

Usage-based PageRank for Web Personalization

Magdalini Eirinaki, Michalis Vazirgiannis

Athens University of Economics and Business, Dept. of Computing

Patision 76, 10434, Athens, Greece

{eirinaki, mvazirg}@aueb.gr

Abstract

Recommendation algorithms aim at proposing “next” pages to a user based on her current visit and the past users’ navigational patterns. In the vast majority of related algorithms, only the usage data are used to produce recommendations, whereas the structural properties of the Web graph are ignored. We claim that taking also into account the web structure and using link analysis algorithms ameliorates the quality of recommendations. In this paper we present UPR, a novel personalization algorithm which combines usage data and link analysis techniques for ranking and recommending web pages to the end user. Using the web site’s structure and its usage data we produce personalized navigational graph synopses (prNG) to be used for applying UPR and produce personalized recommendations. Experimental results show that the accuracy of the recommendations is superior to pure usage-based approaches.

1. Introduction – Motivation

The evolution of world wide web as the main information source for millions of people nowadays has imposed the need for new methods and algorithms that are able to process efficiently the vast amounts of data that reside on it. Users become more and more demanding in terms of the quality of information provided to them when searching the web or browsing a web site. The area of web mining, including any method that utilizes data residing on the web, namely usage, content and structure data, addresses this need. The most common applications involve the ranking of the results of a web search engine and the provision of recommendations to users of – usually commercial – web sites, known as web personalization.

PageRank is the most popular link analysis algorithm, used in order to rank the results returned by a search engine after a user query. The ranking is performed by evaluating the *importance* of a page in

terms of its connectivity to and from other important pages. In the past there have been proposed many variations of this algorithm, aiming at refining the acquired results. Some of these approaches, make use of the so called “personalization vector” of PageRank in order to bias the results towards the individual needs of every user searching the web.

In this work, we introduce PageRank in a totally different context, that of web personalization. Web personalization is defined as any action that adapts the information or services provided by a Web site to the needs of a user or a set of users, taking advantage of the knowledge gained from the users’ navigational behavior and individual interests, in combination with the content and the structure of the Web site [12]. In the past, many approaches have been proposed, based on pure web usage mining algorithms, markov models, or a combination of usage and content mining techniques.

Motivated by the fact that in the context of navigating a web site, a page/path is *important* if many users have visited it before, we propose a novel approach that is based on a personalized version of PageRank, applied to the navigational tree created by the previous users’ navigations. We personalize PageRank by biasing it to “favor” pages and paths previously preferred by many users. We prove that this hybrid algorithm can be applied to any web site’s navigational graph as long as it satisfies certain properties. Thus, it is orthogonal to any graph synopsis we may choose to model the user sessions with, such as a Markov Chain, higher-order Markov models, tree-like synopses, etc. This approach is therefore generic, proved to converge after a few iterations and thus provides fast results, whereas we can fluctuate between simplicity and accuracy by applying it to less or more complex web navigational graph models. More specifically, our key contributions, are:

- *UPR*, a novel usage-based personalized PageRank-style algorithm used for ranking the web pages of a site based on previous users’ navigational behavior.

- A method for creating personalized navigational graph synopses (*prNG*) to be used for applying *UPR*.
- A personalization method which combines usage and structure data for ranking and recommending web pages to the end user.
- A set of experimental results which prove that the incorporation of link analysis in the web personalization process improves the recommendations' accuracy.

The rest of this paper is organized as follows: In Section 2 we overview some related work in the areas of PageRank-based algorithms, web usage mining and personalization. In Section 3 we overview the basic properties and functionality of the PageRank algorithm, whereas in Section 4 we present our novel algorithm, *UPR* and prove its convergence. Section 5 describes the method of creating the Navigational Graph *NtG* and localized (personalized) Navigational Graph synopses *prNG* in order to provide personalized recommendations by applying *UPR*. Section 6 includes some experimental evaluation of the proposed approach, whereas in Section 7 we conclude with insights to our plans for future work.

