Predictive Algorithms for Browser Support of Habitual User Activities on the Web

Janez Brank
Natasa Milic-Frayling
Anthony Frayling
Gavin Smyth

18 November, 2004

Technical Report
MSR-TR-2004-122

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Predictive Algorithms for Browser Support of Habitual User Activities on the Web

ABSTRACT

Routine activities that users perform on the Web result in the revisitation of sites and pages. Standard browser applications provide limited support for this type of habitual behaviour. They typically expose collections of visited URLs that are automatically recorded by the system, for example the navigation history, or those manually created by the user, such as bookmarks. Studies have shown that these approaches are not very successful in supporting the user in site or page revisitation. Informed by the findings of our user research and analysis of the user navigation logs, we designed SmartFavourites, a browser feature that automatically exposes candidate URLs for revisitation, in a context sensitive manner. In this paper we describe and evaluate the algorithms that we use to model the user's habitual behaviour. We demonstrate that the use of a structured navigation history model, which essentially captures the domain specific features, facilitates the discovery of relevant usage patterns and predictive algorithms that are applicable to relatively small sizes of personal navigation history.

Categories and Subject Descriptors

H.5.4 [**Hypertext/Hypermedia**]: Navigation, User Issues.

General Terms

Algorithms, Performance, Design, Experimentation, Human Factors, Verification.

Keywords

Revisitation, predictions, navigation history, browsing, web trails, bookmarks, user study.

1. INTRODUCTION

The Web has become an integral part of business practice and social interaction. Consequently, users regularly access the Web to conduct a range of activities. Some of these activities are performed repeatedly over a period of time. For example, a user's daily routine may involve online banking, monitoring of auction sites, news reading, online entertainment, and similar activities. A person's profession may require regular access to online tools, services, and information resources. Thus, there are sites and pages that users regularly visit, with varying frequency and over different periods of time. In providing support for Web access, it is important to take into account this habitual behaviour.

However, the standard approach of analyzing user logs by treating the navigation history as a simple sequence of URL visits does not help with identifying patterns underlying such activities. Typical analyses of URL accesses reveal highly skewed distribution of revisited URLs, dominated by those that occur in the short term navigation history. These are mostly due to the Back navigation. In order to identify patterns in site and page revisits resulting from a user's habitual activities over a longer period of time, we need to take a different approach. We use a model of the navigation history that derives structure from the user's interaction with the browser as exploited in [10]. In particular, we use the concept of a *Web trail* to approximate the notion of a user activity and, through analysis of user logs, exploit the patterns associated with Web trails.

One can speculate that, with large volumes of individual user's data, it might be possible to identify activity patterns even from URL-level statistics. However, in practice, the quantity of data available for individual users is relatively small. Thus, we need to incorporate application specific information, such as types of user interaction, to compensate for the lack of sufficient data.

The second important issue is the mechanism for delivering support to the user. In theory, effective browsing and search over the user's navigation history should enable revisitation of any site or page that the user has seen before. However, it is not clear that a query-based search would be a preferred method of revisiting pages since query formulation could easily distract from the user's primary task. One mechanism that has been successful in exploiting records of previous ULRs is through URL typing aids, such as the Auto-complete feature in Microsoft Internet Explorer. Auto-complete automatically performs a string search over the stored URLs to match the sequence of characters typed in by the user. With this in mind, we focus our research on supporting habitual revisits to sites and pages in a way that does not require an explicit request for URLs.

Building upon the concepts of the enhanced model of the user's navigation history used in [10] we devise novel algorithms to discover patterns of the user's activities and proactively expose the URLs in a link bar, predicting the possible activities that the user may want to perform next. We present the details of the algorithm evaluation, demonstrating the benefit of the underlying history model.

In the following sections of the paper we first provide an overview of relevant research and describe our approach to analyzing the user navigation history. We continue with a detailed description of the algorithms and experimental set up and, in Section 4, present experimental results. We conclude with a summary of insights gained and plans for future work.

2. BACKGROUND RESEARCH

A number of studies on Web access have shown that the users frequently revisit pages they have seen in the past. Cockburn and McKenzie [5] reported that, on average, 81% of Web accesses are page revisits. A figure of 61% was reported by Catledge and Pitkow [4], and 58% in the study by Tauscher & Greenberg [17].

