Volume 4, Issue 6, June 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Optimized Web Page Retrieval Using Web Log Mining Technique

Ravi Bhushan, Dr. Rajender Nath
Department of Computer Science & Application
Kurukshetra University, Kurukshetra,
Haryana, India

Abstract- Search engines are very helpful in finding the needed information by posing a query. Initially, search engines were using traditional information retrieval techniques, where keyword based similarity function between the query and the documents was used to identify the required documents. This approach of searching provided poor quality of search results. In this paper, a page optimization system is proposed which uses the clustering as well as ranking mechanism to take the advantages of both. The pattern generator module discovers sequential pattern of web pages in each cluster. The Rank Updater module takes the matched documents retrieved by query processor as input. It improves the rank of retrieved pages according to discovered sequential patterns.

Keywords-Clustering; web usage mining; sequential pattern mining; query processor; pattern analysis etc.

I. INTRODUCTION

Many recent researches have proposed ranking methodologies [1] to use link structure of the web, instead of using the content only, to improve the search result quality. Finding the desired information quickly and easily is a problem for end user. In response to a user query, search engines returns a list of URLs. These URLs are ranked according to the relevance of the query. However, these search engines must have a mechanism so that they can find the result pages according to users' previous interests with respect to their queries and then optimize the results correspondingly. To achieve this, web query log maintained by the search engines can play an important role. An approach for resultant page optimization is proposed, which attempts to optimize the search engine's results by improving their page ranks and thus increasing the relevancy of the pages according to users' feedback. By this way, users are found their relevant pages on the top of the result list.

II. RELATED WORK

This section provides a brief description of the concepts which are used in the proposed system and work done under them.

2.1 Web Query Log

Web query logs keep track of information regarding interaction between users and the search engine. They record the queries issued to a search engine and also a lot of additional information such as the user submitting the query, the pages viewed and clicked in the result set, the ranking of each result, the exact time at which a particular action was done, etc. From web query log information, it is possible to find the search sessions, sets of user actions recorded in a period of time.

2.2 Page Ranking

Modern search engines apply some sort of ranking mechanisms to sort the results to be displayed according to users need. Various ranking algorithms have been introduced in the literature like HITS[6], WPR[7], and SALSA[4] etc. In most of the approaches rank score is independent of users query. The pages represented to the user and requirement of user could not be matched. Therefore, the most relevant web pages to users query do not show on the top of the search engine resultant list. Google uses an algorithm named PageRank (PR) that uses the link structure of the web to determine the importance of web pages. A simplified version of PageRank is defined as:

$$PR(u) = (1-d) + d \sum_{v \in B(u)} \frac{PR(v)}{N_v} \quad \text{---} \quad (1)$$

where u represents a web page, B(u) is the set of pages that point to u. *PR(u)* and *PR(v)* are rank scores of page u and v, respectively. N_v denotes the number of outgoing links of page v,and d is called damping factor.

III. PROPOSED PAGE OPTIMIZATION SYSTEM

The main task of search engine is to produce results desired by the user. The process starts from giving the user query and ends with getting the results. Here, the proposed architecture is divided into two main phases- Back end and Front end.

3.1 Back End Architecture

The information contained in web query logs has been used in back end. The major components of back end are- Data preprocessing, similarity analyzer, query database and sequential pattern generator, which are shown in Figure 1. Each of these components is discussed in detail below.

Preprocessing: Preprocessing is performed on web query log to capture information about user access [5]. There are a number of steps performed in preprocessing which includes: data cleaning, session identification, merging logs from several applications and removing requests for robots.

Similarity Analyzer: User browsing behavior including the submitted queries and their corresponding documents accessed are stored in the web logs. This information is continuously analyzed by the similarity analyzer component. There are two types of similarities based on query terms and user past access pattern.

a) Similarity based on query terms

If two queries submitted by the user have similar terms, it means that they required the same information. The similarities between two queries can be calculated by following method:-

$$\mathrm{Sim}_T(X,Y) = \frac{|T(X,Y)|}{|T(X) \cup T(Y)|} \quad \text{------ (2)}$$

Where T*(X)* and T*(Y)* are the terms used in the queries X and Y respectively, T*(X,Y)* is the common terms used in two queries.

