International Journal of Engineering Applied Sciences and Technology, 2016
Vol. 1, Issue 6, ISSN No. 2455-2143, Pages 141-144
Published Online April - May 2016 in IJEAST (http://www.ijeast.com)

NAP: IMPROVING THE QUALITY OF SEARCH BY DEDUPLICATING THE SEARCH RESULTS

Naresh Sharma
Asst. Professor,
Department of CSE,
SRM University, Ghaziabad

Purvi Garg
Department of CSE,
SRM University, Ghaziabad

Amit Mishra
Department of CSE,
SRM University, Ghaziabad

***Abstract*— Search Engine Optimization is a very fast growing and vast area of research and can be achieved by focusing on different aspects of a search engine like Page rank, Duplicacy, Redundancy etc. In this paper, we have proposed a prototype of a search engine called NAP. NAP will optimize and provide high quality search results by removing the redundant or duplicate results from the Search Engine Results Page (SERP). By duplicate results we mean different links or URLs containing the same content. This de-duplication of search results is done by applying a filter between the SERP and the ranked pages and as a result only the unique pages will be displayed to the searcher.**

Keywords— **Redundancy, Duplicacy, SERP, Page Rank, Indexing, Crawler, Searcher, Deduplicator**

I. INTRODUCTION

Accessing the web in search of any information is a common practice. In fact, "Google" itself has become a verb in this digital era. The web is composed of interconnected pages which consists of information which can be accessed by means of search engines. As stated in [9], the current size of the web is 49 billion web pages which is growing at an exponential rate. A large percentage of these web pages are exact or near duplicates. The searcher faces a lot of difficulties in dealing with this duplicate data. Search engine giants like Google, Yahoo and Bing are carrying out a great deal of research to develop an efficient mechanism to manage this duplicate data on the web in order to enhance the quality of search. Many a times it is found that the SERP contains redundancy, which simply means same content on different links which annoys the searcher and distracts him/her from search. The duplicate contents degrade the quality of search and provide a poor search experience.

The main objective of this paper is to provide an efficient mechanism to handle these duplicates and provide clean and good quality search results with only unique results. To attain the goal, the architecture for a better search engine is proposed which has a specific module to deal with these duplicates and eliminate them from the SERP. Section II gives a brief about the latest research done in the field of search results deduplication. Section III explains the architecture of NAP, the proposed search engine. Section IV gives the implementation details. Section V shows the experimentations carried out to test the system and the results obtained. Section VI shows the conclusion and finally Section VII for further readings and references.

II. RELATED WORK

Although the search engine technology is a pretty mature one, there is still much scope for improvements. The amount of data on the web is growing and so is the amount of duplicates. Nearly 40% data on the web is nearly similar to each other and hence it becomes a matter of concern to eliminate those duplicates from the search results on the SERP. Much of the research has been done in this field in the last decade. An apparatus and method for producing non-redundant search results was discovered by Sattler and Gaffga in [1]. Thwel and Thein presented an improved indexing mechanism for data deduplication using the properties of B+ tress in [2]. Zhang, Bhagwat, Litwin, Long and Schwarz presented parallel binning for improved deduplication in [3]. Another method for duplicate document detection during web crawling was developed by Dulitz, Verstak, Ghemawat and Dean in [4]. Mettrop, Nieuwenhuysena and Smuldersb presented in INSCIT 2006 How Google web search copes with very similar documents [5]. In [6] Pabitha showed a method to optimize the search results by elimination of duplicate URLs. Jain, Dahlin and Tewari presented the use of bloom filters for refining the web search results in [7]. SpotSigs [8], an algorithm for detection of duplicates in large web collections was presented by Theobald, Siddhart and Paepcke.

III. SYSTEM ARCHITECTURE

This section will describe the components of NAP and gives the details about its working. All the modules are explained in detailed separately for better understanding. Fig 1 shows the overall proposed architecture.