2. Related Work

The PageRank algorithm [4] is the most popular algorithm proposed for ranking the results of a web search engine. Many variations have been proposed in this context, some of which make use of the so-called “personalization vector” in order to bias the results based on the query [1, 14, 24]. To the extent of our knowledge, however, this is the first approach that uses a PageRank-like algorithm in the context of web personalization, biasing the ranking based on the usage of a web site.

In our approach, we propose the use of web navigational graph synopses, such as Markov models. In the past, many researches have proposed the use of 1st order (Markov Chains) [3, 8, 25], higher-order [16], or hybrid [7, 9, 17, 26] Markov models. Even though Markov Chains provide a simple way to capture sequential dependence, they do not take into consideration the “long-term memory” aspects of web surfing behavior. Higher-order Markov models are more accurate for predicting navigational paths, there exists, however, a trade-off between improved coverage and exponential increase in state-space complexity as the order increases. What's more, such complex models often require inordinate amounts of training data. The hybrid models, that combine different order Markov models, require much more resources in terms of preprocessing and training. We should stress at this point that our approach is orthogonal to the type of synopsis one may choose. This synopsis may be a Markov model of any order (depending on the simplicity

and accuracy required), or any other graph synopsis, such as those proposed in [22, 23].

Apart from Markov models, there also exist many approaches that perform web usage mining for web personalization. The proposed systems are based on data mining algorithms such as association rules mining [12, 18], clustering [2, 21], sequential pattern discovery [6, 27], or frequent pattern discovery from a tree-like navigational graph [13, 28]. For an extensive overview of such approaches the user may refer to [10, 11].

3. Preliminaries & Review of PageRank

PageRank [4] is the most popular link analysis algorithm, used broadly for assigning numerical weightings to web documents and utilized from web search engines in order to rank the retrieved results. The algorithm models the behavior of a random surfer, who either chooses an outgoing link from the page he's currently at, or “jumps” to a random page after a few clicks. The PageRank of a page is defined as the probability of the random surfer being at some particular time step $k > K$ at this page. This probability is correlated with the *importance* of this page, as it is defined based on the number and the importance of the pages pointing to it. For sufficiently large K this probability is unique, as illustrated in what follows.

Consider the web as a directed graph G , where the N nodes represent the web pages and the edges the links between them. The random walk on G induces a Markov Chain where the states are given by the nodes in G , and M is the stochastic transition matrix with m_{ij} describing the one-step transition from page p_j to page p_i . The adjacency function m_{ij} is 0 if there is no direct link from p_j to p_i , and normalized such that, for each j :

$$\sum_{i=1}^N m_{ij} = 1 \quad (1).$$

As stated by the Perron-Frobenius theorem, if M is irreducible (i.e. G is strongly connected) and aperiodic, then M^k (i.e. the transition matrix for the k -step transition) converges to a matrix in which each column is the unique stationary distribution PR^* , independent of the initial distribution PR . The stationary distribution is the vector which satisfies the equation $PR^* = M \times PR^*$ (2), in other words PR^* is the dominant eigenvector of the matrix M .

Since M is the stochastic transition matrix over the web graph G , PageRank is in essence the stationary probability distribution over pages induced by a random walk on the web. As already implied, the convergence of PageRank is guaranteed only if M is irreducible and aperiodic [19]. The periodicity of M is guaranteed in practice in the web context, whereas the irreducibility is satisfied by adding a damping factor ($I-\epsilon$) to the rank

propagation (ϵ is a very small number, usually set to 0.15), in order to limit the effect of rank sinks and guarantee convergence to a unique vector. We therefore define a new matrix M' by adding low-probability transition edges between all nodes in G (we should point out that in order to ensure that M' is irreducible, we should remove any dangling nodes, i.e. nodes with outdegree 0):

$$M' = (1 - \epsilon)M + \epsilon u \quad (3)$$

In other words, the user may choose a random destination based on the probability distribution of u . This process is also known as *teleportation*. PageRank can then be expressed as the unique solution to Equation 2, if we substitute M with M' :

$$PR = (1 - \epsilon)M \times PR + \epsilon p \quad (4)$$

where p is a non-negative N -vector whose elements sum to 1.

