Global Journal of Computer Science and Technology
Network, Web & Security
Volume 13 Issue 13 Version 1.0 Year 2013
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Organizing user Search Histories

By Ravi Kumar Yandluri
Gokaraju Rangaraju Institute of Engineering & Technology, India

Abstract - Internet userscontinuously make queries over web to obtain required information. They need information about various tasks and sub tasks for which they use search engines. Over a period of time they make plenty of related queries. Search engines save these queries and maintain user's search histories. Users can view their search histories in chronological order. However, the search histories are not organized into related groups. In fact there is no organization made except the chronological order. Recently Hwang et al. studied the problem of organizing historical search information of users into groups dynamically. This automatic grouping of user search histories can help search engines also in various applications such as collaborative search, sessionization, query alterations, result ranking and query suggestions. They proposed various techniques to achieve this. In this paper we implemented those techniques practically using a prototype web application built in Java technologies. The experimental results revealed that the proposed application is useful to organize search histories.

Indexterms : *search engine, search history, click graph, query grouping.*

GJCST-E Classification : *H.3.5*

Strictly as per the compliance and regulations of:

Organizing user Search Histories

Ravi Kumar Yandluri

Abstract - Internet userscontinuously make queries over web to obtain required information. They need information about various tasks and sub tasks for which they use search engines. Over a period of time they make plenty of related queries. Search engines save these queries and maintain user's search histories. Users can view their search histories in chronological order. However, the search histories are not organized into related groups. In fact there is no organization made except the chronological order. Recently Hwang et al. studied the problem of organizing historical search information of users into groups dynamically. This automatic grouping of user search histories can help search engines also in various applications such as collaborative search, sessionization, query alterations, result ranking and query suggestions. They proposed various techniques to achieve this. In this paper we implemented those techniques practically using a prototype web application built in Java technologies. The experimental results revealed that the proposed application is useful to organize search histories.

Indexterms : search engine, search history, click graph, query grouping.

I. Introduction

Information is continuously being added to World Wide Web. As the content is dramatically increased and made available to general public, users online make lot of queries to meet their information needs. There are many search engines that help the users in this regard. From a study of search logs of AltaVista [1] and Yahoo [2] it is evident that only 20% of queries are navigational while the rest are transactional. This is because users make task oriented searches like personal finances, travel arrangements, online purchasing and so on. A common thread in all these things is that users make searches by giving a keyword as input. Search engines respond with required information. Sometimes users may make queries that are hierarchical and dependent in nature. To reuse searches and save lot of time, of late, search engines came up with a feature known as "Search History". They are able to maintain search histories which are associated with user credentials. The authenticated users can view their search histories. However, at present the browsers are showing search histories in chronological order only. They do not organize search histories in more useful fashion. Fig. 1 shows the search history of a user with labels such as Yesterday, Today, and then date wise.

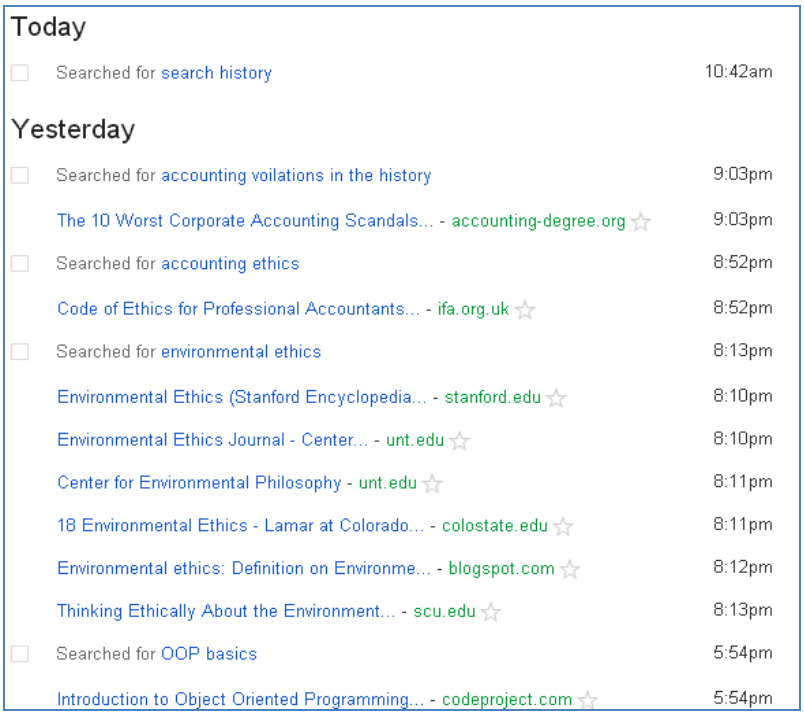

Figure 1 : Search history of a user organized by google

Author : Gokaraju Rangaraju Institute of Engineering & Technology, India. E-mail : yandluri.ravikumar@gmail.com