International Journal of Engineering Applied Sciences and Technology, 2016
Vol. 1, Issue 6, ISSN No. 2455-2143, Pages 141-144
Published Online April - May 2016 in IJEAST (http://www.ijeast.com)

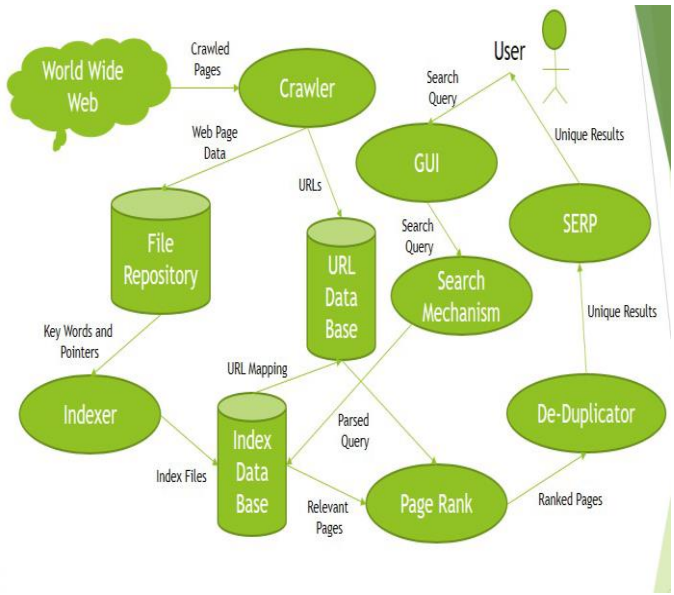

Fig 1: Architecture of NAP

A. Crawler-

A crawler is a program which visits a webpage in order to fetch its data and metadata for the search engine indexer. NAP uses an open source API Apache crawler4j [10] which have b configured accordingly to fetch all the details from the web and store them in the repository and various databases. Crawler4j has the record of crawling the whole Wikipedia in 10 hours. Crawler4J also follows Robot.txt rule while crawling the webpages. Multithreaded crawling is provided by Crawler4j which enables us to run multiple instances of the crawler for parallel processing. The data fetched by the crawler contains the docId (a unique number for each webpage used as an identifier), URL, title, anchor, headings and sub-headings, text contents, URL's on the page (hyperlinks present on the web page) and images & multimedia content. Since NAP only deals with the textual content, we are not concerned about the multimedia part for now.

B. File Repository -

This is the database in which all the information (data/content) of the crawled webpages is stored. NAP stores the data in the form of .txt files which contains the text content of the crawled webpages and are stored in the repository as the with the file name same as the docId of the web page. These text files are stored in the local file system in our case. The entire system of NAP is currently working on a local system and hence the local file system acts as the file repository. For large scale application, a separate file server is to be used as the repository.

C. URL Database -

This is a relational database. It stores the metadata of crawled web pages. This database will store docId, URL, domain, path, subdomain, parent URL, anchor, and title of the crawled pages. Database which we are using here is MySQL database on WAMP server [13]. WAMP stands for Windows Apache MySQL PHP. It is an application software which provides the services of web server and database server.

D. Indexer -

Indexer collects, parses and stores the data to facilitate fast and accurate information retrieval. NAP uses an open source API Apache Lucene 3.5 [11] for the indexing purpose. It indexes file and stores in the form of index and provides an optimal database for faster access. Lucene use inverted index which is a data structure which maps the keyword to its location. It parses the search query entered by the searcher and then carries out the search process on the indexed database.

E. Indexed database-

This is a database which stores indexed files formed by the Lucene. NAP uses the local file system of the host computer as an index database. For large scale applications, a separate file server can be used as the index database. This index structure is highly optimized and compressed so as to provide memory efficiency.