Usually $m_{ij} = \frac{1}{\sum_{p_k \in \text{Out}(p_j)} |p_k|}$ and $u = \left[\frac{1}{N} \right]_{N \times N}$, i.e. the

probability of randomly jumping to another page is uniform. In that case, $p = \left[\frac{1}{N} \right]_{N \times 1}$. By choosing,

however, u , and consequently p , to follow a non-uniform distribution, we essentially *bias* the resultant PageRank vector computation to favor certain pages. Thus, u is usually referred to as the *personalization* vector. This approach is broadly used in the web search engines' context, where the ranking of the retrieved results are biased by favoring pages relevant to the query terms, or the user preferences to certain topic categories [1, 14, 24]. In what follows, we present a usage-biased version of PageRank algorithm, used for ranking the pages of a web site WS based on the navigational behavior of previous visitors.

4. Usage –based PageRank (UPR)

So far, PageRank has been used in the context of web search, either in its original form (assuming the user follows one of the outgoing links with equal probabilities, or uniformly jumps to a random page) or to its personalized form (adjusting the personalization vector to favor certain pages). In our approach, we use this algorithm in a totally different context, that of web personalization. We introduce a hybrid PageRank-style algorithm for ranking the pages of a web site in order to provide recommendations to each visitor. For this reason, we bias the computation using the knowledge acquired from previous users' visits, as they are discovered from the user sessions that are recorded in the web site's logs. To achieve this, we define both the transition matrix M and the personalization vector p in

such way that pages and paths previously visited by other users are ranked higher.

We consider the directed navigational graph $prNG$, where the nodes represent the web pages of the web site and the edges represent the consecutive one-step paths followed by previous users. Both nodes and edges carry weights. The weight w_i on each node represents the number of times page p_i was visited and the weight $w_{j \rightarrow i}$ on each edge represents the number of times p_i was visited right after p_j . $In(p_j)$ is the set of pages that point to p_j and $Out(p_j)$ is the set of pages pointed by p_j (i.e. p_j 's in-links and out-links respectively). We describe the construction of $prNG$ in more detail in the next section.

Following the aforementioned properties of the Markov theory and the PageRank computation, the usage-based PageRank vector UPR is the solution to the following equation:

$$UPR = (1 - \epsilon)M \times UPR + \epsilon p \quad (5)$$

The transition matrix M on NG is defined as the square $N \times N$ matrix whose elements m_{ij} equal to 0 if there does not exist a link (i.e. visit) from page p_j to p_i and

$$m_{ij} = \frac{w_{j \rightarrow i}}{\sum_{p_k \in \text{Out}(p_j)} w_{j \rightarrow k}} \quad (6)$$

otherwise. The personalization vector p is defined as

$$p = \left[\frac{w_i}{\sum_{p_j \in WS} w_j} \right]_{N \times 1} \quad (7)$$

Using the aforementioned formulas, we bias the PageRank calculation to rank higher the pages that were visited more often by previous users. We then use this hybrid ranking, combining structure and usage data of the site, to provide a ranked recommendation set to current users.

Note that Equation (1) holds, that is, M is normalized such that the sum of its columns equal to 1, therefore M is a stochastic transition matrix, as required for the convergence condition of the algorithm to hold. M is as already mentioned aperiodic in the web context and irreducible, since we have included the damping factor $(1 - \epsilon)$. We also eliminate any dangling pages by adding to all nodes with no outlinks, links to all other pages with uniform probabilities. It is therefore guaranteed that Equation 5 will converge to a unique vector UPR^* .