As can be seen in fig. 1, Google search history is shown in chronological order. Google can also show the search history in terms of various categories such as web, images, news, shopping, Ads, videos, maps, blogs, books, visual search, travel and finance. However, it does not organize the search history based on related similarity of the searches. Query groups help search engines in many applications. The key features of search engine can be improved by making query groups meaningfully. The utilities of query groups include collaborative search, sessionization, query alterations, result ranking and query suggestions. For instance "financial statement" is the query which belongs to a group such as {"financial statement", "bank of America"}. This information will boost the performance of search engines while giving ranks. Task level search in collaborative fashion can be done using query groups. The search query groupwhich is the goal of this paper is presented in fig. 2.

Time	Query	Time	Query
10:51:48	saturn vue	12:59:12	saturn dealers
10:52:24	hybrid saturn vue	13:03:34	saturn hybrid review
10:59:28	snorkeling	16:34:09	bank of america
11:12:04	barbados hotel	17:52:49	caribbean cruise
11:17:23	sprint slider phone	19:22:13	gamestop discount
11:21:02	toys r us wii	19:25:49	used games wii
11:40:27	best buy wii console	19:50:12	tripadvisor barbados
12:32:42	financial statement	20:11:56	expedia
12:22:22	wii gamestop	20:44:01	sprint latest model cell phones

Figure 2 : Search history of a user (excerpt from [3])

As can be seen in fig. 2, the search history of a user is given in chronological order. However, it can be organized more meaningfully by grouping related queries. Fig. 3 shows the results of grouping related search words.

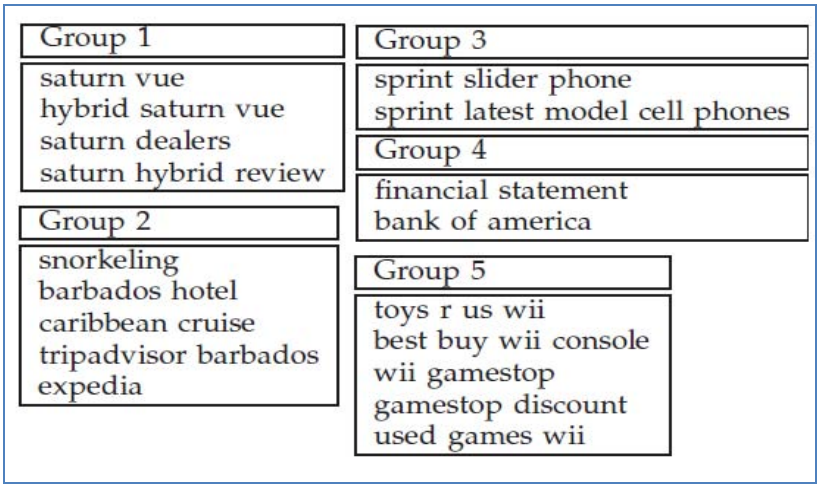

Figure 3 : Query Groups (excerpt from [3])

As can be seen in fig. 3, the search history presented in fig. 2 is grouped into four categories based on the similarity of searches. In group 4 "financial statement" and "bank of America" are grouped together as they are closely related. In the same fashion, all the search strings in group 1 are closely related.

In this paper we implemented the mechanisms proposed by Hwang et al. [3]in which we do not depend on temporal properties or textural properties completely. We depend on the behavioral data present in search engine's logs. First of all we make a query reformulation graph which contains relationships among queries based on the frequency. Then we build a query click graph that reflects relationships based on user clicks.

