International Journal of Computer Theory and Engineering, Vol.2, No.6, December, 2010
1793-8201

Re-Search & Re-Classification Algorithm - An Adaptive Algorithm for Search Engines

Vishwas J Raval*, Amit R Thakkar**, Amit P Ganatra***, Yogesh P Kosta****[1]

Abstract— **Searches of the entire World Wide Web using search engines such as Google, Bing, and Ask have become an extremely common way of locating information. Searches are usually keyword based, which has advantages and disadvantages. The challenge for the user is to come up with a set of search terms which is neither too large (making the search too specific and resulting in many false negatives) or too small (making the search too general and resulting in many false positives). The results retrieved by the search engines are in terms of millions of pages which would not all be useful the user. This paper gives a way of classification of the search results so that the search results can be categorized so as to narrow the million numbers.**

Index Terms— **HTML, DHTML, <meta> tag, Search Engine, Frame, Hyperlinks, Form, Transclusion, Browser, Frame-Bust, Frame-Bust Protection, IFrame, HyperUnique**

I. Introduction to Search Engines

Web search engines are the tools to search the contents stored across World Wide Web. The results generated may be pages, images, ppts or any other types of files. The results of search engines are displayed in the form of a list in which the numbers of pages might be in thousands or millions. The usual working of a search engines consists of following:

- They search the Internet -- or select pieces of the Internet -- based on important words.
- They keep an index of the words they find, and where they find them.
- They allow users to look for words or combinations of words found in that index.

Early search engines held an index of a few hundred thousand pages and documents, and received maybe one or two thousand inquiries each day. Today, a top search engine will index hundreds of millions of pages, and respond to tens of millions of queries per day. The famous search engines are Google, Yahoo, Bing, Ask etc.

All of these search engines have their own search and indexing algorithms. In earlier days the search engines indices were created by searching in <meta> tags. **<Meta> tags** allow the owner of a page to specify key words and concepts under which the page will be indexed. This can be helpful, especially in cases in which the words on the page might have double or triple meanings – the <meta> tags can guide the search engine in choosing which of the several possible meanings for these words is correct. There is, however, a danger in over-reliance on <meta> tags, because a careless or unscrupulous page owner might add <meta> tags that fit very popular topics but have nothing to do with the actual contents of the page. To protect against this, spiders will correlate <meta> tags with page content, rejecting the meta tags that don't match the words on the page. Due to these limitations of <meta> tag, search engines started using web crawlers or known as spiders too. Following figures illustrates the working of a web crawler.

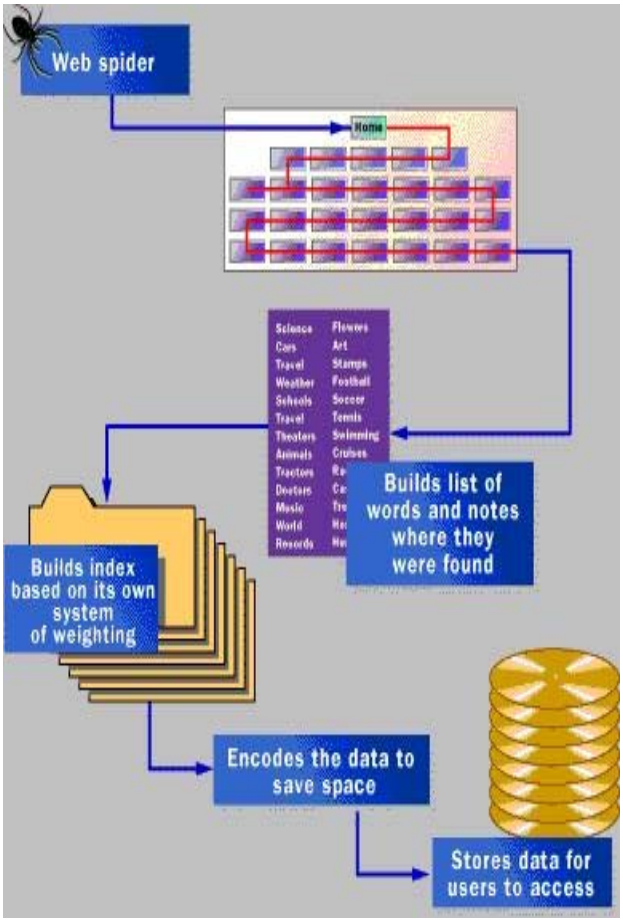

Figure 1. (Source: www.howstuffworks.com)

