IMPROVED WEB SEARCH RESULT RANK OPTIMIZATION USING SEARCH ENGINE QUERY LOG

Kajal Y.VYAS

Asst. Professor, Department of Computer Engineering, G. K. Bharad Institute Of Engineering, Rajkot, Gujarat

kajalvyas55@gmail.com

ABSTRACT: *The World-Wide Web provides access to plenty of information to the users of Internet, but difficulty increases in identifying the relevant piece of information. Users are not interested in retrieving large number of documents, but high usefulness of retrieved documents is desired. Search Engine returns list of documents arranged in sequence according to rank of document. Rank optimization is done to move useful documents upwards in search result list. Search engine stores log for keeping track of user activities including user queries. Here mining of search engine query logs are used to optimize ranks. The purpose of this work to improve performance of rank optimization architecture, so that time of optimization decreases and it returns accurate result. First cluster of similar queries are made, and for each cluster frequently accessed documents are found by using sequential pattern mining. Weights for those documents are count and that is added to original ranks. Documents are arranged according to new ranks. So the useful documents are found at top of the list.*

Keywords—Query logs, Web mining, Search engine, Rank of document, Clustering, Sequential pattern mining

I: INTRODUCTION

Discovery of useful information from the World Wide Web becomes a practical necessity with the explosive growth of data available on the World Wide Web. Search engine is a competent tool of finding information from Internet in modern times. On the interface of search engine, queries are articulated through a combination of keywords. 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. Nowadays various ranking algorithms available to search and display list of documents for given query. Although search engines use advance technologies; there are still many situations, where user get undesired and non-relevant pages in the top results of the ranked list. Because the ambiguity and the short lengths of query terms that do not describe the user's personal requirements enough, search engine will return a lot of web pages which contain many ineffectual pages. Nowadays, return a set of web pages based on user query words is not a big problem in search engines; rather the problem appears at the user side as he has to go through the long result list to find his preferred content.

Web mining can be defined as the application of Data Mining techniques to the web related data. Search engine logs open interesting opportunities for web mining. Existing work on mining such data has mostly attempted to discover knowledge at the level of queries and accessed documents in response to the query. The logs provide an excellent opportunity for gaining insight into how a search engine is used and what the users' interests are.[1-3] Web access pattern, which is the sequence of accesses pursued by users frequently, is a kind of interesting and useful knowledge in practice. It is used to predict users' navigational behavior. In this paper, the purpose of web log mining is to improve performance of the search engine by utilizing the mined knowledge.

II: SEARCH ENGINE RESULT OPTIMIZATION USING LOGS

Most of current ranking algorithms[4] work on keyword matching or link analysis. Long lists of ranked pages are returned by Search engine, from which user have to find out desired information. User can not examine every returned document, so there was need to look for some means of result optimization. So architecture given in was developed. But performance of that architecture is improved over here. Some search engine uses PageRank algorithm.

The given system[2, 5] based on learning from query logs predicts user information needs and modify

ranks, so user desired relevant pages occupy their places earlier in the result list. So it reduces the seek time of the user within the search result list.

The architecture given in consists of following components:

1. Similarity Analyser

It finds out similar queries based on keyword matching and clicked URL matching. If queries contain similar words or two queries lead to the selection of same or similar documents, they are considered as similar queries. Both criteria were combined in [2, 5].

2. Cluster queries

This module generates different clusters of queries based on similarity values. Initially, no queries are assigned to any cluster. Each query is examined against all other queries. If the similarity value turns out to be above the pre-specified threshold value (τ), then the queries are grouped into the same cluster.[2, 5-8] Until all queries get classified to any one of the clusters, the same process is repeated.

3. Pattern Generator

From each cluster, to find out sequential patterns this module can be used. It takes clusters as input, apply mining algorithm and give patterns as output. This sequences shows which web pages were frequently accessed by users in which sequence. GSP algorithm is used in [2, 5].

4. Rank Updater

This module works online at query time and returns updated ranks of documents.

-The sequential patterns of the concerned cluster are retrieved from the local repository maintained by the Sequential Pattern Generator.

-The weights are calculated for every page X present in the sequential patterns.

- Final rank of a page is updated if it happens to be present in the patterns of cluster C. The improved rank is calculated as the summation of previous rank and its weight value.

III: PROBLEM FORMULATION

For finding similarities between queries, the performance can be improved by using other equations of finding similarities.

