Improved Search Technique Using Wildcards or Truncation

Shruti Mishra, Sandeep Kumar Satapathy
Department Of Computer Science & Engineering, Institute of Technical Education & Research, Under S'O'A University, Bhubaneswar, Orissa
shruti_m2129@yahoo.co.in, sandeepkumar04@gmail.com

Debahuti Mishra
Department Of Computer Science & Engineering, Institute of Technical Education & Research, Under S'O'A University, Bhubaneswar, Orissa
mishradebahuti@gmail.com

Abstract— Search engine technology plays an important role in web information retrieval. However, with Internet information explosion, traditional searching techniques cannot provide satisfactory result due to problems such as huge number of result Web pages, unintuitive ranking etc. Therefore, the reorganization and post-processing of Web search results have been extensively studied to help user effectively obtain useful information. This paper has basically three parts. First part is the review study on how the keyword is expanded through truncation or wildcards (which is a little known feature but one of the most powerful one) by using various symbols like * or! .The primary goal in designing this is to restrict ourselves by just mentioning the keyword using the truncation or wildcard symbols rather than expanding the keyword into sentential form. Second part consists of the review on subdivision based on wildcards. It is based on the observation that documents are often found to contain terms with high information content which summarize their subject matter. The third part consists of a proposed algorithm based on the above two. The main goal of this paper is to develop a very efficient search technique by which the information retrieval will be very fast, reducing the amount of extra labor needed on expanding the query.

Keywords- Search Engine, Wildcard, Truncation.

I. Introduction

With rapid development of Internet technologies and Web explosion, searching useful information from huge amount of Web pages becomes an extremely difficult task. Currently Internet search engines are the most important tools for Web information acquisition. Based on techniques such as Web page content analysis, linkage analysis, etc., search engines locate a collection of related Web pages with relevance rankings according to user's query. However, current search results usually contain large amount of Web pages, or are with unintuitive rankings, which makes it inconvenient for users to find the information they need. Therefore, techniques for improving the organization and presentation of the search results have recently attracted a lot of research interest. The typical techniques for reorganizing search results include Web page clustering, document summarization, relevant information extraction, search result visualization, etc. Wildcards are one of the searching techniques which are further improved to provide an effective way of searching according to the user's specification.

.

II. KEYWORD SEARCHING

Keyword searching permits you to search a database for the occurrence of specific words or terms, regardless of where they may appear in the database record. For example, even if the word appears in the middle of the title of an article, or anywhere in the abstract, you can still search for it. Keyword searching was made possible by computers; essentially, the computer looks for any group of characters that has a space on either side of it, considers it a "word," and indexes it. The computer takes this task very literally. Even typos, ("philosophy"), incorrect spellings ("archaeology"), or words that were accidentally typed together without a space between them ("for example"), will be found by the computer and indexed, exactly the way they appear.

A. *Advantages of keyword searching*

There are many advantages to keyword searching [5, 6 and 7]: You can locate a very specific reference, even if it is only mentioned a single time. You can use the most current terminology, jargon, or "buzzwords" being used in a discipline, even when no official subject headings exist yet for the concept. You can combine keywords in various ways to create a very detailed and specific **search query**; the actual search as you enter it is known as a **search statement**. As you begin to search for information on your topic, develop a list of keywords and phrases that represent the most important aspects of your topic. Background information located in books and reference sources can be useful sources for these keywords. Try to come up with at least three words to describe each concept, grouping the keywords by concept. You can then use these keywords to "ask" the computer to search for the specific words and phrases on your list.

For example, if you were researching **the effect of the media on body image and eating disorders**, your keyword lists might look like this:

TABLE I

Concept #1	Concept #2	Concept #3
Media* Mass me $ ia Television$ TV Advertis! Film Movies	Body image Self-esteem	Eating disorders Anorexia Bulimia

III. TRUNCATION

Truncation [5, 6 and 7] allows you to search for alternate forms of words. Shorten the word to its root, then add a special character (*, $, !). When truncating, be sure to include enough of the search word to make it meaningful. For Example, If you wanted to search for alternative forms of The word Advertising, a good choice would be to truncate it as "**advertis**." This will search words such as advertis**e**, advertis**ing**, and advertis**ement**. You wouldn't, however, want to truncate after the **adv**. If you did, your search would include words such as adv**antage**, adv**ance**, adv**enture**, adv**ice**, etc.