Definition (UPR): We define the usage-based PageRank (UPR) of a web page p_i as the n -th iteration of the following recursive formula:

$$UPR^n(p_i) = \epsilon \sum_{p_j \in In(p_i)} \left(UPR^{n-1}(p_j) \times \frac{w_{j \rightarrow i}}{\sum_{p_k \in Out(p_j)} w_{j \rightarrow k}} \right) + (1 - \epsilon) \frac{w_i}{\sum_{p_j \in WS} w_j} \quad (8)$$

computations and would require more time than is allowed for an online process. Therefore, we create a synopsis of NtG , which is subsequently used for extracting the localized $prNG$. The more detailed the synopsis is, the more accurate will the representation of NtG be. On the other hand, the construction of a less detailed synopsis will save time and computational power.

The simplest synopsis of NtG is a Markov Chain. Then, the edge weight $w_{i \rightarrow j}$ is computed as the sum of the weights of all directed edges between nodes p_i and p_j , and the node weight w_i is computed as the sum of all the weights of edges pointing to the multiple occurrences of p_i :

$$w_i = \sum_{k \in \text{In}(p_i)} w_{k \rightarrow i} \quad (9).$$

This representation is simple to construct, depends, however, on the assumption that the navigation is “memoryless”, in other words that the next page to be visited by a user only depends on the page he’s currently at. NG synopses that take into consideration the “long-term memory” aspects of web surfing are higher order Markov models, which can easily be constructed from NtG by computing the k -step path frequencies (where k is the order of the model). In Figure 3 we present the NG synopsis of the NtG of Figure 2 if we choose to model the synopsis as a Markov Chain. The number in parentheses in the nodes denote the number of times the page was visited, whereas the edges’ weights denote the times the respective path was visited. Nodes S and E represent the start and end point respectively. These special nodes are not used when applying UPR on the graph.

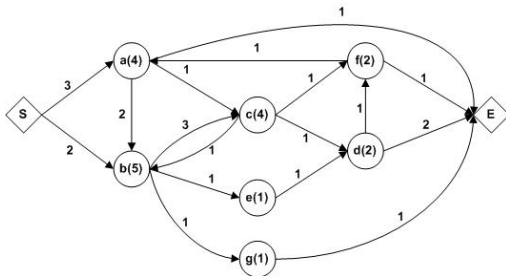


Figure 3. NG synopsis (Markov Chain)

Note that, for any NG synopsis we choose to create, we should eliminate any dangling pages in order to retain the irreducibility of the graph. This is accomplished if, for every page with no outlinks, we add a link to every other page and distribute uniformly the weights. This process is common for any NG graph.

5.2. Localized UPR (l-UPR)

In the previous section we presented UPR algorithm, which can be applied to a web site in order to rank its web pages based both on its link structure and

the paths followed by previous users. This process provides us with a “global” ranking of the web site’s pages. In the context of web site personalization, however, we want to “bias” this algorithm further in order to rank the web pages focusing on the path the current visitor has followed, i.e. producing a “local” personalized ranking. In order to achieve this, we select a small subset of the NG synopsis created in the previous step based on the path the current user has followed so far. We then apply a location-oriented version of UPR , localized UPR (l -UPR) to this subgraph. The resulting ranking is used in order to provide recommendations to the current visitor. This approach enables the online application of l -UPR since the size of the graph is dramatically reduced, as compared to applying UPR to NG . Moreover, the ranking results will be personalized, i.e. based on the current user’s path and similar users’ behavior in the past. The recommendation process is described in more detail in the next section. Here, we describe how we construct the subgraph in order to apply l -UPR.

5.2.1. Construction of $prNG$. It is obvious that we should efficiently “choose” the fraction of the web site on which we want to apply PageRank so that the resulting graph is strongly connected, and therefore irreducible. In short, the process of constructing the personalized subgraph is as follows: Using the path already followed by the user, we expand it, starting from the last page(s) visited, to include all pages (and weighted edges) that are linked from this page in NG synopsis. The length of the path taken into consideration when expanding the graph, depends on the desired “memory” of the system. We subsequently perform this operation for the new pages, until we reach a predefined depth. We then remove any pages that have already been visited by the user, since we don’t want to include them in the final ranking. We keep, however, the remaining path (if it exists) and link it to the previous page. This ensures that we keep all pages visited before by users with similar behavior, without including any higher-level pages used as hubs for their navigation. Finally, we eliminate any dangling pages by adding links to all other pages in the subgraph. This process is shown in Figure 4.

