Volume 4, Issue 3, March 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

# Web Usage Mining for Heightening Search-Result Delivery and Assisting Users to Find Concerning Web Content

**Neha Singh, Prof Manish Varshney**
Department of Computer Science
Shri Siddhi Vinayak Group of Institutions, Bareilly, India

***Abstract— Web usage mining is the practical application of data mining proficiencies to the data generated by the interactions of users with web servers. This kind of data, stored in server logs, constitutes a worthfull source of data, which can be exploited to optimize the document-retrieval task, or to better realize, and thus, meet user needs.***
***Our research concentrates on two important consequences: improving search-engine functioning through static caching of search results, and assisting users to find concerning web pages by advocating news articles and blog posts. Referring the static caching of search results, we demonstrate the query dealing approach. The universal idea is to inhabit the cache with those documents that contribute to the result pages of a large number of queries, as fought to caching the top documents of most frequent queries. For the recommendation of web pages, we demonstrate a graph based approach, which leveraging the user-browsing logs to describe early adopters. These users discover concerning capacity before others, and monitoring their activity we can detect web pages to advocate.***

***Keywords— Web usage mining; Caching; Recommendation***

## 1. MOTIVATIONS FOR THE RESEARCH

The World Wide Web affords an abundance of data to the users, and this builds the recognition of applicable content difficult. Web mining attempts to cover this trouble. It comprises in the practical application of machine learning and data mining proficiencies, which allow for the automatic extraction of significant forms and relationships from large collections of web data. Web mining can be separated into three main subareas: (i) web content mining, which infers knowledge from the content (i.e. text and graphics) of web pages, (ii) web structure mining, which extracts data from data depicting the establishment of web content, and (iii) web us- age mining, which captures usage practices by examining data generated from the interactions of the users with the Web. There is not a clear-cut differentiation among these classes, and all three mining tasks can be combined. Our research work is powerfully related to the field of web usage mining. Particularly, we feat the implicit data given by users that are stored in query and browsing activity logs, in order to help web users to determine data easily and rapidly. The contributions we suggest are twofold: First, we face the trouble of amending search engine performance. Second, we tackle the problem of page recommendation.

**Improving search-engine performance**

Optimization of search-engine functioning is evidently of paramount importance, given that a typical search engine finds a huge number of queries every second, and users anticipate very low response times. To this end, search engines apply a variety of caching techniques as a means to provide results timely and with minimal reduction in quality. Several works suggest using a static cache of documents. This cache stores results that are relevant to the most frequent queries [23, 12]. Although this approach is very simple, it may expect a lot of cache space. In our research work, we suggest schemes for optimizing the space usage of the static cache.

**Web-page recommendation.**

Search engines help people to explore for data on the Internet, but the web search is effectual only when the users have a clear idea of what they want. Frequently, people have no particular information need, for example they surf the Web to read news, concerning blog posts, etc. Recommender systems produce suggestions, and they are effective in static and comparatively noise-free environments. The design of recommend systems for web content poses significant challenges due to the dynamic nature of web pages, and the high level of noise brought in by the analysis of the user-browsing data. We deal these problems by aiming a graph-based approach, which allows finding concerning news articles soon in order to advocate them to the web users.

## 2. RELATED WORK

In this Section, we present the state of the art for the main topics of our research.
Web usage mining. Web usage mining directs to catch, model, and examine the behavioral practices and visibilities of users acting with the Web. Data stored in usage logs can be used for solving navigation problem [24], bettering web search [3], advocating queries [5], proposing authoritative web sites [22], and enhancing performance of search engines [23]. A good survey of web usage mining can be found in [21].

 

Query-log analysis. Several analyzes have examined query log data to realize the conduct of search-engine users. In [20], pink et al. point out those users tends to present short queries and look at few pages of consequences. They choose to refine queries rather than to navigate answer pages deeper, and they do not feat advanced search characteristics.