Different indexes and databases [9] use different symbols after the root word to accomplish truncation. If you are unsure of the truncation symbol for the database you are using, consult the help section for that resource. The most common truncation symbols are: *, $, !.

TABLE II

Symbol	Database	Example
*	• CONSORT/OhioLINK • Web of Science • Yahoo	advertis*
$	• Periodical Abstracts • Humanities Abstracts • Biological Abstracts • MLA Bibliography	advertis$
!		advertis!

III. WILDCARD SEARCH ALGORITHM

A. *Trie Tree*

A trie tree [8] (also known as a "radix tree") [4-5] is a type of search tree; that is, it maps values of a "key" type to values of another type, and it has a tree structure which is usually used as dictionary storage and prefix search. Every node of the tree represents a symbol of the alphabet (e.g. A, B, C... Z), and has 26 child nodes. Let's construct the trie from the following five strings: "BIG, BIGGER, BILL, GOOD and GOSH", which can be seen in Figure1.

When we look for the string “GOOD”, we start at the root and follow the "G" node, followed by the "O" node, another "O" node and finally the "D" node. If we want to look for the string "BAD", we start from the root, follow the"B" node and find out that there is no "A" node after all. Thus "BAD" is not in the text. Because the dictionary always has lots of words with the same prefixes, the trie structure will reduce the data storage for those words by reusing the prefix characters (nodes) in the tree. In addition, it is much efficient for wildcard search based on the prefix term, such as "bi*" and "go*h", but not as good as when the wildcard search term is like "*bi" and "*go*", which we call *no-head wildcard*.

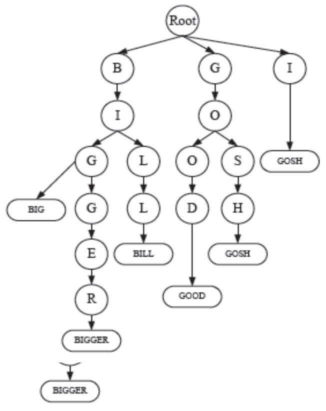

Figure 1. Trie Tree

V. RELATED WORK

A. Interactive Query Refinement

Providing lists of terms from which a user can select for query expansion is not a new idea. Harman [3] generated three different lists from which the user could choose additional terms to add to their query. The results reported from this work were good when users made perfect choices from the lists of available terms. However, others have shown that when presented with a list of potentially relevant terms, users may have difficulties choosing good terms to add to their queries .Others have investigated alternate methods for presenting terms to users for query refinement purposes. For example, Joho et al. [10, 4] generated a hierarchy of query expansion terms from the set of retrieved documents, and presented these to the user via Cascading menus. Although there is value in deducing and representing the relationships among terms within the search results set, their process requires access to the contents of the entire documents within the search results, which is not feasible for interactive Web search systems.

In our work, we can construct a table containing the potential query refinement terms are selected from the top search results returned by the underlying Web search engine. However, rather than collecting the actual document contents, the frequency statistics are based only on the title and snippet provided by the underlying search engine. The title is often descriptive of the information within the document, and the snippet contains contextual information regarding the use of the query terms within the document. These both provide valuable information about the documents in the search results.

B.Calculating frequency of terms using TF-IDF

Essentially, TF-IDF [1,2] works by determining the relative frequency of words in a specific document compared to the inverse proportion of that word over the entire document corpus. Intuitively, this calculation determines how relevant a given word is in a particular document. Words that are common in a single or a small group of documents tend to have higher TF-IDF numbers than common words such as articles and prepositions. The formal procedure for implementing TF-IDF has some minor differences over all its applications, but the overall approach works as follows. Given a document collection D, a word w, and an individual document d ϵ D, we calculate:

$$w_d = f_{w,d} * \log(|D| / f_{w,d}) \qquad (1)$$

Where f_w, d equals the number of times w appears in d, $|D|$ is the size of the corpus, and f_w, D equals the number of documents in which w appears in D (Salton & Buckley, 1988, Berger, et al, 2000). There are a few different situation that can occur here for each word, depending on the values of f_w, d, $|D|$, and f_w, D, the most prominent of which we will examine. Assume that $|D| \sim f_w, D$, i.e. the size of the corpus is approximately equal to the frequency of w over D. If $1 < \log(|D|/ f_w, D) < c$ for some very small constant c, then wd will be smaller than f_w, d but still positive. This implies that w is relatively common over the entire corpus but still holds some importance throughout D. For example, this could be the case if TF-IDF would examine the word Jesus.