A high proportion of revisitations are due to revisits of pages from the recent navigation history. Tauscher & Greenberg [17] estimate 43% chance that the next user navigation step will be a page from

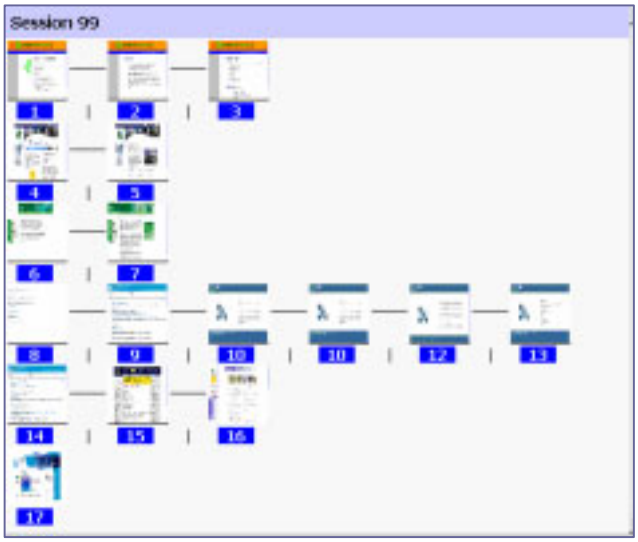

Figure 1. Example of multiple linear navigation trails within a single navigation session.

the 10 most recently accessed URLs. Those URLs that were visited in the more distant past have only a 15% chance of reappearing in the very next step.

In [10], Milic-Frayling et al pointed out that some of the history visits are transient back navigations due to the underlying hyperlinked structure of the Web. Indeed, as in hub and spoke navigation, the users often need to retrace navigation steps to return to a page from which they can explore alternative paths through the hypertext. These transient revisits could be eliminated by providing direct access to navigational hubs or other important pages in the navigation history.

While users continuously visit new pages [5] [17] (with 42% chance that the next visit will be a new URL [17]) they also do return to pages from the past. Indeed, about 26% of revisited URLs are not among 10 most recent URLs, not even in the top 20 or 30 [17]. At the moment, the user can access these pages by typing in a URL or by using a bookmark to a page, if one has been created by the user.

Some users often bookmark pages for future use. In fact, creating bookmarks is relatively easy and, in absence of more specialized features, they are used to accomplish various functions, from 'task lists' to the archive of 'not to be lost' links. Generally they have been viewed as a means of creating personal information spaces on the Web [1]. However, as such, they have not been effective in supporting users' repeated activities. The list of bookmarks becomes unmanageable as it grows in size. It is populated with references to pages that support transient rather than repeated tasks. The need for their explicit management by the user and in-time marking increases the overhead of their usage [1] [9].

As none of the existing mechanisms effectively addresses the issue of returning to Web sites and pages from the long term history, there is an opportunity to provide a feature specifically designed to support access to them.

2.1 Observations and problem definition

User studies presented in [10] show that users perform a variety of activities on the Web. Each such activity may result in a single or several navigation paths, as evident from the users' navigation logs. A starting page in such a path is particularly important as it represents the onset of an activity. The user may access a number of sites and thus follow several paths within some time period. These Web activities may or may not be related. However, if they often co-occur within the same time span, they represent a user's routine for which we could provide support.

These observations motivate a model of the navigation history that comprises *navigation sessions* and *web trails*. These concepts introduce sub-structures in the navigation history induced by the user's activities and routines. They have been exploited in [10] to provide support for back navigation to hubs, starts of trails, and similar. In this paper we show how they are used to predict revisits to sites and pages related to the recurring user's activities.

Navigation Session. We consider navigation sessions as periods of the user's navigation activity that are separated from each other by noticeable periods of the user's inactivity. This definition provides flexibility for looking at the navigation history at different granularity levels, by increasing or reducing the required idle time for session boundary. It also enable us to separate online navigation from application window management, if desirable.

Web Trail. As in [10] [11] [17], a notion of a trail designates a sequence of navigation steps that comprise link executions and Back and Forward navigations. In our definition, the trail is initiated by typing a URL into the address bar, or by selecting a URL from bookmarks or another document, such as an e-mail message containing a link. Beginnings of trails will therefore be considered as the starts of the user's activities (Figure 1).

Previous analyses of navigation histories are focused on estimating the likelihood of predicting the next revisited URL [17]. Here we define our objectives to:

- Predict the next recurring user activity by predicting the start of the corresponding trail(s).
- Predict revisit of pages related to that activity may be revisited
- Capture the sets of co-occurring trails and pages within the same navigation session.

Essentially, we analyze the patterns of revisitations at two levels: the level of trails and the level of the page visits. We also take into account the typical application constraints and requirements:

- Algorithms need to give sensible and comprehensible predictions from start, when no history is available. Otherwise the feature will not be adopted by users.
- User logging may not be systematic and continuous since the user may change computers, upgrade applications, and similar.
- Extent of logged history may vary and change, e.g., due to the history management by the operating system and user's Web access habits.

Before discussing our algorithms, here we provide a brief overview of predictive methods that have been used in various Web related areas.