Another significant finding is that the frequency of queries is power-law disseminated: a lot of queries are presented once or twice, while a few questions are submitted many times [19]. These questions are also popular, so they are shared by unlike user's reecting a spatial locality in the query stream [23]. Further studies emphasize a temporal locality in query stream: repeated compliances of queries on the same topic are broke by a small number of other questions [15, 12].

**Caching of query results.** The neighborhood properties, detected in the query stream, propose caching pre-computed reacts or partial data used in the computing of new answers. The work [17] feats frequent queries to compound the retrieval process. However, the simple quality-based cache may poorly address the issue of temporal locality [15]. In [12], the writers present a Static-Dynamic Cache (SDC), where the motionless portion has fixed entries to store results of commonest queries, and the dynamic portion is handled by a substitute policy to intercept the variations in the query stream. In [4], the writers examine the affect of the tail of the query distribution on caches of computer program.

**Diversification of search outcomes**. Outcome diversification is of use for equivocal queries (e.g., \python") and liberal queries with many possible user intents (e.g., \java programming language"). Some variegation schemes are established on removing documents that are like in order to avoid the redundancy of data [6, 8]. Others make explicit use of the knowledge about the different topics covered by queries and documents [1, 7].

**Web-page recommendation**. Recommendation systems appropriate to detect user tastes and to make recommendations. They can be utilized to advocate products (e.g., books, movies, music, etc.) and web content (e.g., news, photos, etc.). Recommendation systems can be: (i) content based, the system recommends items similar to the ones the user favored in the past, (ii) collaborative-filtering based, the system advocates items that people with similar tastes liked in the past, and (iii) hybrid, the system combines content and collaborative-filtering based methods.

In the last years, a lot of care has been committed to the recommendation of web news. In [18], the writers present a recommend system based on human minds. In [11], Das et al. name clumps of users with similar concerns by overworking the user-click data.

## 3. Suggested RESEARCH

In this Section, we depict the research consequences which we want to focus on. Then, we demonstrate our approaches to cover these issues, and we show some preliminary results, which confirm our proposals.

### 3.1 Amending search-engine functioning

Every day millions of users present questions to search engines and the quantity of data demanded in treating a query is vast. The users desire to receive a list of relevant and authoritative results rapidly, therefore the search engines employ caching echniques to provide high quality results in short time.

Advanced search engines cache sending lists of most frequent query conditions or the results of the queries. In the second case, depending upon the available space, the cache can store the total page of results or single documents. A drawback of the latter approach with respect to the former is that query response times will be higher, since result pages must be constructed by using the documents in cache. On the other hand, this approach presents the advantage that the same document can be used to serve multiple queries, so that the cache space is used efficiently. The search-result cache can be static or dynamic, and a combination of both techniques can be employed [12]. The static cache is subject to periodical updates and the documents to cache are precociously using historical information. The dynamic cache stores results assigning to the query sequence.

For our research employment, we deal the assumption of the static caching of inquiry results, where the cache depots single paperses (snippet and satellite data returned to users in reaction to their questions). We suggest query continuing as a scheme for efficaciously and efficiently caching results of questions. The estimate is to use statistics data about the user-querying behavior, so as to inhabit the cache with a set of documents that in the average will maximize the number of queries that are entirely served by the cache. We analyze the problem of query covering under a simple information retrieval model, in which each document is either relevant or irrelevant to a query [2]. Then, we better this approach by proposing a weighted version of query covering, where weights capture the degree of relevance of the documents to the queries. The weighted approach selects high-ranked documents, which appear in the result lists of several queries. This allows optimizing the cache space, ensuring the quality of the results. Finally, we prove that the burdened advance can be used for different purposes, such as the diversification of cached papers.

#### 3.1.1 Approach

We conceive a simple framework in which users present queries to a document retrieval system over time. The document retrieval system hosts a document collection. In addition, to improve response time, the system uses a document cache. Whenever a query is submitted, the system checks the cache, if all requested documents are stored in the cache, the system builds the final page of results, otherwise it incurs a cache miss, that is, a penalty reacting that the system has to carry out a time-expensive operation, such as the retrieval of documents from the document collection.

