Proc. 1st International Conference on
Machine Learning and Data Engineering (iCMLDE2017)
20-22 Nov 2017, Sydney, Australia
ISBN: 978-0-6480147-3-7

Query Optimization in Search Engines

Baisakhi Chakraborty[1], Nirjhar Datta Chaudhuri[2], Dambar Chetri[3] and Akshay Naik[4]

[1]Asst. Professor, Department of Computer Science, NIT, Durgapur, India
[2]Student, Department of Computer Science, NIT, Durgapur, India
[3]Student, Department of Computer Science, NIT, Durgapur, India
[4]Student, Department of Computer Science, NIT, Durgapur, India

Corresponding author's E-mail: baisakhichak@yahoo.co.in

Abstract

This paper aims to take a look at the way a search engine works when a user query is posted. The query processing part involves optimization of the user queries so that unnecessary and redundant parts of the query may be removed. Only the keywords are forwarded to the next stage of searching, which decreases the length of the posted query as well as the load on the further stages to increase throughput as well as response time. A number of linguistic aspects like methods of optimization, (namely stop words and name words that can be deleted without any change to the meaning of the original query) have been discussed in this paper. The spellchecking of the user query has been looked upon so that it is lingual and in sync with an English Dictionary.

Keywords: processor, optimization, stop word, name word, spell check.

1. INTRODUCTION

Information Retrieval (IR) works on finding the words or symbols through thousands of predefined strings of texts that match a user's query by Kraft H. Donald et al (2017). Any IR system aims to retrieve information based on the user's requirement. Query processing is a major task in IR systems that uses Corpus of information which is the backbone of any IR system by Goel K and Bhatia P (2016). The main activity of IR systems is query processing and retrieving information from a number of information resources by Lujia M et al (2017). IR systems as in Liddy E (2001) also known as search engines, are the focus of this project. The query optimization procedure adopted in this project aims at linguistically analyzing and modifying the query to improve the processing time and quality of the IR system, while saving on resources such as memory and process cycles. The IR systems can be broadly broken down into, a document processor, a query processor, searching and matching facility, and a ranking mechanism as in Murata T (2013). The IR systems works in conjunction with referential documents known as indexes, generated by the document processor. The generation of these indexes bear similarity with the process of optimizing and generating queries, by the query processor. The searching and matching facility connects the two processes; matching the results in the index against which the query was made. The ranking mechanism rounds up the process, adding the element of relevance, frequency, and other parameters that bear significance to the reliability of the system. The document processor normalizes the document, breaks it down, isolates and meta-tags it to form data structures that can be accessible to all the further processes. Furthermore "tokenizing" terms, deleting stop words and term stemming are all steps taken by the document processor towards the eventual extraction of index entries and building the inverted file which becomes the index. The details have been greatly discussed by Esbitan Samira SY (2012) and Stanford University Press (2008). The document processor also computes weights in order to make the entries more relevant and substantial.

2. RELATED WORK

Query Optimization in Search Engines is a widespread topic with efforts being made to improve timings and load decrease in processing the query. Advanced methods like making the system autonomous where it can learn to improve itself based on self-research are also being implemented.

In the research paper of Huston S and Bruce Croft W (2010), it has been mentioned how the removal of stop words, name words and stop structures can help to optimize query processing. It is further extended to the use of Classifiers CRF++ and YamCha for the purpose of sequential tagging of Stop Structures.
The main aspects of linguistic and grammatical concepts of Query Optimization are as follows:

1. Orthographic -- checking for typos, official variants (e.g., German/Dutch spelling), etc.
2. Morphologic -- including all forms of a given word via linguistic normalization (lemmatization).
3. Syntactic -- entity or phrase extraction, anti-phrasing, removing word-sense ambiguity (orange colour vs. fruit), etc.
4. Semantic -- applying a combination of general and specific thesauri and ontologies, automatic phrasing, etc., to understand the intention of the query.

3. ARCHITECTURE OF PROPOSED SYSTEM

Intuitive IR systems in FAST Search Best Practices (2006) allow for a fair amount of abstraction handling vague and misspelled queries, and even filtering and post processing results for a better navigation in results. The purpose of a search engine is to understand the context of the user’s intent and to model the search around it, thereby returning the most relevant search results. The proposed work is using a rule based optimization technique. Each step is describing a single optimization rule.
A query processor follows seven distinct steps, some of which can be bypassed. They are tokenizing, parsing, and deletion of stop words, term stemming, query creation, query expansion and query weighing. As is apparent, query processing shares some steps with document processors as mentioned in Rosario B (2006) and Ioannidis Yannis E (1996).

a) Tokenizing is carried out on the query string in order to make it consistent and meaningful.
b) Parsing deals with determining the semantics that the queries intend to convey, using logical operators, Boolean operators and special symbols which would bear little to no relevance, otherwise .
c) Stop word removal is a focal point in the process of optimization.
d) Certain words share the basic structure among themselves e.g. analyzed, analysis, analyze share the prefix analy-. Grouping them under the same sub structure i.e. “analy-” helps broaden the search while also reducing the space need to store the individual terms
e) Query creation is a process of restructuring used to encompass all possible meaningful interpretations of the submitted query.
f) Query expansion incorporates synonymous interpretations of the query to improve the quality of the search.
g) Again query weighing is a practice which is used to obtain greater relevance in the search based upon parameters guarding the users’ intent.