Then we combine both query reformulation graph and query click graph to generate a query fusion graph. This kind of approach is also followed in [4], [5] for session identification and in [6], [7] for query clustering. However, in this paper our work extends that in two ways. We use information from click graph and also query reformulation graph for capturing similarity in better way. We built a prototype web application to demonstrate the proof of concept.

The remainder of this paper is organized into some sections. Section II presents review of literature. Section III provides the proposed approach for organizing user search histories. Section IV describes prototype implementation details. Section V presents

experimental results while section VI concludes the paper.

II. Prior Works

Organizing user search histories was done earlier with chronological and other orders. There were studies to know whether two queries belong to a single search task. A search task is made up of many queries. Search- task identification was studied in [4] and [5]. In [4] it is explored that search session has a set of tasks and each task is divided into multiple sub-tasks known as goals. The authors used binary classifier which exploited the query logs, time and text to know whether two queries belong to same task. Similar features were employed by [5]. However, Hwang et al. [3] did it differently by considering query pairs additionally. These query pairs will have URLs associated based on their co-occurrence which is presented in a fusion graph. In [4] there is no provision to break the query when it belongs to two groups. Our approach does not need manual labeling. The random walk approach followed by them needs an updated query fusion graph. The aim of their mechanism is to group search queries by identifying tasks at server side. This will help in query suggestions [5] and personalization. Sessionization also focused by some researchers. It is based on the "timeout threshold" which was employed in [8], [9], [10], [11], [12], [13], [14]. However, time is not considered to be a good basis for grouping queries. Overlapping of terms of two queries concept is used in [11] and [15] in order to find out changes in search topics. Various refinement classes were studied in [16] based on based on the queries and the underlying keywords present. They also used Bayesian classifier to predict such classes. Query chains concept was used in [17] by combining textual similarity features with timeout thresholds through a classifier known as Bayesian.

Query clustering is also related to online query grouping in some way. Many researches were made on query clustering [18], [19], [6], [7], [20]. Bipartite graph building concept is used in [6] and [7] for grouping queries. Click graphs were built in [18] using bicliques concept. Queries from different users are clustered in order to make the search histories more meaningful. On graphs random walks are applied in different ways in order to know the important nodes. A Markov random walk concept was applied in [21] and [3] for improving ranking.

III. Prototype Implementation

The prototype application is implemented using web interface. It is to demonstrate the usefulness of grouping search history of users. The environment used for the development is a PC with 4 GB of RAM, Core 2 dual processor running Windows XP operating system. Java technologies used are Servlets and JSP. We also used MVC (Model View Controller) design pattern for its benefits like scalability, availability and maintainability. The implementation of mechanisms is made as described in [3]. An important screen of the web application the organization of user search history is presented in fig. 4.

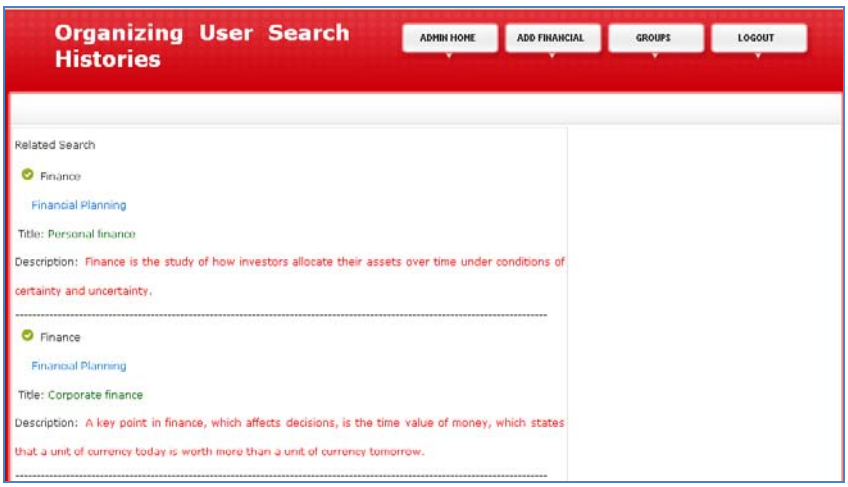

Figure 4 : Web based UI showing grouping of users' search history

As can be seen in fig. 2, the search queries of user's search history are grouped together as per the mechanism presented in section III. The visualization of search history is also presented in fig. 5.