GSP is also based on same concept as Apriori[9] and AprioriAll algorithms. Apriori is widely known algorithm for pattern mining. AprioriAll is next to Apriori. GSP[10] is faster than both these algorithms. But GSP has some drawbacks. GSP uses candidate generate-and-test approach in which huge set of candidate sequences generated, more memory is required. When min_support drops, the number of frequent sequences grows up exponentially and it costs an exponential growth amount of time to process a pretty small database. So it is inefficient for mining long sequential patterns. WAP-tree mining is a non-Apriori method which stores the web access patterns in a compact prefix tree, called WAP-tree, and avoids generating long lists of candidate sets to scan support for. However, WAP-tree algorithm has the drawback of recursively re-constructing numerous intermediate WAP-trees during mining in order to compute the frequent prefix subsequences of every suffix subsequence in a frequent sequence. This process is very time-consuming.

For calculating weight of any page, the formula given in [5] was,

$$\text{Weight(X)} = \frac{\ln(len_{pattern(X)})}{level(X)}$$

Where $len_{pattern(X)}$ is the effective length/depth of the sequential pattern in which X occurs and level(X) is the depth of X in the pattern.But level of page in the sequence doesn't depends on access frequency, so here weight equation should not depend on level.

IV: IMPROVED WEB SEARCH RESULT RANK OPTIMIZER AS SOLVER

1. Cosine equation for finding similarity of queries using keywords and URLs:

New equation for finding similarities between two queries can be used.[11]

$$Sim_{keyword}(p,q) = \frac{\sum_{i=1}^{k} cw_i(p) \times cw_i(q)}{\sqrt{\sum_{i=1}^{m} w_i^2(p)} \times \sqrt{\sum_{i=1}^{n} w_i^2(q)}} \text{---(A)}$$

Where $cw_i(p)$ and $cw_i(q)$ are the weights of the i^{th} common keyword in the query p, and q respectively and $w_i(p)$ and $w_i(q)$ are the weights of the i^{th} keywords in the query p and q respectively. In this paper, for query weighting tf(term frequency) is used.

For finding similarities using clicked URLs, each clicked URL for a query is split by pre-defined separators, such as slashes (/), dots (.), etc. Then numbers (except for those in website domains) and URL stop words (e.g. www, .com and .index) are removed. Finally, the URL is represented by tokens.

Each query q_i is then represented by a vector of tokens as follows:

$$\vec{q_i} = < w_{i1}, w_{i2}, \ldots, w_{in} > \text{---(B)}$$

Where w_{ij} is the weight of token t_j in q_i. Here weight is calculated by taking term frequency. Then, the similarity between two queries p and q- $Simclicked_{URL}(p,q)$ is determined using cosine similarity as per above equation (A). Here weights of tokens are taken instead of weights of keywords. Both criteria can be combined and advantages of both can be used. Both query keywords and the corresponding document clicks can partially capture the users' interests when considered separately. Therefore, it is better to combine them in a single measure. A simple way to do it is to combine both measures linearly as follows:

$$Sim_{combined}(p,q) = \alpha * Sim_{keyword}(p,q) + \beta * Sim_{click}(p,q)$$

---- (C)

Where α and β are constants where 0<= α and $\beta \leq 1$ and $\alpha + \beta = 1$. In the current implementation, these parameters are taken to be 0.5 each.

In this way, this module helps to find relating queries that happen to be worded differently but stem from the same information need.

2. Generation of Clusters

From previous step for two queries p and q, if $sim_{combined}(p,q) \geq \tau$, Where τ is predefined threshold, then p and q are added to same cluster. All queries are divided into different clusters.

3. Pattern minning

Pre-Order Linked WAP-Tree Mining (PLWAP) algorithm[12, 13] is a new sequential pattern mining algorithm for web logs, which is based on WAP-tree but avoids recursively re-constructing intermediate WAP-trees during mining of the original WAP tree for frequent patterns. PLWAP algorithm is able to quickly determine the suffix trees or forests of any frequent pattern prefix under consideration by comparing the assigned binary position codes of nodes of the tree. Here PLWAP-tree algorithm is used to mine frequent patterns from query log data. It is better than GSP.

4. Rank Updation

Here weight calculation equation is changed to,

$$\text{Weight}(X) = \frac{len_{pattern(X)}}{\ln([len_{pattern(X)}])} \text{ ---(D)}$$

Where $len_{pattern(X)}$ is the total number of pages in the sequential pattern in which X occurs. Weight is found by using length of pattern divided by natural log of length of pattern.