Google is the most popular search engine among all the available search engines. When the Google spider looked at an HTML page, it took note of two things:

- The words within the page
- Where the words were found

Words occurring in the title, subtitles, **<meta> tags** and

[1] * Asst. Professor, CE-IT, CITC (vishwasraval.it@charusat.ac.in)
** Asso. Professor, CE-IT, CITC (amitthakkar.it@charusat.ac.in)
*** Asso. Professor, CE-IT, CITC (amitganu@yahoo.com)
**** Dean, Faculty of Engineering & Technology (ypkosta@yahoo.com) (IEEE Member) (SCPM, Stanford University)
Charotar University of Science Technology (CHARUSAT), Education Campus, Changa – 388421, Ta – Petlad, Dist – Anand, Gujarat (INDIA)

other positions of relative importance were noted for special consideration during a subsequent user search. The Google spider was built to index every significant word on a page, leaving out the articles "a," "an" and "the." Other spiders take different approaches. [1][6][7][11][12]

These different approaches usually attempt to make the spider operate faster; allow users to search more efficiently, or both. For example, some spiders will keep track of the words in the title, sub-headings and links, along with the 100 most frequently used words on the page and each word in the first 20 lines of text. [1][6][7][11][12]

II. Problems Associated With The Search Engines

A. A huge number of pages as a result

All search engines have different techniques for indexing. Following experiment shows comparison results of three search engines for the same text of "Top universities in world". Following figure 2 shows the text in Google search engine.

Figure 2. Google Search Engine with search text

Figure 3 shows the same text being searched in Ask search engine.

Figure 3. Ask Search Engine with search text

Figure 4 shows the same text being searched in Bing search engine.

Figure 4. Bing Search Engine with search text

Figure 5, 6 & 7 shows the results returned by Google, Ask and Bing search engines respectively.

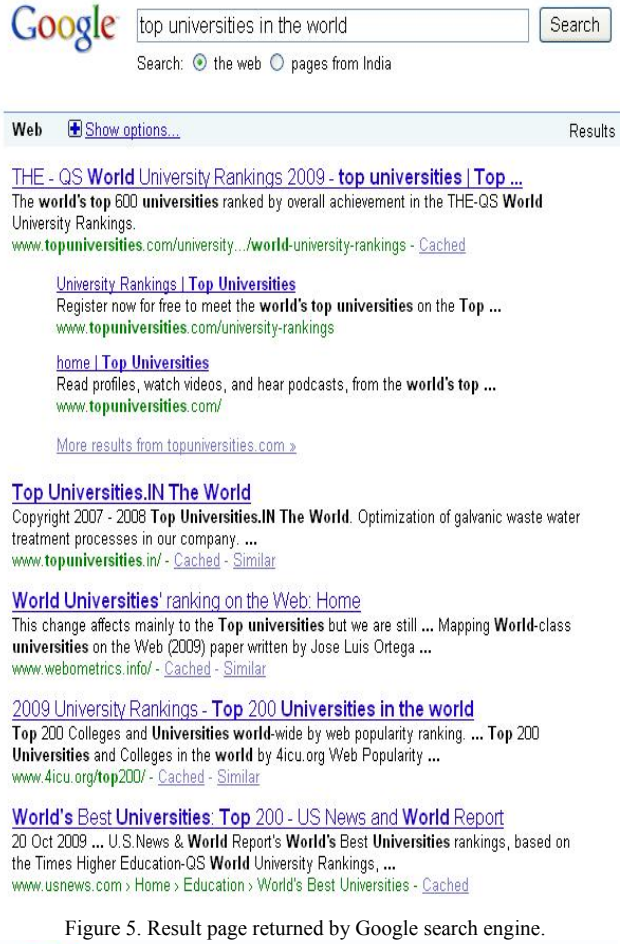

Figure 5. Result page returned by Google search engine.