 

We presume that we have full cognition of the papers set, and in particular we know what paperses are applicable to what queries. Moreover, we assume to have no cognition about questions that will be presented in the future, but we have a comparatively good judge of their dispersion. We want to produce a ecumenical map from each question to a set of applicable documents for the question, so that, when a question is issued, the system can get the paperses from the cache and use them to do the current query as well as possibly next ones.

The trouble can be seen as a random abstraction of the set cover trouble [14], in which components represent to questions and paperses represent to sets. We need to determine a fixed function from components to crossing sets absent cognition of the elements to cover, since we have no a priori cognition of the future queries.

Our theoretic determining is that experiencing the question dispersion serves to furnish algorithmic program with logarithm estimates of the optimum result. In [2], we demonstrate that there survives a set of paperses that blankets a large divide of the questions and a simple avid advance is able to find it. Furthermore, we demonstrate a greedy algorithm, which takes the most cost-effective paperses, namely, the paperses coming along in the result lists of a large number of revealed queries. This greedy algorithm dismisses the degree of relevance of the papers to the query. Hence, we suggest overcoming this restriction by using a new approach, which companions weights to query-document pairs.

The burdened version of the greedy algorithmic program gives taste to the paperses that blanket the biggest total weight among the uncovered queries. We believe that the weighted approach is general and versatile, and it may be used for different optimization criteria by suitably defining weights. In particular, we propose two definitions of weights:

- **Document relevancy.** The weights respond the degree of relevancy of the papers to the question. The burden of the written document d for the query q diminishes as the position of d in the result list of q increases.

- **Document diversification.** We want to cache paperses maximizing the reportage of all potential intents behind user queries. Hence, the weight of a document depends on the position of the document within the result list and on the extent to which the document covers different intents for the query. Following Capannini et al. [7], we interpret the specializations of a given query as possible subtopics (intents) of that query. The weight of the document d for the query q is computed as the sum of the values of its utility score for every specialization q0 of q, each weighted by the chance of that transition between q and q0. The utility score captures the similarity between the document, which is a result of the query, and each document appearing in the result list of the query specialties.

### 3.1.2 Experiments

For the experiments, we apply a real-world question log dataset. We break up the dataset into periods of time (e.g., one day or one week). Questions presented in the $i^{th}$ time period form the aiming set, and they are used to learn the query distribution and to select a static collection of documents for the cache. Queries of the $(i + 1)^{th}$ time period, demonstrate the test set, and they are overworked to measure the performance of the query covering approach.

To evaluate the performance we use: (i) recall, the percentage of queries from test set covered by cached documents, (ii) num_doc, the number of cached documents, and (iii) p@n (precision-at-n), given a query in the test set, it is the proportion of the top-n results that are found in the cache. Then, we compute the average over all covered queries from test set.

We announce with Unweighted, the carrying out of the greedy algorithm demonstrated in [2]. For the burdened approach, we suggest to design two unlike effectuations which represent to different resolutions of covering: the query is crossed if the overall weight of its results is at least a given threshold W (Threshold algorithm), or if the number of results is at least a given cardinality k (Cardinality algorithm). We equate the public presentation of our approaches against $Top^k$ performance. $Top^k$ selects first k results for most frequent queries. This approach is naive, but it gives us an upper bound of the coverage that can be achieved with a given collection of documents.

The prelim consequences of recall and num_doc for k = 10, and W = 10:0, changing the percent of questions from aiming set we want to cover, are shown in Figure 1(a) and (b). As anticipated Topk has the best recall, nevertheless greedy approaches are following closely. Moreover, if we observe the proportion of the cache, for Topk the required space grows linearly with the number of queries in the training set that are covered, while the num_doc is optimized with crossing approaches, for which we observe a steep increase after 40% of covered queries. This leads us to an important observation: we can just cover a smaller fraction of queries from training set, for example 30% or 40%, to have a good recall and at the same time a limited usage of cache space.