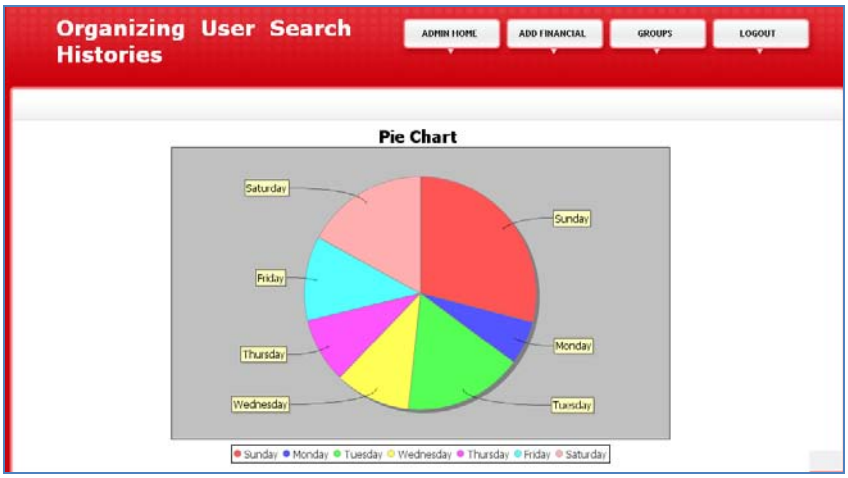

Figure 5 : Visualization of search history

As can be seen in fig. 2, it is evident that the user's search history is broken into different days. The search volumes are presented in a pie chart. This will reflect the user's search behavior on different days of a week. However, the subsequent section shows more experimental results.

IV. Experimental Results

Experiments are made based on different mix of click and query graphs, varying damping factor, varying click importance, varying related queries, varying similarity threshold, varying recency weight, and varying time threshold.

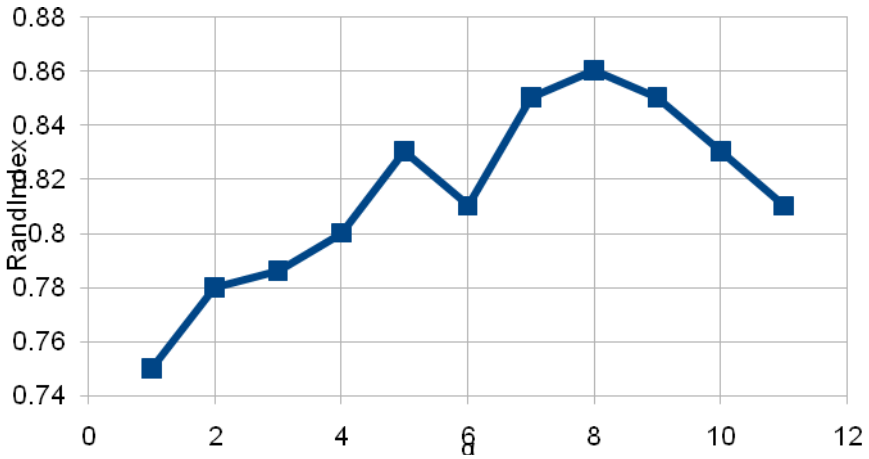

Figure 6 : Illustrates varying mix of query and click graphs

As can be seen in fig. 6, the horizontal axis represents weight of query edges that come from query reformulation graph while the vertical axis shows the performance based on RandIndex metric.