2.2 Predictions on the Web

Discovery of patterns and predictions have been researched in the context of various Web application areas, including Web server performance management, personalization of Web user experience, Web site design and modification, Web assistants for in information access, etc. Srivastava et al. [15] offer a detailed

overview of research areas, projects, and commercial services that involve Web usage mining. While they all address significantly different problems from the one we defined, there is a commonality in the applied techniques and we wish to comment on those further.

2.2.1 Application areas

Optimization of Web server performance is essential with the growing number of users on the Web. Web usage logs are therefore mined to identify traffic patterns [12] and decide on developing policies for Web caching, pre-sending, load management, and site redesign. Records of end user accesses to the site, without client tracking, are not precise. Typically, one is restricted by knowing only the IP addresses and server-side click-stream. Thus, an individual user's navigation activities are dispersed across different Web servers. However, usage modeling of a population of users and prediction of the future page requests are still possible and very useful.

Related to this are Web proxy servers which perform the function of intermediate caching between the clients and the Web servers. They facilitate communication between multiple clients and multiple Web servers. Their performance depends on their ability to predict the future page requests. However, in contrast to client based logging that captures the individual user's patterns, proxy traces contain cross user data.

Yet another related application area is proactive recommendation of URLs by online services in order to provide personalized Web experience or assistance. Search engines, shopping sites, and similar online services make dynamic recommendations based on the user profiles and usage behaviour. Information agents such as Letizia [8], WebWatcher [7], and SiteHelper [13] exploit individual users' navigation patterns and contents of viewed pages to create user profiles and proactively look for pages that might be relevant to the user.

2.2.2 Methods and algorithms

Modeling of usage and user interest involves a spectrum of techniques. We are particularly interested in those that capture relationships or groupings of items and discover sequential patterns. Clustering of user pages and activities is a common way of classifying users and personalizing the content or recommendations to be delivered to them [12]. A technique widely used throughout a range of applications is mining of association rules. This method provides a prediction of whether a particular page or item will be requested if a set of other items have been already accessed within a session. Since most of the applications involve server side logging and large data sets, research has been concerned with performance and efficiency issues [2]. Accordingly, the performance tuning and evaluation have been based on the assumption that the large logs of navigation are available. In many applications it is beneficial to preserve information about the order in which the items are observed. In particular, sequences of user navigation in proxy traces or traversals of paths in web sites, stored by web servers, can inform on the cache management and web site design, respectively. Thus the algorithms for efficient mining of sequential patterns have been devised and evaluated as in [6] [11] [3]. Both of these will be considered in our algorithm design and evaluation.

2.3 Summary

Most of the current work on predicting user activities has focused on mining Web logs on Web servers and thus dealing with large datasets. In our task we perform client based logging and deal with much smaller data sets. Therefore, we cannot assume that the standard Web mining methods will perform well on our problem.

Furthermore, our 'recommendations' to the user are from the set of URLs that the user has seen in the past. The history of navigation is gradually growing and the system is continually learning about the user. Most of the existing recommender systems immerse the user in the large populations of users, characterized by statistics that are much less volatile.

We provide generic support for navigation and therefore are not concerned with the typology of users' activities beyond the characteristics that are observed in the navigation patterns. We use logs of user interaction with the browser with minimal information about the page content: only the identifier, i.e., URL of a page.

In relation to the research in [17], our aim is complementary to the design of recent navigation history. We model repetitive user activities that are observed across the longer term history and not the revisits in short term history.

3. SMART FAVOURITES DESIGN

3.1 Objectives

The main objective of SmartFavourites is to proactively expose two types of links: (1) a URL that the user needs to access in order to start the next Web activity and (2) URLs that the user has visited in the past while performing the current activity. Thus, both types of links are aimed at pages from the more distant past, depending on how frequently the activity is performed.

We have chosen to expose the predictions in the form of a link bar that may contain up to 3 links of each type. Thus the algorithms evaluation will be based on whether the recommended 3 links contain the next activity and the next visit, respectively.

Furthermore, predictions of the Back navigation, i.e., the visits to previously seen page will not be counted as a success.