The exactness (p@n) is the dimension of the top-n results determined in the cache. In Figure 1(c) is reported the p@n, for n = 5; 10, of the covered questions from the test set. As expected, good exactnesses are achieved with Topk, Threshold and Cardinality, while Unweighted performs worse.

### 3.2 Web-page recommendation

The systems for web-page testimonial are based on cooperative-filtering comings. The key estimate is to exploit the user-browsing logs (e.g. the click information), in order to distinguish users with similar sakes and tastes [11]. These recommendation systems deal with very high levels of noise, since visiting a web page is not a clear indication of interest as renting a DVD or buying a book. Further, the web pages to recommend are not an input of the recommendation algorithm, and the systems have to discover the concerning pages to recommend. We cover these issues by leverage not only the user similarity, but also the latent temporal patterns of user visits to web pages. In [16], we demonstrate a graph-

 

based approach called early-adopter graph. We suggest using this graph to identify those users who discover pertaining pages before others. These users are called early adopters, a term we adopt from social sciences, economics, and marketing research, in which early adopters are people who embrace new technologies before others, buy new products soon after their departure, and play an important role in inuencing others to adopt innovations. By tracking the browsing activity of early adopters we can identify new

Interesting pages early, and advocates these pages to users who share concerns with the early adopters. In particular, we consider news and blog pages, as these pages are more dynamic and more proper for testimonial tasks.

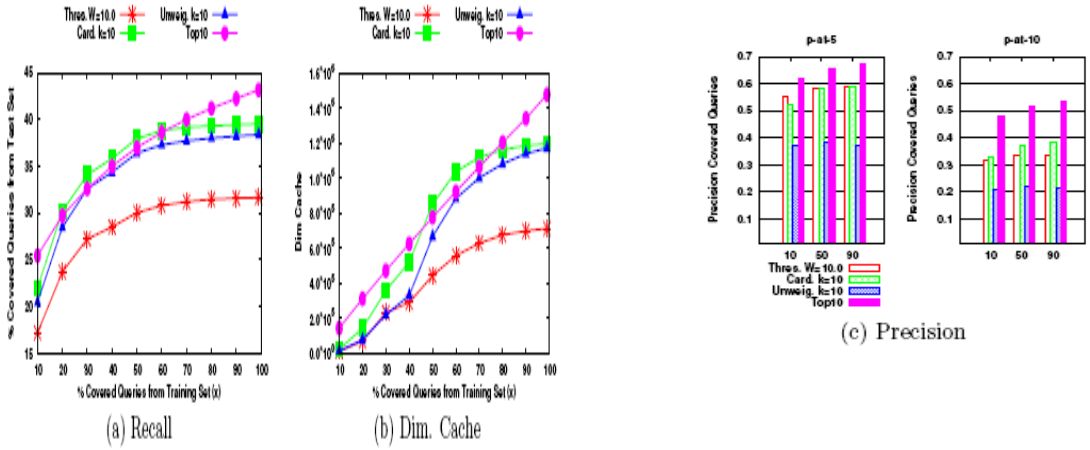


Figure 1: Recall, dimension of the cache, and precision-at-n vs. % of covered queries of training set.

### 3.2.1 Approach

The early-adopter graph G is an assigned, directed and burdened graph, where guests represent to users and an edge between two users, u and v, expresses the fact that the two users visited the same page and that the temporal rule \u visited the page before v" holds. We build the graph by extracting from a toolbar log the triples (u; p; t), where: (i) u is the anonymous identifier of the user, (ii) p is the url of the visited page, and (iii) t is the timestamp of the visit of u at p. Given a page pj , we organize the visits that pj received by all users in a chronologically-sorted access list A(pj) : h(u1; t1j); (u2; t2j); : : : ; (un; tnj)i, where tij is the timestamp of the first visit of user i to the page j. Obviously, the early adopters tend to appear at the beginning

of the access list. We propose different measures of the early adoption. More in detail, the early-adopter score (u) is computed by considering the position of the user in the access list (relative position), or the timestamp of the visit (time distance).