F. Search mechanism-

The search mechanism is NAP is two-step process. First step is to parse the search query and optimize it so as to find the actual keywords in the search query which can be searched on the index database for efficient search. The second step is to search that keyword on the database. These two steps are carried out by separate modules which are explained below.

a. **Query parser:** It is a program that extracts the main keyword from the query and makes it easy for the search algorithm to resolve the query. It basically help the search mechanism to understand what is to be searched. The query parsing in NAP is done by Lucene. Lucene has a separate class called "QueryParser.java" for parsing the search query.

b. **Search Algorithm:**
Since Lucene uses inverted index and the data structure used by Lucene is a hash table hence, hashing mechanism is used for searching the keyword in indexed database. Searching in Lucene is termed as "scoring". The Lucene "Searcher.java" class scores the webpages on the basis of weight which means frequency of keyword in the indexed database and gives the list of docIds of files having that key word. This list is used to fetch the respective URL and other details from the URL database and show it to the user.

International Journal of Engineering Applied Sciences and Technology, 2016
Vol. 1, Issue 6, ISSN No. 2455-2143, Pages 141-144
Published Online April - May 2016 in IJEAST (http://www.ijeast.com)

G. Page Rank-

Page Rank is one of the most important factors for quality ranking of web pages. Different search engines have different measures for page rank calculations. NAP uses the frequency of keywords to calculate the Page Rank of the pages received as a result of the search mechanism. The pages are arranged in the order of Page Rank from highest to lowest in order to give the best results first. The rank calculation in NAP is also done by Lucene.

H. Deduplicator-

The Deduplicator is the most important component of NAP. NAP mainly focuses on the elimination of duplicate links i.e. different links with similar content. To achieve this goal we have applied a filter between the ranked pages and SERP. This filter is named as "Deduplicator". The Deduplicator does string to string comparison of content of web pages to check its uniqueness. This comparison is done by matching the bit stream of the text files which are stored in repository. If two files are found to have 90% of the data similar, they are considered as duplicates. To check the originality of these duplicate pages, we have made an assumption that the webpages crawled earlier is the original one and hence the time stamp of the text file from the file repository is checked to find the original page and discard the duplicates from the search result. The process of comparison and timestamp matching is done by using an open source apache API CommonsIO 2.4 [12].

I. SERP :-

SERP stands for Search Engine Result Page. This is the GUI on which the final unique results are displayed to the user. It is a simple HTML page used just to show the results.

IV. IMPLEMENTATION

NAP is implemented in Java SE 1.7 using Eclipse Kepler IDE. All the APIs stated above have been configured and integrated together to work as a complete system. Separate modules have been developed for the Crawler, Indexer and the Search Mechanism. The Deduplicator has been embedded with the Search Mechanism so as to filter the search results just before they are shown to the user.

V. EXPERIMENTATION AND RESULTS

In order to test the system, a virtual web is created on a local system by downloading the full web structure of around 50,000 websites and hosting them in the local system using WAMP Server. The tool used for downloading the web structure is Win HTTrack [14], which is an open source web crawler and offline browser. Several duplicates were added to the data set so as to make the total count of websites to 80,000. The system used for testing was a Windows 10 System with Core i5 processor and 4 GB of RAM. The tests for all the modules were carried out separately on this data set. Each of those tests are discussed below.

A. Testing the Crawler

To test the crawler, the virtual web described above was crawled using five instances of the crawler. The index page of some of the hosted websites was added as the seed URLs and the crawling process was started. The crawler took nearly 13 hours to crawl 80,000 web pages. The size of the text files created during the crawling process was 204 MBs. These text files were stored in the File Repositories, which in our case was a directory in the local file system of windows.

B. Testing the Indexer

To test the indexer, the path to the file repository was given as input to the indexing module in eclipse. It took just 1.6 minutes to index the repository containing 80,000 text files. The index data base created by the indexer was also stored in another directory of the local file system. The size of the index of 80,000 pages was found to be 58.25 MBs.

C. Testing the Searcher and Deduplicator

To test the Searcher and Deduplicator, a web based GUI was developed using HTML and PHP and the "NAP_Searcher.jar" file which is runnable jar of the searcher module was embedded in the web application and was hosted on WAMP Server on the same system. Keeping in mind the configuration of the host system, the maximum pages to be shown as result was set to ten and the results was recorded in Table 1 which shows the query, total number of pages found, number of unique pages found and the time take to resolve the query.