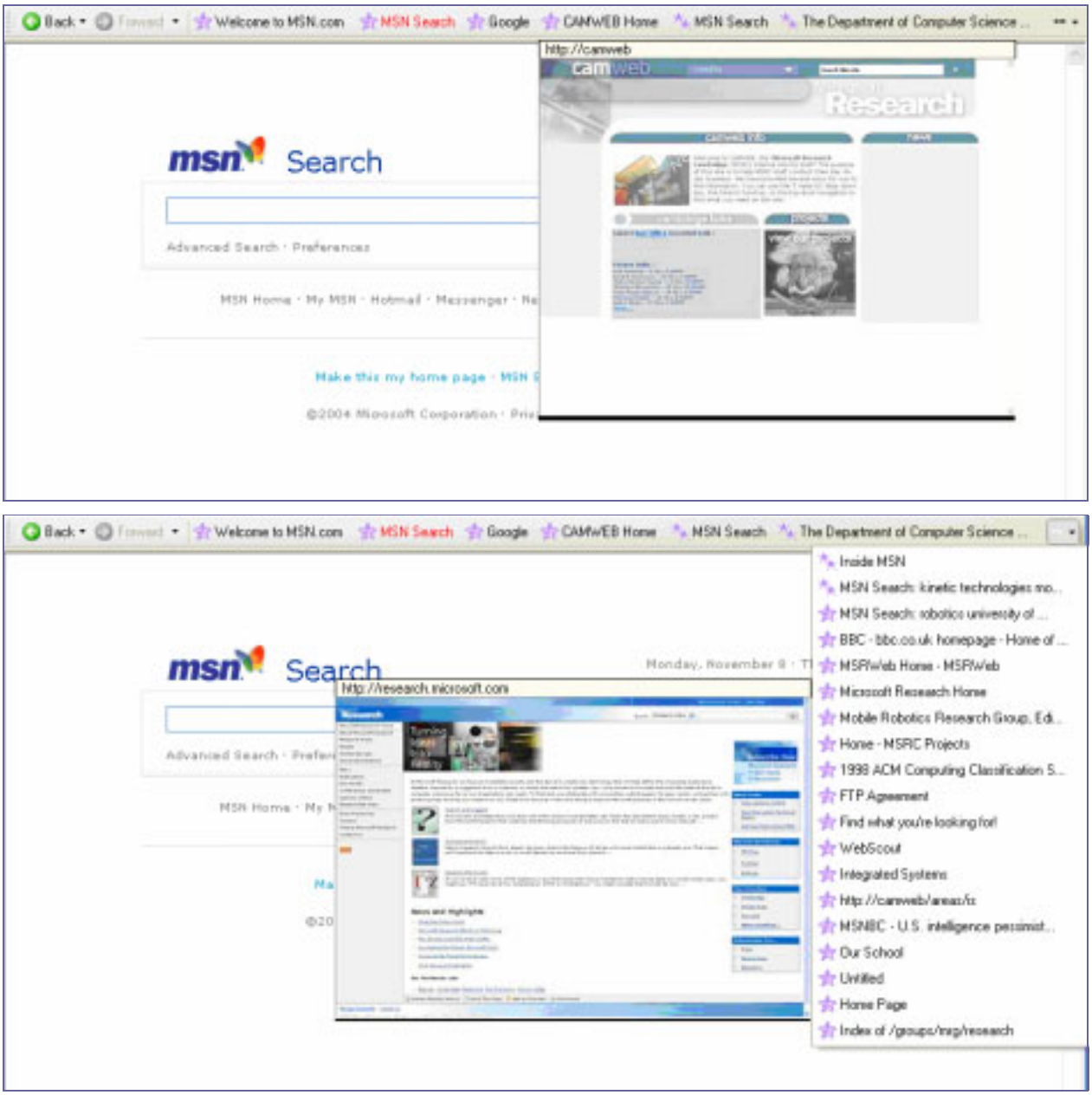

Figure 2: SmartFavourites LinkBar. Top: Toolbar with a thumbnail image of the 4th link on the link bar. Bottom: SmartFavourites Overflow menu showing selection of start-of-trail (large purple stars) and page predictions (pair of small purple stars), with the thumbnail image of the 6th page in the list.

3.2 User Interface Design

For illustration and clarity we provide snapshots of the currently implemented interface for SmartFavourites.

The SmartFavourites link bar includes links that correspond to starts-of-trails, designated by large purple star icons. Links marked by a pair of small stars are computed relative to the currently viewed start-of-trails. More precisely, these are pages that have been in the past visited in the navigation trails with the same start-of-trail and thus related to that particular user activity. In addition, we provide an extension of the SmartFavourites link bar in the form of Overflow menu, which presents additional recommendations (Figure 2).

The number of recommendations visible in the link bar varies with the size of the browser window. Furthermore, the set of presented links may not always contain the small double star items. These links are context sensitive and do not exist if the user is performing a new activity. Links that represent starts-of-trails are always present, exposing URLs of the next possible user activity. Since page titles are not always informative, we provide for each link a thumbnail of the page as seen during the last page visit.

