vijayawada for his extreme guidance and support.

VIII. REFERENCEs

[1] Stefan Pohl, Alistair Moffat, and Justin Zobel, “Efficient Extended Boolean Retrieval,” IEEE Transactions on Knowledge and Data Engineering, Vol. 24, No. 6, PP: 1014-1024, June 2012

[2] Holger Bast, Ingmar Weber, “Type Less, Find More: Fast Autocompletion Search with a Succinct Index,” *SIGIR’06,* Seattle, Washington, USA. Copyright 2006 ACM, PP: 364-371, August 2006.

[3] Rares Vernica and Chen Li, “Efficient Top-k Algorithms for Fuzzy Search in String Collections,” KEYS’09, June 28, 2009, Providence, Rhode Island, USA. Copyright 2009 ACM.

[4] S. Karimi, J. Zobel, S. Pohl, and F. Scholer, "The Challenge of High Recall in Biomedical Systematic Search," Proc. Third Int'l Workshop Data and Text Mining in Bioinformatics, PP: 89-92, Nov. 2009.
[5] J.H. Lee, "Analyzing the Effectiveness of Extended Boolean Models in Information Retrieval," Technical Report TR95-1501, Cornell Univ., 1995.
[6] G. Salton, E.A. Fox, and H. Wu, "Extended Boolean Information Retrieval," Comm. ACM, Vol. 26, No. 11, PP: 1022-1036, Nov. 1983.
[7] H. Turtle and W.B. Croft, "Inference Networks for Document Retrieval," Proc. 13th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval, PP: 1-24, 1990.

About Authors

MVVNS Narayana received the MCA Degree from IGNOU, Vijayawada, India. He is currently pursuing M.Tech (CSE) Degree at the Dept of Computer Science & Engineering, PVP Siddhartha Institute of Technology, Vijayawada, AP, INDIA. His interest in Data Mining and Computer Networks

IMV Krishna received the M.Tech (CSE) Degree from RVR & JC College of Engineering, Affiliated to ANU, Guntur, AP, India. He has more than 5 years teaching experience. He is currently working as an Assistant Professor in the Dept of Computer Science & Engineering, PVP Siddhartha Institute of Technology, Kanuru, Vijayawada and it is affiliated to JNTU Kakinada University, AP, India. His interests are Sofware Engineering, Data Mining and Computer Networks.