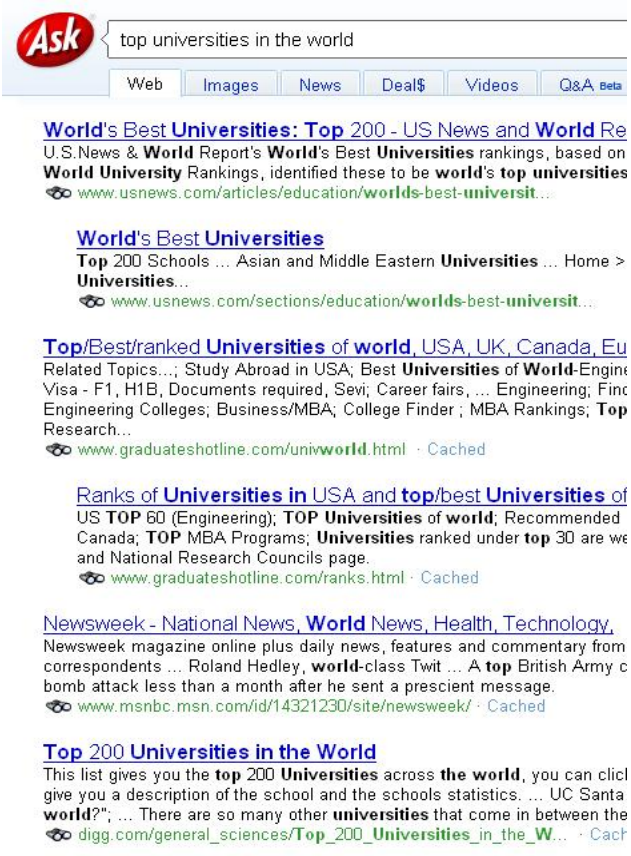

Figure 6. Result page returned by Ask search engine.

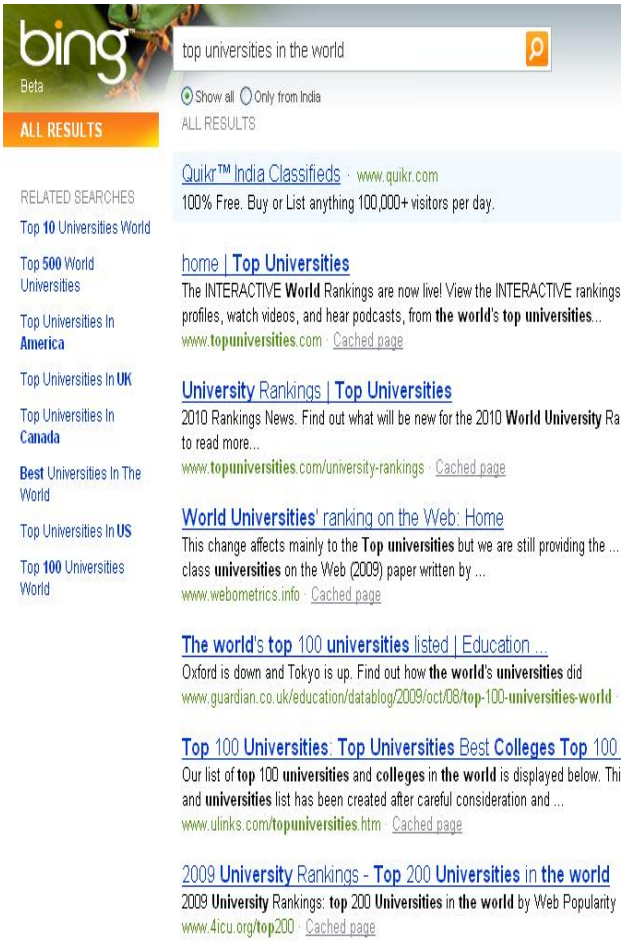

Figure7. Result page returned by Bing search engine.

TABLE 1 SHOWS THE COMPARISON OF THE RESULTS OF ALL THREE SEARCH ENGINES WITH STATISTICS.