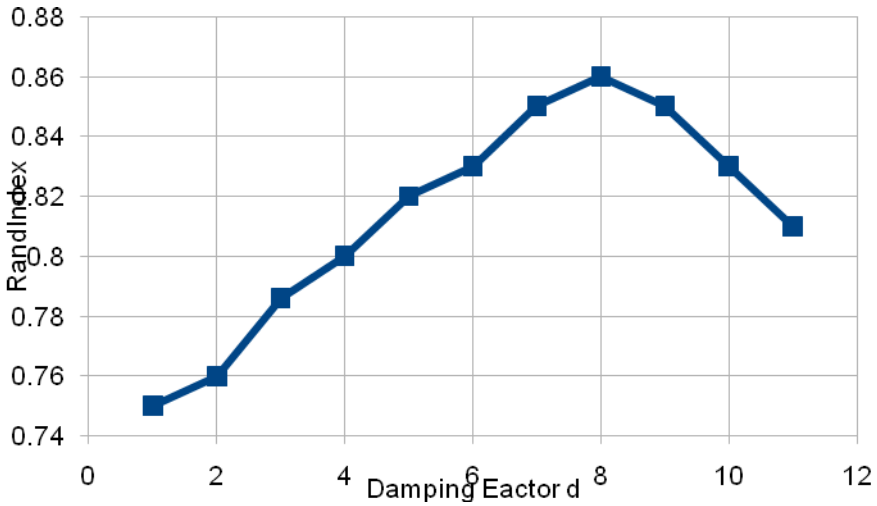

Figure 7 : Illustrates varying the damping factor

As can be seen in fig. 7, the horizontal axis represents damping factor while the vertical axis shows the performance based on RandIndex metric.

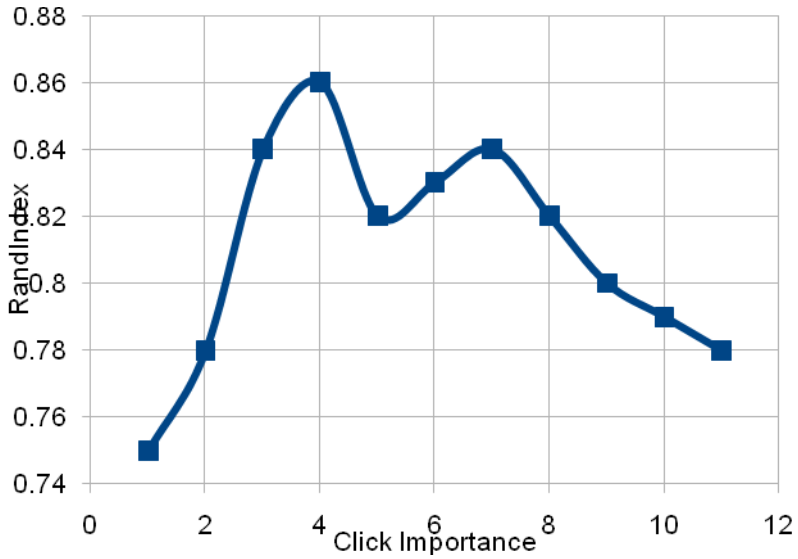

Figure 8 : Illustrates varying click importance

As can be seen in fig. 8, the horizontal axis represents click importance while the vertical axis shows the performance based on RandIndex metric.

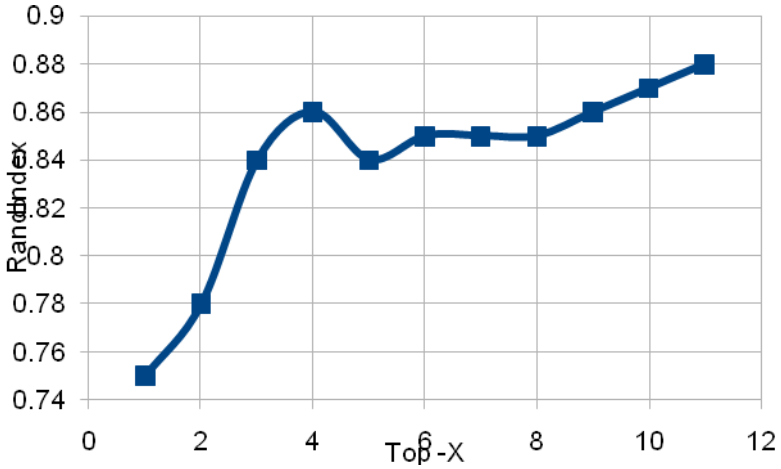

Figure 9 : Illustrates varying the fraction of related queries

As can be seen in fig. 9, the horizontal axis represents fraction of related queries while the vertical axis shows the performance based on RandIndex metric.