Actual Rank used in a search engine can be improved by adding weight value to actual rank. Formula is as below:

$$New\_Rank = Rank(X) + Weight(X) \text{ ---(E)}$$

V: EXPERIMENTAL DATA

USERID	QUERY	CLICKED URL
1220051	PRICE MARUTI SWIFT	WWW.MARUTISWIFT.COM WWW.GAADI.COM WWW.MARUTIDZIRE.COM
1220051	MARUTI SWIFT DZIRE	WWW.MARUTIDZIRE.COM WWW.CARDEKHO.COM
1220051	MARUTI SWIFT	WWW.MARUTISWIFT.COM WWW.GAADI.COM WWW.CARWALE.COM
1220051	MARUTI SWIFT DZIRE PRICE	WWW.MARUTISWIFT.COM WWW.MARUTIDZIRE.COM WWW.CARWALE.COM WWW.CARDEKHO.COM
1220051	RAY BAN SUNGLASSES	WWW.RAY-BAN.COM WWW.HISUNGLASSES.COM
1220051	RAY BAN SUNGLASSES INDIA	WWW.RAY-BAN.COM WWW.EMPORIUMONET.COM
1220052	MARUTI SWIFT PRICE	WWW.MARUTISWIFT.COM WWW.GAADI.COM WWW.CARWALE.COM
1220052	MARUTI SWIFT	AUTO.INDIACAR.COM
1220052	RAY BAN SUNGLASSES INDIA PRICE	WWW.EYEWEARTOWN.COM WWW.RAY-BAN.COM WWW.APPARELL.SHOP.EBAY.IN
1220053	MARUTI SWIFT PRICE	WWW.MARUTISWIFT.COM WWW.GAADI.COM WWW.CARWALE.COM
1220054	MARUTI SWIFT DSIRE PRICE	WWW.CARDEKHO.COM WWW.MARUTISUZUKI.COM WWW.MARUTITRUEVALUE.COM
1220054	MARUTI SWIFT	WWW.GAADI.COM WWW.CARWALE.COM
1220054	MARUTI SWIFT PRICE	WWW.MARUTISWIFT.COM WWW.MARUTISUZUKI.COM WWW.MARUTITRUEVALUE.COM
1220054	RAY BAN SUNGLASSES	WWW.RAY-BAN.COM WWW.APPARELL.SHOP.EBAY.IN WWW.EMPORIUMONET.COM

q_1= Maruti Swift Price so q_1={Maruti, Swift, Price}

q_2= Maruti Swift Dzire so q_2={Maruti, Swift, Dzire}

$Sim_{keyword}(q_1,q_2) = \frac{1+1}{\sqrt{1+1+1}\sqrt{1+1+1}} = 0.66$

$Sim_{clickedURL}(q_1,q_2) = \frac{1}{\sqrt{1+1+1}\sqrt{1+1}} = 0.41$

Final $Sim(q_1,q_2)$ is 0.53 > τ. So both queries are similar. So both queries are put in same cluster.

Final 2 clusters obtained are:

C1= {q_1, q_2, q_3, q_4, q_7, q_8, q_{10}, q_{11}, q_{12}, q_{13}}

C2= {q_5, q_6, q_9, q_{14}}

Comparisons of equations in this paper and reference papers are given in Fig.1

If URLs are assigned to different variables for easy calculation, let

A=www.marutiswift.com/1=www.marutiswift.com

B=www.gaadi.com/ 2=www.gaadi.com

C=www.marutidzire.com/3=www.marutidzire.com

D= www.cardekho.com/ 4= www.cardekho.com

E= www.carwale.com/5= www.carwale.com, or and so on.

From PLWAP-tree frequent patterns mined are {(A), (AB), (B), (BE), (D), (D)(A), (D)(B), (D)(BE), (D)(B)(A), (D)(E), (D)(E)(A), (D)(BE)(A), (ABE), (B)(A), (BE)(A), (E), (E)(A)}

Or

{(1), (12), (125), (15), (2), (2)(1), (25), (25)(1), (4), (4)(1), (4)(2), (4)(2)(1), (4)(25), (4)(25)(1), (4)(5), (4)(5)(1), (5), (5)(1)}

Frequent Patterns are same as output of GSP algorithm. But runtime of PLWAP-tree is very less compared to GSP. The comparison of runtime of both these algorithms is as given in figure 2. Here runtime is compared for 98KB data. Runtime is compared for minimum supports 0.005, 0.01, 0.02 and 0.05.

Maximum length frequent pattern is (D)(BE)(A) or (4)(25)(1) corresponding to C1. Using equation (D), Weight(E)= Weight (D) =Weight (B) =Weight (A) = 4/ln(4)=2.88 Actual ranks of D, E, B, A were found 5,4,6,4.