Search Engine	Number of Pages returned	Time taken
Google	32500000 (figure 5)	0.14 seconds
Ask	9990000 (figure 6)	-
Bing	22200000 (figure 7)	-

The difference of the result can be noted from the table. The reason behind this is all search engines have different indexing techniques and the results are retrieved from the relevant pages where the index matches with keywords. Due to this all the pages returned, might not be useful to the users. Many a times most of the pages returned in the result seems to be useless. [11][12]

B. *Unclassified Result*

The results returned by search engines are not classified based on the search text. They are classified based on the chronology, images, videos etc. See the figure 8.

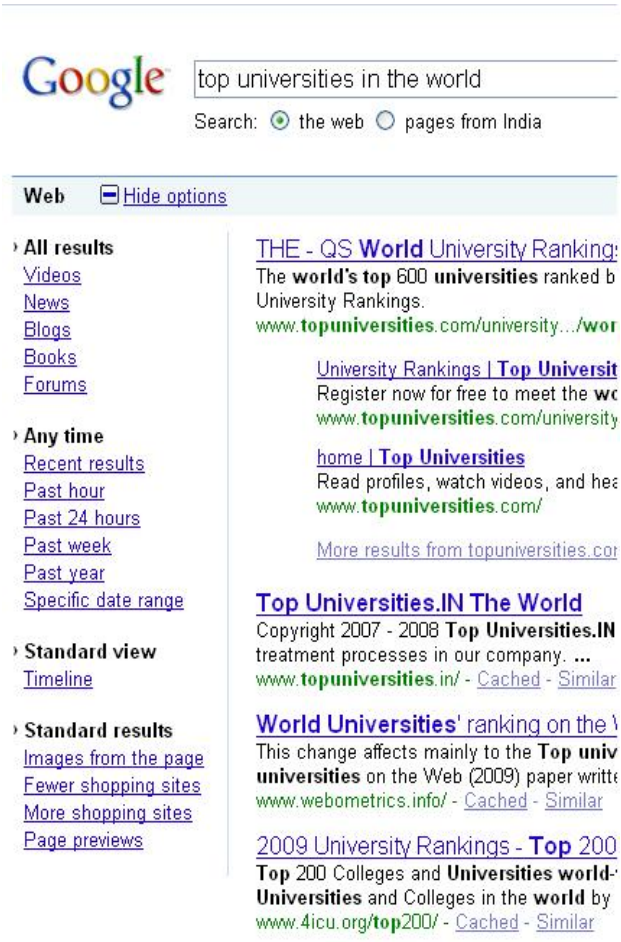

Figure 8. Classification of results by Google.

Due to unclassified pages, end-users are not able to go to the exact page they want to go for which leads to tedious work of looking towards each and every page from the result.

C. *Redundant Pages*

As search engines matches keywords in the indices, they return the all the locations i.e. hyperlinks, where it finds match. However many of these links are redundant and which increase the number of returned pages unreasonably. See figure 9.

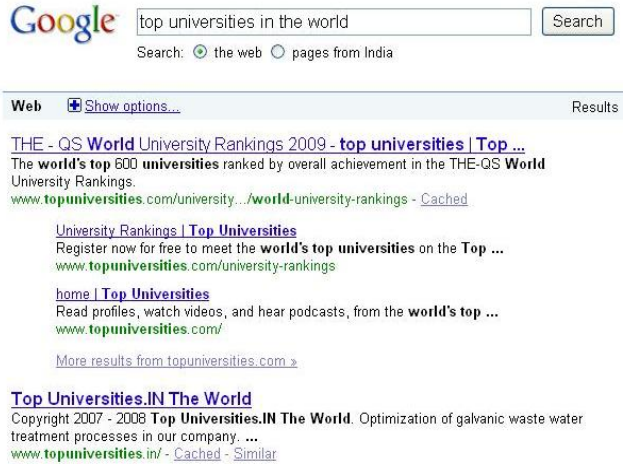

Figure 9. Redundant results by Google.