5.2.2. l-UPR. We want to apply UPR in the resulting subgraph in order to rank all the pages that can be found in the outgoing paths of a certain depth, for further producing a recommendation set. To do that, we should ensure that the transition matrix L of the subgraph is strongly connected, and that Equation 1 holds. The removal of dangling pages by adding links to all pages for pages with no outgoing links guarantees the graph’s irreducibility. Note here that $prNG$ does not include the

previously visited pages. Equation 1 is satisfied if we normalize the outgoing weights of each node. We then apply *UPR* to *prNG* in order to calculate the rankings of the candidate pages for recommendation. The recommendation process is explained in more detail in the subsequent section.

```

Procedure Create_prNG(CV, NG)
Input: Current User Visit CV, Navigational Graph NG
Output: Subset of NG prNG
1. start
2. CV = {vp};
3. cp = lastVisited(CV);
4. expNG = expand(cp, NG, depth);
5. removeVisited(expNG, CV);
6. updateEdges(expNG);
7. prNG = eliminateDangling(expNG);
8. end
#####
Procedure expand(cp, NG, depth, eNG)
Input: last page/path visited cp, navigational graph synopsis NG, depth of expansion depth
Output: expanded navigational graph eNG
1. start
2. P := cp;
3. R:= rootNode(eNG);
4. tempd = 0;
5. addNode(eNG, R, cp);
6. while (tempd <= depth)do
7. for every (p∈P of same level)do
8. forevery np = linksto(NG, p, np, w)do
9. addNode(eNG, p, np, w);
10. P += np;
11. done;
12. done;
13. tempd +=1;
14. done;
15. end

```

Figure 4. Construction of *prNG*

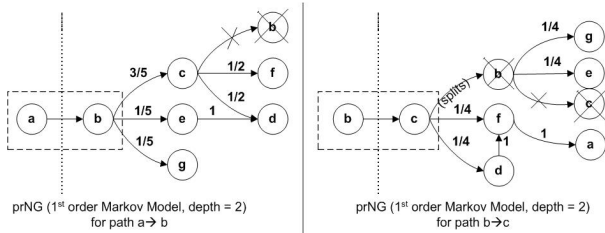


Figure 5. *prNG* of Markov Chain *NG* synopsis

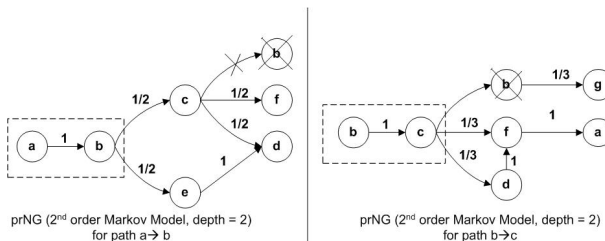


Figure 6. *prNG* of 2nd order Markov model *NG* synopsis

To explain the aforementioned algorithm, we present the *prNG* for paths {*a*→*b*} and {*b*→*c*} for a Markov Chain and a 2nd order Markov model in Figures 5 and 6 respectively. Note that the resulting nodes to be

ranked by *l-UPR* differ depending on the *NG synopsis* we choose. The edges' weights are normalized, whereas we don't include the edges from dangling nodes to all other nodes for simplicity.

5.3. Personalized Recommendations

Based on what was previously presented, the final set of candidate recommendation pages can be created by following 3 different methods:

- 1) Use *NG synopsis* to compute *prNG path probabilities*. If for example *NG synopsis* model is a Markov Model, then the prediction is straightforward and trivial, since it is based on the path probabilities produced using the chain rule (common to Markov theory).
- 2) Apply *UPR* to *NG synopsis*.

This approach provides us with a “global” usage-based ranking of the pages of our web site.

- 3) Apply *l-UPR* to *prNG*.

This approach provides us with a “personalized” usage-based ranking of the pages based on the current user's path.