New_Rank=Actual Rank + Weight value

New_Rank(D)=5+2.88=7.88

New_Rank(E)=4+2.88=6.88

New_Rank(B)=6+2.88=8.88

New_Rank(A)=4+2.88=6.88

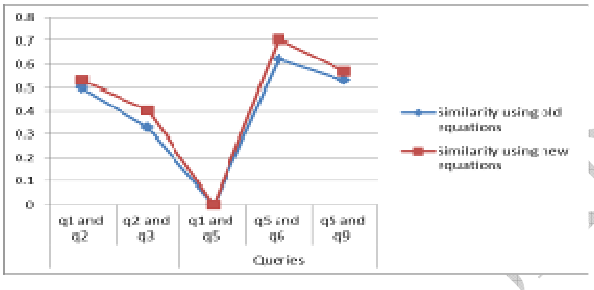

Fig 1. Comparison between old equations and new equations (X-axis=Queries → Y-axis=Similarity)

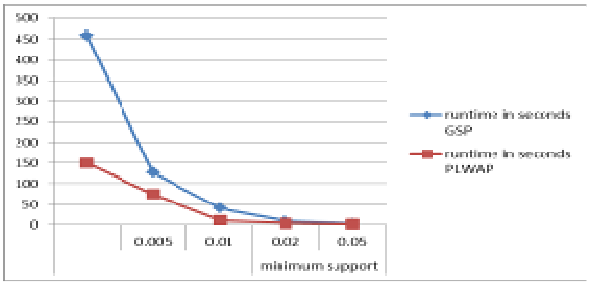

Fig 2. Comparison between runtimes of GSP and PLWAP algorithm

(X-axis=Minimum Support → Y-axis=Runtime in seconds)

VI: CONCLUSION

Search Engine returns results based on various ranking algorithm. In response to user's query, search engine returns very long result list. It is very time consuming process to find related pages from it. Rank optimization technique was suggested based on query log mining, which optimize the results of a search engine by returning the more relevant and user desired pages upward in search result list. It reduces user's efforts and seeking time. Here performance is improved, so it gives better optimization of search engine results in less time.

REFRENCES

[1] Thorsten Joachirns, "Optimizing search engines using clickthrough data," Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 133-142, [2002].

[2] Neelam Duhan and A.K.Sharma, "Rank Optimization and Query Recommendation in Search Engines using Web Log Mining Techniques," JOURNAL OF COMPUTING, vol. 2, December [2010].

[3] Neelam Duhan and A.K. Sharma, "A Novel Approach for Organizing Web Search Results using Ranking and ClusteringInternational Journal of Computer Applications," International Journal of Computer Applications, vol. 5, [2010].

[4] Ashutosh Kumar Singh and Ravi Kumar P, "A Comparative Study of Page Ranking Algorithms for Information Retrieval," International Journal of Electrical and Computer Engineering, vol. 4, [2009].

[5] A.K. Sharma, Neelam Duhan, Neha Aggarwal and Ranjna Gupta, "Web Search Result Optimization by Mining the Search Engine Query Logs," IEEE, [2010].

[6] M. Barouni-Ebrahimi, Ebrahim Bagheri and Ali A. Ghorbani, "A Frequency Mining-based Algorithm for Re-Ranking Web Search Engine Retrieval," Proceedings of the 21th Canadian Conference on Artificial Intelligence, Windsor, Canada, [2008].

[7] Yi Liu, Liangjie Zhang, Ruihua Song, Jian-Yun Nie, and Ji-Rong Wen, "Clustering Queries for Better Document Ranking," CIKM'09, Hong Kong, China, [2009].

[8] J. Wen, J. Mie and H. Zhang, "Clustering user queries of a search engine," In Proc. at 10th International World Wide Web Conference, [2001].

[9] R. Agrawal and R. Srikant, "Mining sequential patterns," In Proc. 1995 Int. Conf. Data Engineering (ICDE'95), pp. 3-14, [1995].

[10] Ramakrishnan Srikant and Rakesh Agrawal, "Mining Sequential Patterns: Generalizations and Performance Improvements," Proc. of 5th International Conference Extending Database Technology (EDBT), France, March [1996].

[11] Kajal Y. Vyas and Kiran R. Amin, "Web Search Result Rank Optimization Using Search Engine Query Log Mining," International Journal of Systems , Algorithms & Applications, IJSAA, vol. 2, pp. 85-89, May [2012].

[12] C.I. Ezeife and Yi Lu, "Mining Web Log Sequential Patterns with Position Coded Pre-Order Linked WAP-Tree," Data Mining and Knowledge Discovery, Springer Science + Business Media, vol. 10, pp. 5-38, [2005].

[13] C.I. Ezeife, Yi Lu and Yi Liu, "PLWAP Sequential Mining: Open Source Code," WOODSTOCK '97 El Paso, Texas USA, [1997].