A. Encoding TF-IDF

The code for TF-IDF [1,3] is elegant in its simplicity. Given a query q composed of a set of words w_i, we calculate w_i, d for each wi for every document d ϵ D. In the simplest way, this can be done by running through the document collection and keeping a running sum of f_w, d and f_w, D. Once done, we can easily calculate w_i, d according to the mathematical framework presented before. Once all w_i , d.s are found, we return a set D^* containing documents d such that we maximize the following equation:

Either the user or the system can arbitrarily determine the size of D^* prior to initiating the query. Also, documents are returned in a decreasing order according to equation (3).This is the traditional method of implementing TF-IDF.

TABLE III

First eight documents with highest *wd* returned by TF-IDF for term = drugs * Colombia.

Return Pos.	Document #	Sum $f_{w,d}$	Sum w_d
1	788	24	28.09
2	426	72	26.73
3	881	56	23.96
4	253	43	19.16
5	1007	37	16.5
6	362	29	15.42
7	520	33	12.34
8	23	58	10.79

This paper reviews the work described above and proposes an algorithm to expand a term using Wildcards or truncation searching techniques (*,$ or !).

VI. STEPS FOR ALGORITHM

- In the first step, the user gives the initial term or a long query using wildcard or truncation symbols(placing it anywhere in the term or query)
- Then in the second step the first 20 results are viewed and scanned thoroughly.
- The third step is to represent the result into a table. The term or the entire query occurring for highest number of times are calculated and are placed accordingly in the table.
- After the table is displayed, it is up to the user to decide which the particular is or the nearest data for which he/she is willing to go for.

A. Step 1

Since this algorithm is applied in the post processing phase so any kind of Information Retrieval tool can be used. This returns a list of documents like Google or Yahoo.

B. Step 2

1.) The first 20 results are taken into consideration.

2.) In this step the search result produced can contain the whole term or a part of the term along with some other relevant terms for which we have used the symbols(*,?).

C. Step 3

1.) Now the weight of each data or term is calculated that has occurred for highest number of times.

2.) A table is formed having the frequency value along with the specific term with the type of data, which has occurred for the highest number of times.

3.) The table can contain highest frequency value first with the lowest term value at last or vice-versa.

D. Step 4

This is the last step of this entire process associated directly with the user. Once the table is displayed now it is up to the user to decide which particular or nearest data he/she is willing to view. A user can view the data by simply clicking on it.

VII. CONCLUSION AND FUTURE WORK

This paper has presented an interactive method for term or query Expansion using wildcards or truncation searching techniques (*,!, $) based on term weighting. The method is found on the fact that documents contain some terms with high information content, which can summarize their subject matter. Those terms can be found out efficiently through this proposed algorithm. This particular algorithm helps us to save much of our time in typing. But each day is passing and new advancements are coming into light. So, my future aspects would be to implement this strategy and make it more efficient to deal with. Also, my target would be to implement this strategy into various fields and industry to see how efficiently it works and also comparing it with other searching techniques so that I can make this as one of the best searching technique ever used till date.

REFERENCES

[1] Interactive Web Information Retrieval Using WordBars , Orland Hoeber and Xue Dong Yang , Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada S4S 0A2.

[2] G. Salton, C. Buckley, Term-weighting approaches in automatic, text retrieval, Inform. Process. Manage. 24 (5) (1988), 513–523.

[3] D. Harman. Towards interactive query expansion. In Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval, 1988.

[4] H. Cui, J.-R.Wen,W.-Y. Ma, Query expansion by mining user logs, IEEE Trans. Knowl. Data Eng. 15 (2003) 829– 839.

[5] Google-Information literacy tutorial- Searching Techniques

[6] Google-DukeLibraries-Searching Techniques.

[7] A New Wildcard Search Method for Digital Dictionary Based on Mobile Platform , Xin Zhou Yunlong Xu Gongming Chen Zhigeng Pan CAD&CG State Key Lab Zhejiang University School of computer Science Soochow University

[8] SAIL-APPROX: An Efficient On-line Algorithm for Approximate Pattern Matching with Wildcards and Length Constraints.

[9] Simple Deterministic Wildcard Matching Peter Clifford a and Raphafel

[10] Clifford b, aDept. of Statistics, University of Oxford, UK ,Dept. of Computer Science, University of Bristol, UK. E.N. Efthimiadis, Query expansion, Annu. Rev. Inform. Syst. Technol. 31 (1996) 121–187.