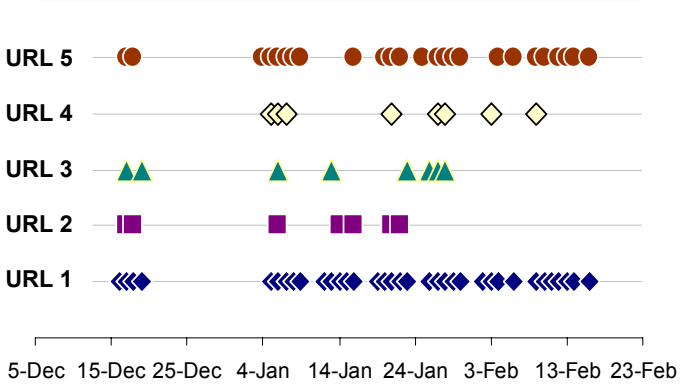

Figure 3. Co-occurrence pattern of starts-of-trails from one of the user's logs. Each horizontal line corresponds to the timeline for a particular URL. Each icon on the line represents the occurrence of a particular URL.

3.3 Algorithms

In this section we present in detail a set of algorithms that we explored in relation to the SmartFavourites design. Based on the literature about Web usage and our own user research we designed and applied two algorithms for predicting starts-of-trails and two for determining relevant pages within the trails.

While the user interface lends itself to various strategies for optimizing SmartFavourites, e.g., through mixing and exposing various types of predictions, in this paper we focus on the evaluation of individual algorithms, each with regards to its own specific objective.

3.3.1 Predicting starts-of-trails

Keeping in mind that trails are essentially an approximation of the user's activities, our goal is to identify periodic occurrences of starts-of-trails and apply scoring schemes that give more weight to those that:

- Occur more frequently throughout the history, i.e., across navigation sessions
- Occur more recently in the long term history
- Occur more frequently within individual navigation sessions.

Thus, we use cumulative frequency of the start-of-trails across sessions, which automatically penalizes absence of the URL in intervening navigation sessions. However, we also introduce a parameter that enables us to increase or to reduce explicitly the bias towards recent trails in the form of an exponential time decay factor. The time decay is applied over navigation sessions. Furthermore, the frequency of a start-of-trail occurrence within a session is normalized to prevent dominance of those starts-of-trails that may occur frequently within a single session. More precisely, the algorithm, here referred as [ST-Fr], involves the following steps:

Algorithm [ST-Fr]

For each URL that corresponds to a start-of-trail in a navigation session n,

1. Determine the frequency r_n of the URL within the navigation session n
2. Calculate the normalized frequency score f_n from r_n based on the transformation

   $$f_n = 2 \times r_n / (r_n + 1)$$

   which maps the raw frequencies r_n onto the interval [1,2).

   This transformation is informed by the frequency normalization approaches in information retrieval. We examine its effect in the experiments presented in the following sections.
3. Add the frequency score f_n to the cumulative frequency from the previous session S_{n-1}.

   $$S_n = \alpha \times S_{n-1} + f_n$$

   where α is a *time decay factor* with the value in the interval [0,1]. The value $\alpha=1$ means that no time decay is applied.

   Thus, S_n is the accumulated frequency score for a particular start of trail after n navigation session and incorporates exponential time decay at a specified rate α:

   $$S_n = \alpha^{n-1} \times f_1 + \alpha^{n-2} \times f_2 + \alpha^{n-3} \times f_3 + \ldots + \alpha \times f_{n-1} + f_n.$$

Context sensitive exposure of stars- of-trails

Analyses of users' logs also reveal that, in some cases, the user establishes a routine for accessing the Web. That routine may involve a number of activities, repeated across navigation sessions in a more or less similar manner. For example, the user may spend lunch times reading news, checking personal e-mail accounts, and playing a couple of on-line chess games.

This calls for an algorithm that identifies groups of activities within navigation sessions. Based on such groups we can expose starts-of-trails in a context sensitive manner: if one of the start-of-trails is detected, SmartFavourites can present the others from the group. To this end, we designed an algorithm that traces the co-occurrences of starts-of-trails and let us specify how the order of URL access, proximity, and co-occurrence frequency are used:

Algorithm [ST-Co]

For a given navigation session n, identify the time based sequence of starts-of-trails as they occur in the session (see Figure 3 for a sample pattern of co-occurrences).

1. For each pair of URLs (a,b) in the navigation session n, identify the frequency of "a occurring before b" (a « b) within the distance d, i.e., separated by d starts-of-trails: $c_{n,d}(a \ll b)$
2. Specify a weighting scheme to reward the proximity of URLs: define the value of the maximum weight M to be associated with the adjacency of two starts-of-trails (at distance $d=0$). For those with distance d, the weight factor is determined as $w_d=M+1-d$, where $d=1,\ldots,M$.
3. Calculate the total co-occurrence score for a « b the session n:

   $$c_n(a \ll b) = \sum_d c_{n,d}(a \ll b) \times w_d.$$

4. Accumulate the co-occurrence score across sessions, applying the decay factor α:

   $$C_n(a \ll b) = \alpha \times C_{n-1}(a \ll b) + c_n, \text{ with } 0 \le \alpha \le 1.$$

5. If a is visited during session $n+1$, look up the ranked list $C_n(a \ll x)$ and select top scored URLs x.

Essentially, we collect co-occurrence statistics for a trail a and M subsequent trails. We retain information about the order by storing separately the co-occurrence scores for a « x and x « a and

use them in our experiments. It is possible to combine the two scores in case the exact order in which the user performs activities is not considered relevant.