Query	Total Number of Pages	Number of Unique Pages	Time Taken (Milliseconds)
SRM	10	9	26
Engineer	6	5	31
BHU	3	2	27
Delhi	10	6	25
Computer	10	8	25
Placement	10	9	25

Table 1: Search results for certain queries in NAP.

VI. CONCLUSION AND FUTURE WORK

NAP is the prototype for an advanced full featured web search engine which can provide good quality and highly optimized search results by eliminating the redundancy from the SERP. The test results have found to be impressive and it can be assumed that if tested in server environment with high configuration web servers, it show much better results. Providing high quality search results is the prime goal of any search engine and NAP has proven to be fully up to the mark in providing unique results.

No system is perfect and there is always a scope of improvement. The prime goal for near future of NAP is to improve the time complexity and also include image and multimedia search feature.

V. REFERENCE

[1] Sattler, Juergen, and Joachim Gaffga. "Method and apparatus for non-redundant search results." U.S. Patent Application 11/176,122, filed July 6, 2005.

[2] T. T. Thwel and N. L. Thein, "An Efficient Indexing Mechanism for Data Deduplication," 2009 International Conference on the Current Trends in Information Technology (CTIT), Dubai, 2009, pp. 1-5.doi: 10.1109/CTIT.2009.5423123

[3] Z. Zhang, D. Bhagwat, W. Litwin, D. Long and S. J. T. Schwarz, "Improved deduplication through parallel Binning," 2012 IEEE 31st International Performance Computing and Communications Conference (IPCCC), Austin, TX, 2012, pp. 130-141.doi: 10.1109/PCCC.2012.6407746

[4] Dulitz, Daniel, Alexandre A. Verstak, Sanjay Ghemawat, and Jeffrey A. Dean. "Duplicate document detection in a web crawler system." U.S. Patent 7,627,613, issued December 1, 2009.

[5] Mettrop, Wouter, Paul Nieuwenhuysena, and Hanneke Smuldersb. "How Google Web Search copes with very similar documents." (2006).

[6] Pabitha, International Journal of Emerging Research in Management &Technology ISSN: 2278-9359 (Volume-4, Issue-1), Search Engine Optimization by Eliminating Duplicate Links. http://www.ermt.net/docs/papers/Volume_4/1_January2015/V3N12-121.pdf

[7] Jain, Navendu, Michael Dahlin, and Renu Tewari. "Using Bloom Filters to Refine Web Search Results." In WebDB, pp. 25-30. 2005.

[8] Theobald, Martin, Jonathan Siddharth, and Andreas Paepcke. "Spotsigs: robust and efficient near duplicate detection in large web collections." In Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, pp. 563-570. ACM, 2008.

[9] Worldwidewebsize.Com | The Size of the World Wide Web (The Internet).*Worldwidewebsize.com*. N.p., 2016. Web. 17 Apr. 2016. www.worldwidewebsize.com

[10] Ganjisaffar, Y. "Crawler4j–Open Source Web Crawler for Java." (2012). https://github.com/yasserg/crawler4j

[11] Zhou, Deng-Peng, and Kang-Lin Xie. "Lucene search engine." *Jisuanji Gongcheng/ Computer Engineering* 33, no. 18 (2007): 95-96. https://lucene.apache.org/core/

[12] Team, Commons. "Commons IO - Commons IO Overview". *Commons.apache.org*. N.p., 2016. Web. 18 Apr. 2016. https://commons.apache.org/proper/commons-io/

[13] "Wampserver". WampServer. N.p., 2016. Web. 18 Apr. 2016. http://www.wampserver.com/en/

[14] Roche, Xavier. "Httrack website copier-offline browser." *Computer Software]. Retrieved Jan* 30 (2007): 2011. https://www.httrack.com/