At the end of the aforementioned process, all pages of *NG* or *prNG* will be ranked in descending order based on their importance (as defined by their linkage, previous usage, and current user's path). Before providing the list of recommendations to the end user we may also want to remove all pages the user has already visited (this is guaranteed by our proposed algorithm, i.e. in methods 2 and 3), and/or remove all pages that are already contained as links in the page. Moreover, in the case of none or not enough candidate pages produced using the proposed method (3rd) we use the “global” ranking of the web site in order to provide recommendations. A further consideration would be to have a pre-computed set of recommendations for all popular paths in the web site, in order to save time when computing online the final recommendation set. In the experimental evaluation that follows we compare the outcomes of methods 1 and 3.

6. Experimental Evaluation

As already mentioned, *UPR* algorithm as well as the process of creating personalized sub-graph synopses (*prNG*) and applying *l-UPR* is orthogonal to the navigational graph synopsis we may choose to model our data with. In this work, we choose the Markov Chain to synopsise the Navigational Tree *NtG*. We present here initial results regarding the impact of incorporating our proposed method in the Markov Chain prediction model. More specifically, we select the top-*n* most popular paths, as derived from the web logs. For these paths we compute a recommendation set, using two variations of Markov Chains and our algorithm

applied on a Markov Chain synopsis. We then compare the recommendation sets with the actual paths followed by the users.

6.1. Experimental Setup

For our experiments we used the publicly available *msnbc.com* dataset [20]. This dataset includes the page visits of users who visited *msnbc.com* on 28/9/99. The visits are recorded at the level of URL category (for example sports, news, etc.). It includes visits to 17 categories (i.e. 17 distinct pageviews). We selected 65,000 distinct sessions of length more than one and less than 50 pageviews visited. *msnbc.com* includes the visits to a very big portal. This means that the number of sessions, as well as the length of recorded paths is very big. This dataset has however the characteristic of very few pageviews, since the visits are recorded at the level of page categories. We therefore expect that the category-level web graph includes links from/to almost any node (category), and the visits to this web site are homogeneously distributed among the 17 different categories.

We created 3 different setups for producing recommendations. The first two (further referred to as *start* and *total*) are the ones commonly used in Markov models for computing prior probabilities. More specifically, *start* assigns prior probability to a page proportional to the visits to this page as start page of the navigation, whereas *total* assigns prior probability to a page proportional to the total visits to it. The third setup, further referred to as *upr* is in essence our proposed algorithm applied to a Markov Chain *NG* synopsis. For the *upr* setup, we set the damping factor ($1-\epsilon$) to 0.85 and the number of iterations to 100. We also expand each path to depth $d=2$.

Using the frequency of path visits as derived from the session data, we select the m most popular paths followed by the users of the web site. Assuming these paths have already been visited, we create a set of recommendations using the three aforementioned approaches. For the first two setups, we compute the probability of visiting a page given the user has already visited the path. We then select the n most probable pages. For the third setup, we expand the subgraph and apply *l-UPR* to rank the pages included in it. We then again select the n higher ranked pages. Finally, we compare these results with the n most frequent next pages of each path, as derived from the training data (i.e. the actual paths followed by the users). We use two metrics for comparing two top- n rankings r_1 and r_2 . The first one, denoted as $OSim(r_1, r_2)$ indicates the degree of overlap between the top- n pages of two sets A and B (each one of size n) to be $\frac{|A \cap B|}{n}$ [14]. The second,

$KSim(r_1, r_2)$ is based on Kendall's distance measure [15] and indicates the degree to which the relative orderings of two top- n lists are in agreement:

$$KSim(r_1, r_2) = \frac{|(u, v) : r_1', r_2' \text{ have same order of } (u, v), u \neq v|}{|A \cap B|(|A \cap B| - 1)} \quad (12)$$

where r_1' is an extension of r_1 , containing all elements included in r_2 but not r_1 at the end of the list (r_2' is defined analogously) [14].