The edge weight w(u; v) represents the likelihood that a page visited by u is then visited by v. We propose and evaluate different edge-weighting schemes to compute the strength of the connections. The definitions of the edge weights are inspired by the Bernoulli and Jaccard measures described in [13]. The Bernoulli measure interprets each visit of u to a page as a hypothetical attempt of u to in- uence v in visiting the same page. The Jaccard measure considers also the pages visited by v and not by u, and it captures whether v follows mostly the actions of u and not many more.

Our recommendation approach leverages the information found in the early-adopter graph G. Given an arc (u; v) we consider suggesting to user v pages that have been visited by the user u. To improve the relevance of our recommendations, we rank recommendations by using the early-adopter score _(u) of the user u from whom the recommendation originates, as well as the edge weight w(u; v). Additionally, we use page topics to boost scores of pages whose topics match the interests of the user v. When a page p is suggested to v by different early adopters, the final recommendation score s(v; p) is the sum of the contributions of the early adopters for the user v.

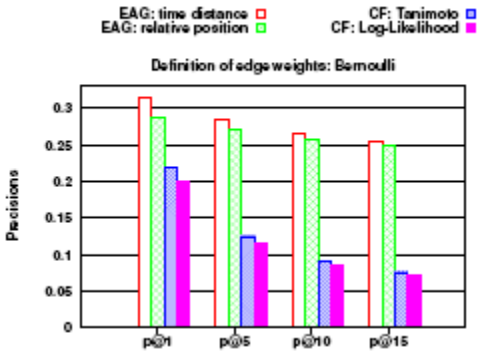


Figure 2: Precision of recommendations: early- adopter graph (EAG) vs. collaborative-filtering approaches (CF).

 

**3.2.2 Experiments**

We evaluate our testimonial algorithm using Yahoo! Toolbar dataset. We split our dataset, at the level of paginates, in two portions: training and test sets. The training subset is used to build the early-adopter graph, learn the early adopter scores, and the topics of interests of the users. Given a user v and the set of v's early-adopters, namely, the in-neighborhood of v (N(v)), the algorithm recommends to v the pages visited by u 2 N(v). Then, it ranks the pages by the recommendation scores.

The testimonials are measured with exactness-at- k (p@k) for k = 1; 5; 10; 15, which gives an indication of the percent of recommended pages that are actually visited by v. As we can see from Figure 2, the resulting system outperforms other out-of-the-shelf testimonial systems based on collaborative filtering, and where the user similarity is given by Animator or Log-Likelihood coefficients.

## 4. OPEN RESEARCH ISSUES

Open problems are related to the optimization of computer program functioning. First, it would be concerning to design and examine a dynamic caching policy able to cope with spikes in the query distribution. These spikes could be due to sudden events. Second, we propose to create a two level static cache, where a first-level cache stores documents relevant to most frequent queries, while a second-level cache stores documents chosen by the weighted greedy approach. For the diversification purpose, we aspire to find a dynamic approach which decreases the weights of paperses that satisfy topics already covered, in order to favor those documents that are relevant to uncovered topics.

For the variegation function, we aim to determine a dynamical approach which decreases the weights of paperses that meet topics already covered, in order to favor those documents that are relevant to uncovered topics.