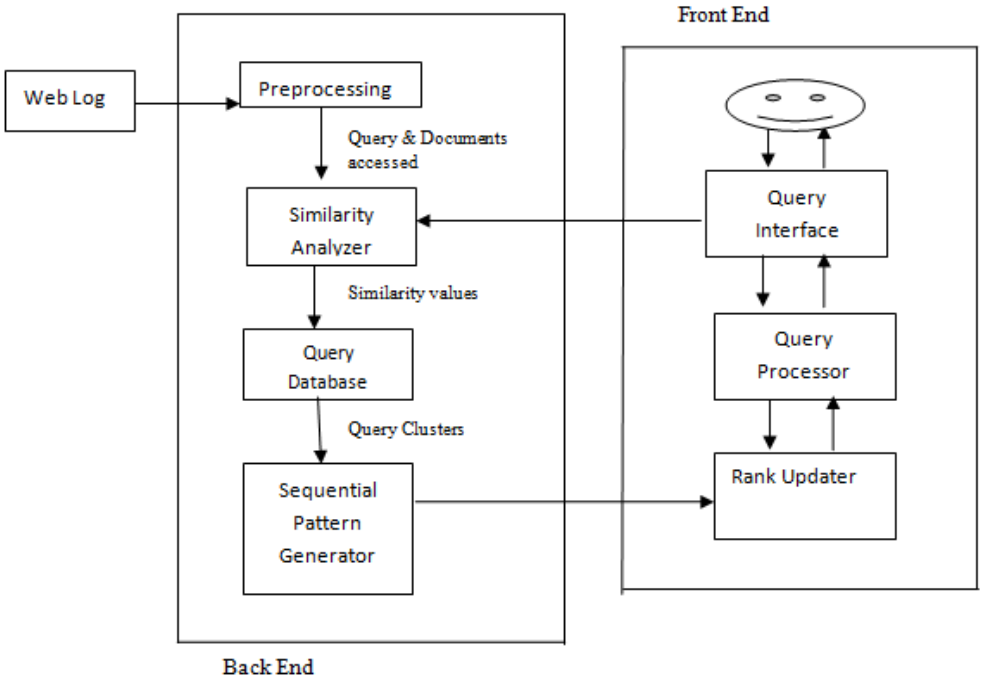

Figure 1: Proposed Page Optimization System

b) Similarity based on User Past access pattern

D. Beeferman and A. Berger [2] proposed that if two queries access the same documents on web, then they can be considered as similar queries. The formula for similarity based on user access pattern is given as:

$$\mathrm{Sim}_{DA}(x,y) = \frac{\sum D(x,d_1) + D(y,d_1)}{\sum D(x,d_2) + D(y,d_2)} \quad \text{------- (3)}$$

Where D(x, d_1) are common documents accessed corresponding to query x ; D(y, d_1) are common documents accessed corresponding to query y; D(x, d_2) are all documents accessed corresponding to query x and D(y, d_2) are all documents accessed corresponding to query y

c) Combining both similarities

The two methods of finding similarities explained above have their own advantages. In first method, queries having similar terms can be grouped together. In second method, queries based on user access pattern can be grouped together.

Both similarities capture the user interest partially when considered separately. Therefore, it is more appropriate to combine both of these similarities as follows:

$\mathrm{Sim}_{total}(x,y) = \alpha.\ \mathrm{Sim}_T(x,y) + \beta.\ \mathrm{Sim}_{DA}(x,y)$ ---- (4)

Where α, β are the constants having values between 0 and 1, α+ β=1. Generally equal importance is given to both types of similarities.

Query Database: Query database holds the output of the similarity analyser component which is the combinations of queries and their corresponding accessed documents. This component finds out the intuitions of users by analyzing their previous history. As user submitted queries on search engine is dynamic in nature, so the information got stored in web

log is also dynamic in nature. Therefore, it is necessary to create the clusters of queries and an algorithm is used for this as shown in Figure 2. The algorithm used here is incremental in nature as information within web log is changing regularly.

Algorithm: Query Clustering

Input: Set of queries and corresponding accessed URLs ; α, β=0.5; similarity threshold ʊ
Output: Set of m query clusters represented by C_m
begin

Step 1: Initialize the number of cluster as 1.
Step 2: Determine the first query from the collection of queries.
Step 3: Set the Cluster_Id as null initially as there is no query clustered.
Step 4: For each query x in query array Set Cluster_Id(x)= C_m
Step 5: Take second query y from collection of queries such that x≠y
Step 6: Find the different types of similarities as

$$Sim_T(X,Y) = \frac{|T(X,Y)|}{|T(X) \cup T(Y)|}$$

$$Sim_{DA}(x,y) = \frac{\sum D(x,d_1) + D(y,d_1)}{\sum D(x,d_2) + D(y,d_2)}$$

$$Sim_{total}(x,y) = \alpha . Sim_T(x,y) + \beta . Sim_{DA}(x,y)$$

Step 7: Now compare the $Sim_{total}(x,y)$ with similarity threshold ʊ
If ($Sim_{total}(x,y) \geq$ ʊ)
Then set Cluster_Id(y)= C_m And add query y into same cluster C_m
Else drop that query as that cannot be grouped into same cluster
Step 8: Repeat same process from step 3 until all queries are grouped to any one of the Clusters.

end

Figure 2: Algorithm for Query clustering

3.2 Front End Architecture

User submits his query on search engine to find relevant information. The query must consist of words or phrases describing the specific information of user interest. The major components of front end phase are: User Interface, Query Processor and Rank Updater.