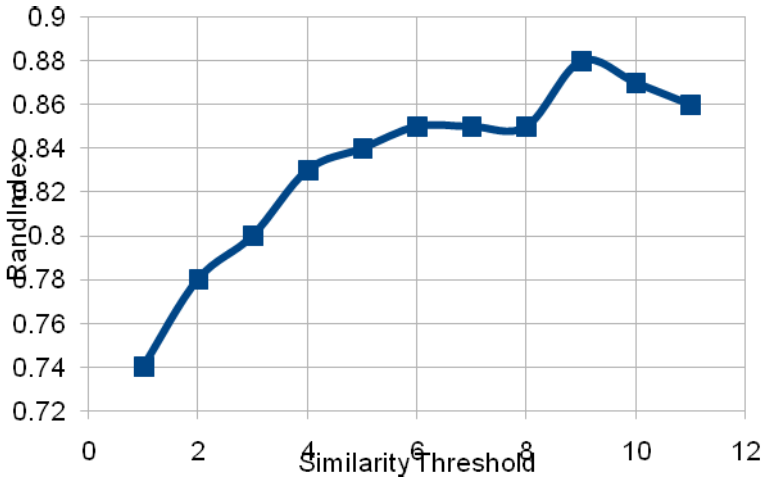

Figure 10 : Illustrates varying the similarity threshold

As can be seen in fig. 10, the horizontal axis represents similarity threshold while the vertical axis shows the performance based on RandIndex metric.

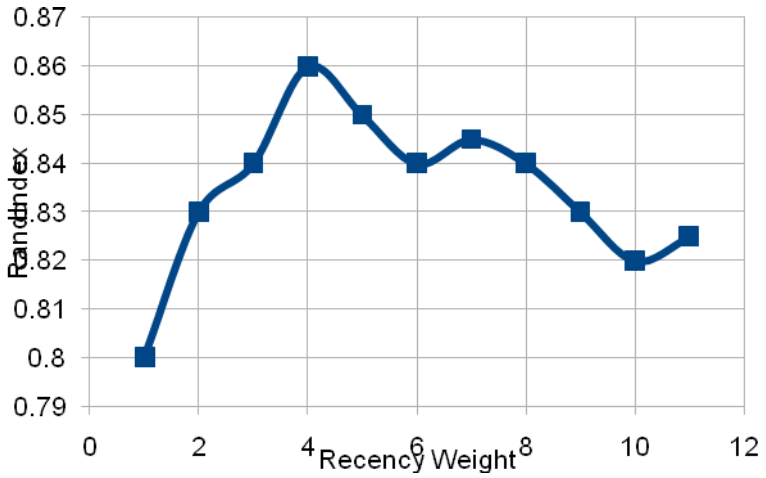

Figure 11 : Illustrates varying the recency weight

As can be seen in fig. 11, the horizontal axis represents recency weight while the vertical axis shows the performance based on RandIndex metric.

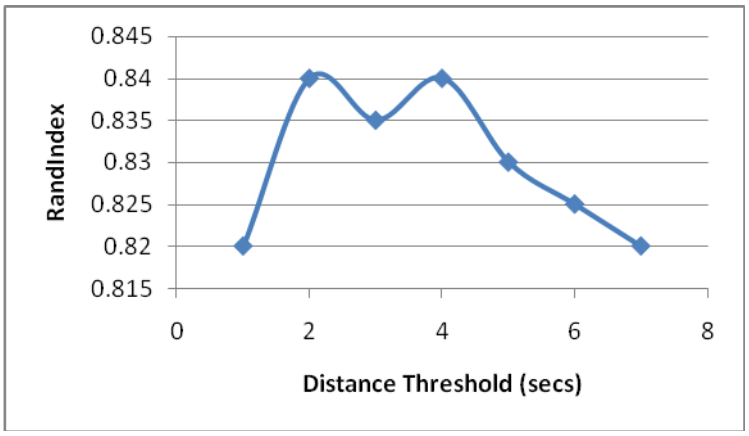

Figure 12 : Illustrates varying the time threshold

As can be seen in fig. 12, the horizontal axis represents time threshold while the vertical axis shows the performance based on RandIndex metric.