Query optimization by Slawski B (2009) is a process intended to make the context of the users' intent clearer to improve the quality of the search. Though supplementary, this phase of the query processing carries its worth through the resources that it helps save. Certain methods used in the optimization process are methods of suggestion. The method of query suggestions pulls up a list of candidate queries, and calculates their weight based on relevance and frequency. Upon completion the candidate queries are presented as a clustered list, based on rank. Also, the relevance of the suggested queries is specified. Suggested searches for related and frequently appearing queries are shown to searchers with the intention of making searching a better experience for the people who use search engines. The structure of suggestions that are presented can be decided upon various considerations such as popularity, subject matter, frequency, key words etc. The objects of considerations in this form of optimization are key words, their combination, popular phrases, and functionally or semantically similar suggestion cluster terms.

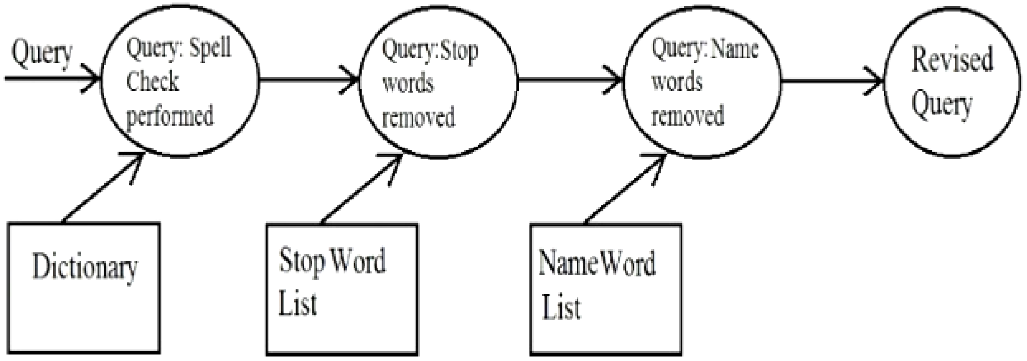

Figure1. Data Flow Diagram of Optimization Techniques used

3.1. Stop Words Removal

Stop words by Natural Language Processing department (2008) are extremely common words which do not serve any distinguishable purpose in the query processing function of an engine. The filtering out of such words plays an important role in query processing since it reduces the size of the query thereby reducing the size of storage and processing time needed for this purpose, without significantly affecting the quality of the results obtained. Search engines maintain a list of such words which they use as a reference in processing the query string. This list is not fixed or standard for a search engine and varies from engine to engine.
For example:

"Can I work while study in Europe?"

This technique produces the query:

"Work study Europe"

For removing the stop words from anywhere in the string, initially a separate text file is made with all the stop words. They are stored such that the actual source code does not need to be accessed every time a stop word has to be added or deleted from the original list. 25 words are chosen from a report as in Natural Language Processing department (2008).

After the stop words are stored, the next stage is to add them to the source code for matching and subsequent removal from the original string. Each of the values in the file are tested and then stored separately as words in an array. Then for each value we encounter in the aforementioned array, the original string is tested and if there is a match with any stop word in the list, the stop word is removed and replaced with a simple space.

3.2. Name Words Removal

Name words also add to the load of a query by adding unnecessary and unrelated question words at the start of a query. For example:

“Where is Delhi?”

This technique produces the query:

“*is Delhi?*”

And a combination of the above two techniques further shortens the query to

“Delhi”

One important factor that is a major difference between stop words and name words is that the name words are only present at the start of questions and hence the name words at the start of the query should be considered and not the ones in between as they may hold a separate meaning, different from a pure name word. The initial stage for Name word removal is the same as that for a Stop word removal, where a separate text file is made to avoid changes to the original source code. Then the name words present in the file are imported and extracted one by one onto an array. Then, they are tested with the original query and if there is a match, the position of the name word within the query is saved. If the position is zero, that is the name word is present at the initial position, then it satisfies the criteria of a name word and hence is removed with the help of the left trim php command ltrim.

3.3. Spellcheck

Spell check in Phpspellcheck (2004) at the GUI end is one of the foremost applications in query processing. Having a corrected query, unless it is a specific name, helps in finding the intended query correctly rather than having a redundant result as a consequence of incorrect spellings. For the purpose of checking spelling, a PHP code known popularly as “Phpspellcheck” has been embedded. It is a very useful code segment that considers a particular text area. The text within it co-relates it with a predefined English dictionary and provides alternatives to the incorrect word and even allows the user to choose among a number of possible alternatives if more than one appears. Phpspellcheck is an easy to use PHP Spell Checker script for websites and intranets. The Phpspellcheck component adds fast, reliable spell-checking to websites and intranets. Installed on either IIS or Apache Server, PHP Spell Check works on all major browsers since IE5. PHP spell check provides "As-You-Type" spellchecking with red-squiggly-underlines as well as a more traditional "Spell-checking Dialog" popup.