User Interface: This module provides the way of interaction to the proposed framework. Every search engine uses Graphical User Interface (GUI) to search the information. The user initiates this process by submitting his query on this interface. When a user gives a query to the search engine, then this query is further forwarded to the query processor.

Query processor: Query Processor tokenizes and parses the query string. Tokenizing is the process of breaking the query into understandable stream. Since users may use special operators in their query like Boolean, adjacency, or proximity operators, the system needs to parse the query into query terms and operators. Stop words like a, an, the, of, etc are deleted from the query as they have very less importance in the query terms.

Rank Updater: Query Processor give matched documents according to user query to the Rank Updater Component. The main task of this component is to update the rank score of the pages based on detected sequential patterns. This component operates at the query time. Here those documents are considered for rank update, which are most frequently accessed by users and appear in any one of the sequential patterns.

Value of Page Calculation

For every page x in the sequence pattern, calculate its value which is based on order in which that has been accessed and important for the user. This is calculated as:

$$val(x) = \frac{\ln(total\_depth)}{level(x)} \quad \text{-------} \quad (5)$$

Where total_ depth is the effective depth of the sequential pattern sequence in which page x lies level(x) is the level of page x in the sequence.

Rank Updation

Rank updation of page is done by adding value of page into its existing rank. The new improved rank of page x is calculated as:

$$Improved\_Rank(x) = Rank(x) + val(x) \quad \text{-----} \quad (6)$$

Where rank(x) is the existing rank of page and val(x) is value of page in sequential pattern which represents the popularity of page x.

Performance Evaluation

It may be observed that the pages which are frequently accessed by users have a change in their rank values. Some of the pages have same rank as before. It can be evaluated from the results that the ranking of many web pages have been modified. Thus, more relevant Web pages occupy the top positions in the result list as shown below in Figure 3.

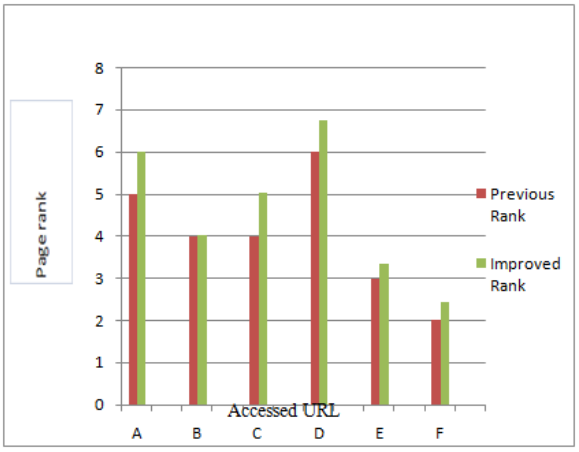

Figure 3: Previous and Improved Rank

IV. SUMMARY

Every search engine produces the result pages in some order of search. When the concept of user previous access pattern is taken in account, then it may be known in advance that in what result pages, the user are interested in. On the basis of sequential patterns generated, the value of a particular document can be calculated. The calculated value of document is added to the previous rank of the page to get a new improved rank of the page. So, the proposed approach uses the concept of re-ranking the retrieved results. This process of re-ranking adjusts the most important documents on the top of the list and reduces the seek time for users to access those documents.

REFERENCES

[1] S. Chakrabarti, B. Dom, D. Gibson, J. Kleinberg, S. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, "Mining the link structure of the world wide web", IEEE Computer, Pages 60–67, 1999.

[2] Doug Beeferman and Adam Berger, 2000. Agglomerative clustering of a search engine query log. In Proceedings of the 6th ACMSIGKDD International Conference on Knowledge Discovery and Data Mining, (August). Acm Press, New York, NY, 407–416.

[3] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke, and S. Raghavan, "Searching the Web," ACM Transactions on Internet Technology, Vol. 1, No. 1, pp. 97-101, 2001.

[4] R. Lempel and S. Moran, "SALSA: The Stochastic Approach for Link-Structure Analysis". ACM Transactions on Information Systems, 19(2), Apr 2001, pp: 131–160.

[5] Bhushan Ravi; Nath, R., "Recommendation of optimized web pages to users using Web Log mining techniques," *IACC, 2013 IEEE 3rd International* , vol., no., pp.1030-1033, 22-23 Feb. 2013 ISBN: 978-1-4673-4527-9.

[6] S. Chakrabarti, B. E. Dom, S. R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. Kleinberg. Mining the Web's link structure, 32(8):60-67, 1999.

[7] Wenpu Xing and Ali Ghorbani, "Weighted PageRank Algorithm", Proc. of the 2nd Annual Conference on communication Networks & Services Research, 2004