6.2. Recommendations' Accuracy Evaluation

We selected the 10 most popular paths and created a top-3 and a top-5 recommendation list for each setup. We compared each one of them from every setup (*upr*, *start*, *total*) to a reference recommendation list representing the most frequent "next" pages, as derived from the session data, using *OSim* and *KSim* metrics. Figures 7 and 8 depict the average *OSim* and *KSim* values for the top-3 and top-5 rankings of the 10 paths with respect to the reference list. In the first case (top-3 page predictions) we observe that *upr* behaves better in terms of prediction accuracy (*OSim*). Based on the respective lower *Ksim* we may conclude that *upr* managed to predict the "next" pages but not in the same order. We should point out, however, that the ordering is not so important in such a small recommendation list. In the second case (top-5 page predictions), we observe that *upr* behaves better in both prediction accuracy and relative ordering than the other two approaches.

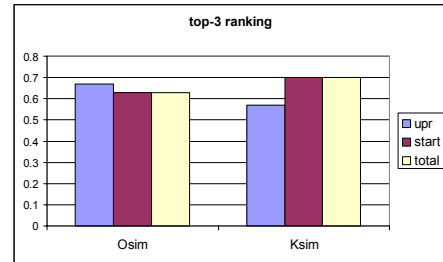


Figure 7. Average *OSim* and *KSim* of top-3 rankings

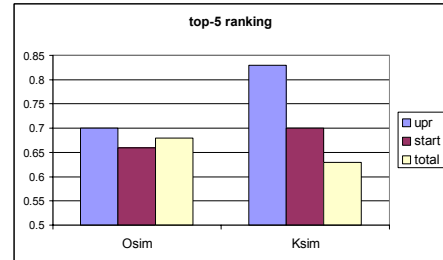


Figure 8. Average *OSim* and *KSim* of top-5 rankings

We chose to perform the experiments using small recommendation sets because this resembles more to what happens in reality, i.e. the system recommends only a few "next" pages to the user. Even though at a first glance the differences may seem insignificant, the

importance of the outperformance of *upr* is justified if we take into account the nature of the dataset used. As already mentioned, the number of distinct pageviews of the dataset is very small and therefore the probability of coinciding in the predictions is quite high. Nevertheless, in both experiments, the results confirm our claim that the incorporation of structure data in the prediction process enhances the accuracy of the recommendations.

We should also stress at this point that the whole process of expanding a given path and ranking the possible “next” pages takes less than 2 seconds, and therefore the algorithm may be applied online without delaying the users’ navigation.

7. Conclusions – Future Work

There is a wealth of recommendation models for personalizing a web site based on previous users’ navigational behavior. Most of the models, however, are solely based on usage data ignoring the link structure of the web graph visited. In this paper we propose a novel algorithm, *UPR*, which can be applied to any navigational graph synopsis, to provide ranked personalized recommendations to the visitors of a web site, capitalizing on the structural properties of the navigation graph. The first experimental results are very promising. We plan to evaluate the proposed algorithm using more datasets and other navigational graph synopses, as described before.

8. References

[1] M.S. Aktas, M.A. Nacar, F. Menczer, *Personalizing PageRank Based on Domain Profiles*, in Proc. of WEBKDD 2004 Workshop, August 2004, Seattle, USA

[2] R. Baraglia, F. Silvestri, *An Online Recommender System for Large Web Sites*, in Proc. of ACM/IEEE WI’04 Conference, China, September 2004

[3] J. Borges, M. Levene, *Data Mining of User Navigation Patterns*, in Revised Papers from the International Workshop on Web Usage Analysis and User Profiling, LNCS Vol. 1836, pp.92-111, 2000

[4] S. Brin, L. Page, *The anatomy of a large-scale hypertextual Web search engine*, Computer Networks, 30(1-7): 107-117, 1998, Proc. of WWW7 Conference

[6] A.G. Buchner, M. Baumgarten, S.S. Anand, M.D. Mulvanna, J.G. Hughes, *Navigation pattern discovery from Internet data*, in Proc. of WEBKDD’99 Workshop, August 1999, San Diego, CA