3.3.2 Predicting pages associated with trails

Once the SmartFavourites detects that the user has started an activity it can expose URLs associated with that activity in the past. For example, the user may typically visit a news site and from there navigate to a *comic* page and *science and technology* updates. We can expose these two URLs in the link bar and thus provide quick access.

This is accomplished by two page prediction algorithms:

Algorithm [PP-Co]

Apply [ST-Co] to the URLs in the individual web trails to capture co-occurrence of pages.

Note, in this instance one can apply time decay on the level of trails or sessions.

Algorithm [PP-Seq]

1. For each URL in a trail, keep a count of navigation sequences of length *s* that start with that URL.
2. Accumulate the frequencies across trails by apply the time decay factor over navigation trails.
3. If the user accesses a URL *a*, determine the most frequent sequence of length s for *a*. Present to the user the last page in that sequence most frequent sequence.

The [PP-Seq] algorithm is motivated by the analysis of user logs collected in the study of 9 participants over a period of two weeks [10]. By applying the pattern detection module algorithm (PDM) to identify the longest repeated subsequences in URL visits [6], we observed that the users traversed the total of 107 sequences of 3 or more, associated with 30 distinct URL sequences. 68 (63%) of these repetitions were associated with the sequences of length 3. Based on this information we decided to implement the algorithm [PP-Seq] for the fixed size sequences of 3 visits. For any URL that the user revisits, we can present the pages that have been at least one step removed in the previous sequences and thus save on forward navigation.

3.3.3 On alternative algorithms

The notion of context sensitive recommendation of items fits well with the objective and design of the algorithms for the discovery of association rules. Association rules have a wide range of applications and have been customized and optimized to fit different application scenarios.

The basic algorithm starts by considering the data as a collection of sessions containing items. It processes the data across sessions to identify 'inference' rules of the form: if a session contains an item set $\mathcal{X}$ then, with the probability *p(i)* it will also contain the item *i*. It allows us to specify the *minimum support* for the rules, i.e., the minimum number of sessions that should contain all the items involved in the rule.

We include the association rules algorithm in our comparative experiments.

4. ALGORITHM EVALUATION

Our objective is to assess how the algorithms defined in the Section 3 perform on individual tasks they are set to accomplish: to predict the next activity (i.e., revisit to start-of-trail) and to predict the page that the user may revisit next, respectively.

4.1 Description of Data

In our experiments we use navigation logs from 30 individuals who participated in our three recent observation studies. While some of these logs may have resulted from non-standard browsers, enhanced by new and experimental features, we assume that the influence of these features on what the person intended to revisit is minimal. Thus, we treat the logs as a sequence of navigations that our algorithms should aim to predict, within the scope of their objectives, i.e., the prediction of revisits.

Table 1 presents statistical information about data logs, including the time period over which the data is collected (Days), number of navigation sessions and web trails, and number of unique URLs and those revisited more than once.

Table1. Statistics about the data sets used in the experiments: User Id, number of sessions, trails, URLs, URL visits, URLs visited more than once, and the number of days.

User	No. Ses.	No. Tr.	URLs	Visits	Vis>1	Days
1	1208	2480	3795	8933	1416	94
2	952	1461	1792	4374	627	74
3	631	1036	1700	3307	559	40
4	429	1046	1334	3070	391	27
5	372	830	1237	1951	325	29
6	432	760	1083	2074	329	24
7	141	355	1010	1727	215	17
8	313	678	923	1674	281	35
9	169	309	893	1920	265	10
10	219	542	758	1649	251	37
11	164	344	744	1314	210	18
12	319	518	742	1326	161	28
13	131	422	700	1536	139	20
14	156	375	535	1164	198	31
15	132	285	450	774	133	10
16	187	261	378	685	106	18
17	55	132	330	853	107	9
18	89	152	236	401	53	17
19	108	126	219	623	87	14
20	95	146	192	434	66	18
21	69	86	170	474	72	8
22	50	102	151	367	63	6
23	28	54	149	303	56	6
24	47	57	125	245	32	10
25	23	45	111	177	25	8
26	25	35	109	293	46	6
27	45	51	107	247	49	10
28	28	39	82	197	39	11
29	23	31	28	55	9	8
30	6	6	3	6	1	3

4.2 Experiment Design

4.2.1 Evaluation Measures

At each point in time, algorithms learn from the navigational history that is available up to that point, with no special provisions taken for the initial condition, when the history log is empty. Each prediction algorithm presents up to 3 links – this is in accordance with our interface design (see Section 3.2). We are interested in the average recall of revisited links for each of the algorithms individually. Essentially, at each step we check whether the user has revisited a page from the past, and if so whether the link was among the three that we predicted at that step. We obtain the success rate over the history log for each user and then average over the user sample.