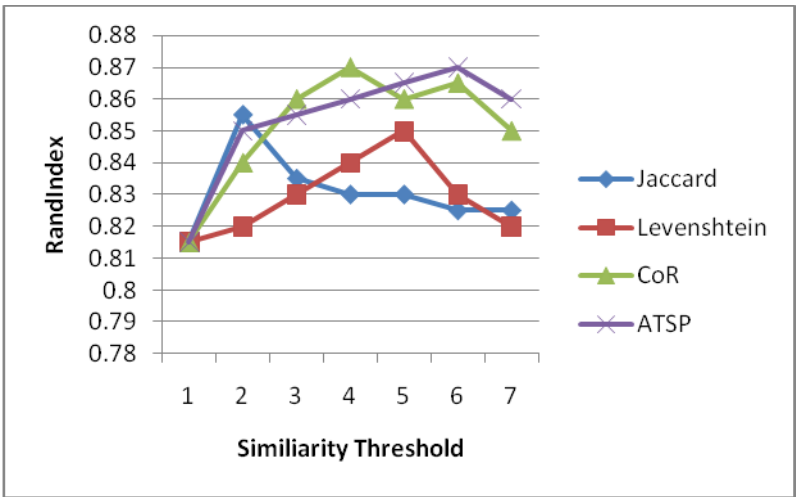

Figure 13 : Illustrates varying the similarity threshold

As can be seen in fig. 13, the horizontal axis represents similarity threshold while the vertical axis shows the performance based on RandIndex metric.

V. Conclusion

Search engines maintain historical data. However, they do not organize search histories well. They only present the search histories in chronological order. In this paper we implemented the mechanisms to group or organize user search history such as query reformulation and click graphs proposed by Hwang et al. [3]. Organizing user search histories have very important utilities. They include collaborative search, sessionization, query alterations, result ranking and query suggestions. The application we built demonstrates how the search histories of users are grouped together. Such organized search results are valuable to search engines for various applications mentioned above.

References Références Referencias

1. Broder, “A taxonomy of web search,” SIGIR Forum,. (2002). *36*, 3–10.
2. J. Teevan, E. Adar, R. Jones, and M. A. S. Potts, (2007). “Information reretrieval: repeat queries in yahoo’s logs,”. 151–158.
3. Heasoo Hwang, Hady W. Lauw, LiseGetoor and AlexandrosNtoulas, (2012). “Organizing User Search Histories”,. *24* (5).
4. R. Jones and K. L. Klinkner. (2008). “Beyond the session timeout: Automatic hierarchical segmentation of search topics in query logs”.
5. P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna, (2008). “The query-flow graph: Model and applications,” in CIKM,.
6. D. Beeferman and A. Berger. (2000). “Agglomerative clustering of a search engine query log”.
7. R. Baeza-Yates and A. Tiberi. (2007). “Extracting semantic relations from query logs,” .
8. P. Anick. (2003). “Using terminological feedback for web search refinement: A log-based study”.

9. B. J. Jansen, A. Spink, C. Blakely, and S. Koshman, (2007). "Defining a session on Web search engines: Research articles". *Journal of the American Society for Information Science and Technology,* 862–871.
10. L. D. Catledge and J. E. Pitkow, (1995). "Characterizing browsing strategies in the World-Wide Web". *27*, 1065–1073.
11. D. He, A. Goker, and D. J. Harper, (2002). "Combining evidence for automatic Web session identification". *Information Processing and Management, 38*, 727–742.
12. R. Jones and F. Diaz, (2007). "Temporal profiles of queries". *ACM Trans- actions on Information Systems, 25*, 14.
13. L. Montgomery and C. Faloutsos, (2001). "Identifying Web browsing trends and patterns". *34*, 94–95.
14. Silverstein, H. Marais, M. Henzinger, and M. Moricz, (1999). "Analysis of a very large Web search engine query log". *SIGIR Forum, 33*, 6–12.
15. H. C. Ozmutlu and F. C, avdur, (2005). "Application of automatic topic identification on Excite Web search engine data logs". *Information Processing and Management, 41*, 1243–1262.
16. T. Lau and E. Horvitz, (1999). "Patterns of search: Analyzing and modeling Web query refinement".
17. F. Radlinski and T. Joachims. (2005). "Query chains: Learning to rank from implicit feedback".
18. J.-R. Wen, J.-Y. Nie, and H.-J. Zhang, (2002). "Query clustering using user logs". *ACM Transactions in Information Systems, 20*, 59–81.
19. J. Yi and F. Maghoul. (2009). "Query clustering using click-through graph".
20. E. Sadikov, J. Madhavan, L. Wang, and A. Halevy, (2010). "Clustering query refinements by user intent".
21. N. Craswell and M. Szummer, (2007). "Random walks on the click graph," in *SIGIR*.