III. RE-SEARCH & RE-CLASSIFICATION ALGORITHM

Looking towards these unaddressed problems of search

engines we propose an experimental development of a "Re-Search & Re-Classification Algorithm". *Re-Search & Re-Classification Algorithm* is to refine the results returned by the search engines. This can be implemented as an add-on plug-in which would run as a daemon to process the results. This is sufficiently robust and verified for its functioning to solve the problems addressed above. There are key points to be considered.

- This algorithm would work as and when user will go on accessing a new page from the result pages on the fly.
- The algorithm would work on the retrieved results by the search engines.

The following steps are conceptual to the design of the *Re-Search & Re- Algorithm*:

1) Create a web page using frame[2]; here we divide the window into two parts viz: Navigational control and Content frame. Figure-9 shows the left/top of the window to work as *Navigation Frame* and right/bottom part as *Content Frame* - when user starts browsing.

Figure 10. A Framed browser window

2) Scan the web page which is returned by the search engine. [2][3][4][5]
3) Create list of distinct hyperlinks calls *HyperUnique* (here the redundant links will be eliminated).[9][13][14]
4) Scan the web pages given on the HyperUnique and match for the exact sentence or phrase. [8][10][15][16]
5) Create categories equals to the number of phrases matched with the web page contents and add these classes into the Navigation Frame using WebPage Transclusion. [19]
6) Add the corresponding hyperlinks into the respective classes using WebPage Transclusion. [19]
7) If web page contains <form> tag, then the Call command for *Frame-Bust Protective Algorithm* is invoked in order to prevent Frame-Busting. [18]
8) Repeat steps 2 to 7 whenever user accesses new result page or till browser's instance is destroyed.
9) Exit.

Figure 11 shows the Re-Search & Re-Classification Process.

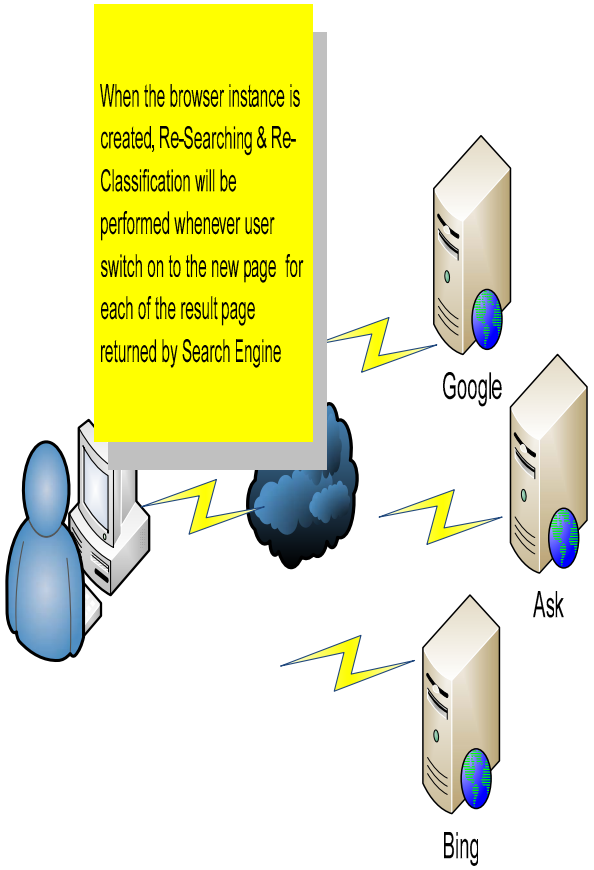

Figure 11. Re-Search & Re-Classification Process

IV. Pros & Cons

A. Pros

1) Redundant links will be eliminated which will reduce the retrieval number.
2) For each page user visits, the algorithm would find closest match so as to eliminating the possibility of visiting useless pages.
3) Due to the classification, pages will be divided into categories as per the phrases and content matched, so user can directly go to the category which is closest to its search text.
1) As the pages would be opened into the content frame,

[2] Framing is the concept of dividing the window of a given webpage into several sections and sub-sections thereafter (in a predetermined manner/sequence), basically for the purpose of ease of accessibility of multiple information efficiently without loss of continuity and sequencing during information search or retrieval. Each section contains a separate frame (smaller than the mother frame) which displays a different HTML document/information. While, the headers and sidebar menus remain static and visible through out the process. The accessing of information or document (by surfing within) especially when the information content frame is larger and scrolling is necessary. [20]

users would never loose navigation.