Specifically, for the start-of-trail algorithms ST-Fr and ST-Co, we include in the calculations all the trails up to the given point in time. If the next revisited start-of-trail is included in our predictions, we mark it as a success and calculate the success rate as a percentage of all revisited starts-of-trails that we predicted. We average the success rate across the users. Note, we do not penalize the system for not predicting new starts-of-trails as that is out of the domain of the algorithm.

The page predictors, PP-Co and PP-Seq are evaluated slightly differently. The objective of page predictors is not necessarily to predict the next URL request by the user but to provide shortcuts to pages that may not be accessible from the current page by the link navigation or by the standard Back and Forward buttons. In the user logs that have been obtained from users of the standard browsers, these pages will appear further in the navigation trails. For example, if the user repeats a sequence of URLs $a \rightarrow b \rightarrow c$ with the intention to access c, and if our objective is to present a link c to the user, we need to verify the validity of our prediction by considering the pages following the page b in the log. Since we are exploring co-occurrences of various types, the only practical way to evaluate the algorithms is based on the overlap of predictions with the remainder of the navigation trail.

More precisely, for the predictions at the visit to URL a in the trail t we will check whether any URL from the remainder of the trail is included in the predictions. If so, we give a credit of 1 to the predictor. The total score is averaged over all the predictions for which there was at least one revisit present in the remainder of the trail that the predictor could predict. We provide the average score for PP-Co and PP-Seq when the predictors can display one and three choices to the user.

4.2.2 Evaluation Set-up

We design experiments to investigate several parameters associated with the algorithm. We need to investigate the impact of the time decay α and the frequency at which we update the statistics used in the algorithms. Indeed, the statistics that is used for predictions could be updated at different rate: after each navigation session, after each end-of-trail, and after each end-of-session. The latter two are of relevance to starts-of-trails; all three may apply to the page predictions.

However, probably the most important is a comparison with a reasonable baseline for each of the prediction tasks. These are discussed in details for each experiment. All comparative results are checked for statistical significance.

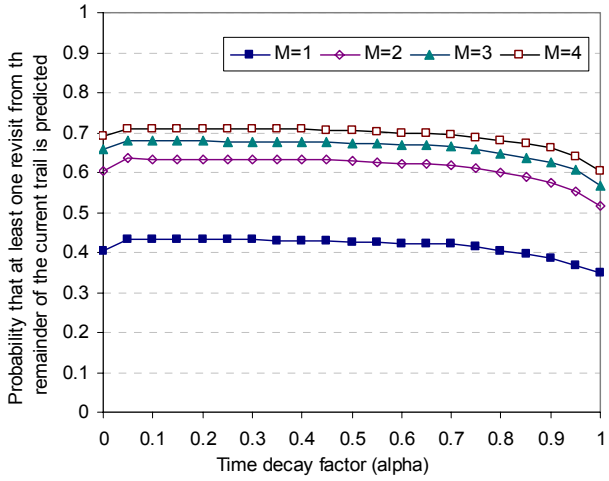

Figure 4. Average performance of the page predictor PP-Co over the set of users, showing the probability that the algorithm predicts at least one page from the remainder of the current trail, if the remainder contains a page that we can predict (i.e., contains at least one page seen by the user in the past). Parameter M refers to the number of subsequent trails used for collecting co-occurrence statistics. (Section 3.3.1).

4.3 Experiment Results

4.3.1 Influence of the Time Decay

In the experiments with the start-of-trail prediction, the time decay of frequency scores did not produce statistically significant difference in performance. Thus, setting the time decay factor to $\alpha = 1$ produces the best results both with the St-F and the St-Co algorithm.

In the page prediction experiments, PP-Co and PP-Freq, time decay is more helpful. The difference between the best performance, achieved for $\alpha < 0.5$, and the performance for $\alpha = 1$ is statistically significant. Equally good results are achieved at a wide range of α with the best values of α around 0.2, for both PP-Co and PP-Seq. We speculate that a more significant impact of α is observed for pages simply because we are dealing with longer 'sequences' of data then in the case of trails.