[7] I. Cadez, S. Gaffney, P. Smyth, *A general probabilistic framework for clustering individuals and objects*, in Proc. of ACM KDD2000 Conference, Boston, 2000

[8] I.Cadez, D.Heckerman, C.Meek, P. Smyth, S. White, *Visualization of Navigation Patterns on a Web Site Using Model Based Clustering*, in Proc. of ACM KDD2000 Conference, Boston MA, 2000

[9] M. Deshpande, G. Karypis, *Selective Markov Models for Predicting Web-Page Accesses*, in Proc. of the 1st SIAM International Conference on Data Mining, 2001

[10] M. Eirinaki, *Web Mining: A Roadmap*, Technical Report, DB-NET 2004, available at <http://www.db-net.aueb.gr>

[11] M. Eirinaki, M. Vazirgiannis, *Web Mining for Web Personalization*, in ACM TOIT, 3(1), February 2003, pp.1-29

[12] M. Eirinaki, M. Vazirgiannis, I. Varlamis, *SEWeP: Using Site Semantics and a Taxonomy to Enhance the Web Personalization Process*, in Proc. of ACM KDD2003 Conference, August 2003, Washington DC

[13] M. El-Sayed, C. Ruiz, E.A. Rundensteiner, *FS-Miner: Efficient and Incremental Mining of Frequent Sequence Patterns in Web Logs*, in Proc. of WIDM ’04, November 2004, Washington DC

[14] T. Haveliwala, *Topic-Sensitive PageRank*, in Proc. of WWW2002 Conference, Hawaii USA, May 2002

[15] M. Kendall, J.D.Gibbons, *Rank Correlation Methods*, Oxford University Press, 1990

[16] M. Levene, G. Loizou, *Computing the Entropy of User Navigation in the Web*, in Intl. Journal of Information Technology and Decision Making, 2:459-476, 2003

[17] E. Manavoglou, D. Pavlov, C.L. Giles, *Probabilistic User Behaviour Models*, in Proc. of ICDM 2003

[18] F. Masseglia, P. Poncelet, M. Teisseire, *Using Data Mining Techniques on Web Access Logs to Dynamically Improve Hypertext Structure*, in ACM SigWeb Letters, Vol. 8, N. 3, pp. 13-19, October 1999

[19] R. Motwani and P. Raghavan. *Randomized Algorithms*, Cambridge University Press, United Kingdom, 1995.

[20] *msnbc.com* Web Log Data, available from *UCI KDD Archive*, <http://kdd.ics.uci.edu/databases/msnbc/msnbc.html>

[21] M. Perkowitz, O. Etzioni, *Towards Adaptive Web Sites: Conceptual Framework and Case Study*, in Artificial Intelligence 118[1-2] (2000), pp. 245-275

[22] N. Polyzotis, M. Garofalakis, *Structure and Value Synopses for XML Data Graphs*, in Proc. of the 28th VLDB Conference, 2002

[23] N. Polyzotis, M. Garofalakis, Y. Ioannidis, *Approximate XML Query Answers*, in Proc. of SIGMOD 2004, Paris, France, June 2004

[24] M. Richardson, P. Domingos, *The Intelligent Surfer: Probabilistic Combination of Link and Content Information in PageRank*, in Neural Information Processing Systems, 14, pp.1441-1448, 2002

[25] R.R. Sarukkai, *Link Prediction and Path Analysis Using Markov Chains*, in Computer Networks, 33(1-6): 337-386, 2000

[26] R. Sen, M. Hansen, *Predicting a Web user’s next access based on log data*, in Journal of Computational Graphics and Statistics, 12(1):143-155, 2003

[27] M. Spiliopoulou, L.C. Faulstich, and K. Wilkner, *A data miner analyzing the navigational behaviour of Web users*, in Proc. of the Workshop on Machine Learning in User Modelling, Greece, July 1999

[28] Q. Zhao, S.S. Bhowmick, *Mining History of Changes to Web Access Patterns*, in Proc. of PKDD 2004, Italy, September 2004