Furthermore, we suggest investigation the application of crossing comes on to the optimization of other expressions of the web search. For example, significant search practicality is represented by the phrase queries. Such queries are characterized by quotation marks, and can be issued to the search engine in order to find documents that contain an exact sequence of words. The standard way to process a phrase query is using an index with positional information. However, processing phrase queries is complex and rather expensive, especially if the phrase presents stop words, which cannot be ignored. There are different techniques to optimize the phrase-query processing, and one of them is employing a phrase index. For large collection of documents, indexing all possible phrases is prohibitive, and a challenging task comprises in discovering the set of phrases to index. We conceive that the covering strategies can help to take this decision: given a set of potential questions, we extract phrases from them, and then we can use a greedy approach to describe those phrases which cover as many queries as possible. These phrases constitute good candidates to be indexed.

Referring the early-adopter graph, we plan to use dissimilar determine models to learn the edge weights. Then, we would like to look into the application of the early-adopter model to other domains.

**REFERENCES**

[1] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In WSDM, 2009.
[2] A. Anagnostopoulos, L. Becchetti, S. Leonardi, I. Mele, and P. Sankowski. Stochastic query covering. In WSDM, 2011.
[3] R. Baeza-Yates, C. Hurtado, and M. Mendoza. Improving search engines by query clustering. J. Am. Soc. Inf. Sci. Technol., 58(12):1793{1804, 2007.
[4] R. Baeza-Yates, F. Junqueira, V. Plachouras, and H. F. Witschel. Admission policies for caches of search engine results. In SPIRE, 2007.
[5] P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna. The query-ow graph: model and applications. In CIKM, 2008.
[6] A. Bookstein. Information retrieval: A sequential learning process. Journal of the American Society for Information Science, 34(5):331{342, 1983.
[7] G. Capannini, F. M. Nardini, R. Perego, and F. Silvestri. E_cient diversi_cation of web search results. Proc. VLDB Endow., 4:451{459, 2011.
[8] J. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In SIGIR, 1998.
[9] C. L. Clarke, N. Craswell, and I. Soboro_. Overview of the TREC 2009 Web Track. In TREC, 2009. [10] C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. B□uttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In SIGIR, 2008.
[11] A. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative _ltering. In WWW, 2007.
[12] T. Fagni, R. Perego, F. Silvestri, and S. Orlando. Boosting the performance of web search engines: Caching and prefetching query results by exploiting historical usage data. ACM Trans. Inf. Syst., 24(1):51{78, 2006.
[13] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. Learning inuence probabilities in social networks. In WSDM, 2010.
[14] F. Grandoni, A. Gupta, S. Leonardi, P. Miettinen, P. Sankowski, and M. Singh. Set covering with our eyes closed. In FOCS '08, pages 347{356. IEEE Computer Society, 2008.
[15] E. P. Markatos. On caching search engine query results. Computer Communications, 24(2):137{143, 2001.

 

[16] I. Mele, F. Bonchi, and A. Gionis. The early-adopter graph and its application to web-page recommendation. In CIKM, 2012.
[17] V. V. Raghavan and H. Sever. On the reuse of past optimal queries. In SIGIR, 1995.
[18] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative _ltering of netnews. In CSCW, 1994. [19] C. Silverstein, M. Henzinger, H. Marais, and M. Moricz. Analysis of a very large web search engine query log. In ACM SIGIR Forum, pages 6{12, 1999.
[20] A. Spink, D. Wolfram, M. B. J. Jansen, and T. Saracevic. Searching the web: the public and their queries. J. Amer. Soc. Inform. Sci. Tech., 52(3):226{234, 2001.
[21] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan. Web usage mining: discovery and applications of usage patterns from Web data. SIGKDD Explor. Newsl., 1(2):12{23, 2000.
[22] R. W. White, M. Bilenko, and S. Cucerzan. Studying the use of popular destinations to enhance web search interaction. In SIGIR, 2007.
[23] Y. Xie and D. O'Hallaron. Locality in search engine queries and its implications for caching. In IEEE Infocom 2002, pages 1238{1247, 2002.
[24] J. Zhu, J. Hong, and J. G. Hughes. Pagecluster: Mining conceptual link hierarchies from web log _les for adaptive web site navigation. ACM Trans. Internet Technol., 4(2), 2004.

 