4. IMPLEMENTATION

In this paper, the Phpspellcheck in XAMPP-Apache Friends (2015) code has been applied to the GUI page rather than the processing side such that the user can check the spelling on words he has written incorrectly and correct them instantly. The “include.php” file is needed and is imported as this file contains the basic necessities for the implementation of the entire process. The contents of the text area entered in the form are considered for correction by indicating that in the specifications of the imported php file. The final implementation requires hosting the php files, along with the embedded spellchecker, so that they can be accessed and the subsequent files can be found in a single folder location. For this purpose, a XAMPP server has been used that hosts the entire project on port number 80 of the localhost, which is 127.0.0.1. XAMPP, along with WAMPP, is a very useful tool for this very purpose. XAMPP is a free and open source cross-platform web server solution stack package, consisting mainly of the Apache HTTP Server, MySQL database, and interpreters for scripts written in the PHP and Perl programming languages. The entire PHP based implementation of this project can be uploaded in a web server such as Apache HTTP server, WampServer, etc. This PHP based project has been implemented on Intel Corei3 based machine with 4 GB RAM and 500 GB Hard Disk.

5. RESULT

Since most of the applications of a search engine involve the user loads it initially and provides the query accordingly, hence we can say that the first case mentioned in the above table is the most frequent use, plus bringing in the entire page initially uses up comparatively more time than backtracking or refreshing.

Table 1. Time calculation of initial page load with embedded dictionary

Sl. No.	Page load situation	Time(in ms)
1	Initial loading of the page	316
2	First refreshing the page	60
3	Nth refresh of the page	18
4	Backtrack from 2nd page to 1st	15

Table 2. Query processing and result display

Sl. No.	Query situation	Time (in ms)
1	Small query (<10 words) without stop word and name word	1
2	Small query (<10 words) with stop word and name word	1
3	Medium query (100 words) with stop word and name word	1
4	Large query (1000 words) with stop word and name word	2

Thus, we have concurred that the time required for processing the query by subjecting it to the stop word and name word list is very negligible, hardly going past a couple of milliseconds at the most. Maybe with a wider stop word list the time will increase, but when compared to the time required for loading the dictionary in place, it will always be negligible to that.

6. CONCLUSION

Thus, the project we have conducted for the purpose of quantization in search engine query processing appears to be fruitful and the timings we have generated by using the timer and applying the various quantization techniques seem to be within limits and do not add much of an amount to the overall timing of the search process.
Nevertheless, the steps we have conducted are very preliminary steps in comparison to the vastness of all query optimization techniques. Yet our work is completely fault proof and in sync with the original objective of query optimization.

7. REFERENCES

Kraft H D, Colvin E, Marchionini G (2017). Fuzzy Information Retrieval, Synthesis Lectures on Information Concepts, Retrieval, and Services, 1, 63 pp.

Goel K, Bhatia P (2016). Information retrieval system using UNL for multilingual question answering. 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 888 pp.

Lujia M, Wei B, Wugedele B, Wuriga Y, Tao H, Xiaobing Z (2017). A Mongolian Information Retrieval System Based on Solr. 2017 9th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), 335 pp.

Esbitan Samira SY (2012). A Personalized Context-Dependent Web Search Engine Using Word Net (Sama Search Engine), Islamic University of Gaza.

Huston S, Bruce Croft W (2010). Evaluating Verbose Query Processing Techniques, Centre for Intelligent Information Retrieval, Department of Computer Science, University of Massachusetts, Amherst.

Ioannidis Yannis E (1996). Query Optimizayion. ACM Computing Surveys *(CSUR)*, 28 (1), 121-123.

Liddy E (2001). How a Search Engine Works. Director of the Center for Natural Language Processing Professor, School of Information Studies, Syracuse University. Searcher, 9.

Murata T (2013). The Mechanism of the Search Engine. http://wiki.c2.com/?MechanismOfSearch Engine.

Rosario B (2006). Applied Natural Language Processing, University of California, Berkeley.

Slawski B (2009). How search engines may decide upon and optimize query suggestions. (DOI = www.seobythesea.com/?p=2409)

Search Query Processing (2006). Business white paper, FAST Search Best Practices.

Stanford University (2008). Stemming-and-lemmatization, Stanford University Press.

Natural Language Processing department (2008). Dropping common terms: stop words, Stanford University, USA.

Features of Phpspellcheck (2004). http://www.phpspellcheck.com/Features.

Phpspellcheck (2004). http://www.phpspellcheck.com/Main_Page.

XAMPP-Apache Friends (2015). http://www.apachefriends.org/en/xampp.html.