2) In background it will use result retrieved by any of the search engine only which would already have filtered out the pages.
3) It will save time of user from useless page searching.
4) In case of Frame-Busing, with The Frame Bust Protection Algorithm user would feel transparency during the web browsing. [18]

B. Disadvantages

1) Implementation point of view, it would be difficult to create HyperUnique, classify and add the closest pages into the appropriate class. For this some fast & efficient computing method would be required.
2) If Frame-Bust Protection is used to copy contents for Bust-Protection would result in violation of copyrights for which prior permission would be required [17][18]

ACKNOWLEDGMENT

We are thankful to The Omnipotent GOD for making us able to do something. We express our gratitude to the management of CHARUSAT; Shri Charotar MotiSattavis Kelavani Mandal, for providing us research opportunities and their wholehearted support for such activities. Finally, our acknowledgement can not end without thanking to the authors whose research papers helped us in making this research.

REFERENCES

[1] The Anatomy of a Large-Scale Hypertextual Web Search Engine, Sergey Brin and Lawrence Page
[2] Search Engine Content Analysis (2008), by John D King
[3] Authoritative Sources in a Hyperlinked Environment (1997), by Jon M. Kleinberg (Cornell)
[4] SearchPad: Explicit Capture of Search Context to Support Web Search (2000), by Krishna Bharat (Google)
[5] Improved Algorithms for Topic Distillation in a Hyperlinked Environment (1998), by Krishna Bharat, Monika R. Henzinger (Digital Equipment Corporation)
[6] Ranking Search Results By Reranking the Results Based on Local Inter-Connectivity (2003), by Krishna Bharat (Google)
[7] Dynamic Data Mining: Exploring Large Rule Spaces by Sampling (1999), by Sergey Brin, Lawrence Page (Stanford University)
[8] Finding Near-Replicas of Documents on the Web (1998), by Narayanan Shivakumar, Hector Garcia-Molina (Stanford University)
[9] Collaborative value filtering on the Web (2000), by Andreas Paepcke, Hector Garcia-Molina, and Gerard Rodriquez (Stanford University)
[10] Beyond Document Similarity: Understanding Value-Based Search and Browsing Technologies (2000), by Andreas Paepcke, Hector Garcia-Molina, Gerard Rodriguez, and Junghoo Cho
[11] The Google File System (2003), by Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung (Google)
[12] Searching the Web (2001), by Arvind Arasu, Junghoo Cho, Hector Garcia-Molina, Andreas Paepcke, and Sriram Raghavan (Stanford University)
[13] Efficient Crawling Through URL Ordering (1998), by Junghoo Cho, Hector Garcia-Molina, Lawrence Page (Stanford University)
[14] Finding Replicated Web Collections (2000), by Junghoo Cho, N. Shivakumar, and Hector Garcia-Molina (Stanford University)
[15] Evaluating Strategies for Similarity Search on the Web (2002), by Taher Haveliwala, Aristides Gionis, Dan Klein, and Piotr Indyk (Stanford University)
[16] Similarity Search on the Web: Evaluation and Scalability Considerations (2001), by Taher Haveliwala, Aristides Gionis, Dan Klein, and Piotr Indyk (Stanford Unversity)
[17] Nelson, T. H. (1998). "Transcopyright: Pre-Permission for Virtual Republishing".
[18] Raval, Vishwas J (et al.) ; "Frame-Bust Problem & Bust-Protective Algorithm". Internation Conference on Software Technology & Engineering-2009. World Scientific (ISBN: 978-981-4289-97-9(pbk) July-2009 Pg: 8-11)
[19] Raval, Vishwas J (et al); "WebPage Transclusion – An Adaptive Algorithm for Web Technologies". International Journal of Computer Theory & Engineering (ISSN: 1793-821X (Online Version); 1793-8201 (Print Version)
[20] Special Edition Using HTML 4.0, MacMillan Computer Publishing