4.3.2 Prediction of Starts-of-Trails

Table 2 presents results from the experiments with the starts-of-trails algorithms ST-Fr and ST-Co. For comparison we define a baseline algorithm which is a variation on the standard random selection approach. Initially we intended to use as the baseline a random selection of 3 URLs from recently visited starts-of-trails. However, it turned out that by far the best results are achieved when the three most recently starts-of-trails are selected. Therefore, we used this more effective algorithm as the baseline for comparison with our algorithms ST-Fr and ST-Co.

Both ST-Fr and ST-Co have the frequency component (i.e., frequency scores and co-occurrence scores, respectively) which can be updated at the end of the current navigation session, or after each trail is completed. In the latter case, the scores associated with the trails from the current session are used straight away for the predictions. The results in the Table 2 show that continuous frequency score updates during the navigation session yield a higher performance across the algorithms and the resulting

difference is statistically significant. This is not surprising, since our exploration of the baseline algorithm has already indicated that users are most likely to repeat one of the three last activities.

Table 2. Success rate of different algorithms on the start-of-trail recommendation problem.

Algorithm	Update weights at the end of each session	Also update weights before the current trail
ST-Fr with raw frequencies (r_n)	0.5995 ($\alpha = 0.9$)	0.5247 ($\alpha = 1$)
ST-Fr with normalized frequencies (f_n)	0.6015 ($\alpha = 0.75$)	0.5320 ($\alpha = 0.95$)
ST-Co	0.5888 (M = 3, $\alpha = 0.9$)	0.5233 (M = 3, $\alpha = 1$)
Baseline	0.5010	

The statistics presented in Table 2 represent the highest performance achieved by each method over the range of time decay values, when 3 predictions are made by the system. While it seems that ST-Fr performs slightly better than ST-Co but the difference is not statistically significant. Both methods perform significantly better than the baseline method.

We also observe that ST-Fr does not benefit from normalization of frequencies. This is expected as most of the raw frequency counts are simply 0 or 1, because the navigation session tend to be relatively short and rarely contain multiple trails with the same start URL.

Predictions using Association Rules

We compare our ST-Co and PP-Co algorithms with the standard method for discovering and applying association rules. Each navigation session comprises a set of start-of-trail URLs. All the sessions up to the current one are used to train a set of association rules of the form "if this set of starts-of-trails appears in a session, then URLs are also likely to appear within the same session". We use these rules within the current session to anticipate what start-of-trail URLs are likely to appear in the remainder of this session.

The experiments with association rules show that, in over 90% of cases, none of the rules have a left-hand side that matches the current session. Thus, no link predictions can be obtained based on this approach. Even setting the threshold for the required minimum support at 2 sessions only does not make a difference in improving the generated rules. Therefore, we conclude that, while association rules may be helpful for analysis of large-scale web usage data, the logs of individual users do not contain enough repetitive patterns to yield useful rules.

4.3.3 Prediction of Pages

Table 4 shows results of predicting the pages using PP-Co and PP-Seq algorithms. As described in Section 4.2, the evaluation measure used here is the average number of recommended URLs that actually appear in the remainder of the current trail. Since Back navigation is a prevalent means of page revisitation, and easily accessed using Back button, we do not want to predict or get credit for predicting a page that is accessible via Back navigation. Thus, we remove such URLs from the remainder of the trail and modify the prediction algorithms to avoid recommending it.

Table 3. Statistics shows the percentage of times the system predicted a page from the remainder of the trail, if at least one repeated visit in the remainder of the trail existed. We show the performance in case when the system can present k=3 choices and only k=1 choice to the user.

Algorithm	Update weights at the end of each trail	Also update weights before the current page
PP-Co (k=3)	0.6057 (M = 4, $\alpha = 0.45$)	0.7103 (M = 4, $\alpha = 0.25$)
PP-Seq (k=3)	0.3041 ($\alpha = 0.9$)	0.4116 ($\alpha = 0.05$)
Baseline (k=3)	0.4570	
PP-Co (k=1)	0.4908 (M = 4, $\alpha = 0.2$)	0.5419 (M = 4, $\alpha = 0.4$)
PP-Seq (k=1)	0.1849 ($\alpha = 1$)	0.2501 ($\alpha = 0.05$)
Baseline (k=1)	0.2204	

The baseline algorithm considered here is analogous to the one in the previous section: it recommends the last three visited URLs, after the immediately preceding page is removed. We investigated predictions based on random selections of pages nd found that this is by far the best performance. Thus we use it as our baseline.

Similarly to ST-Fr and ST-Co in the previous section, we evaluate algorithms PP-Co and PP-Seq by updating frequency scores of individual pages in two ways: (1) we wait for the entire trail to be completed and then update the scores of individual pages, and (2) we update scores after each individual page is viewed. In the latter case, the statistics of pages that have been seen during the current trail are available for calculating predictions. More frequent updates again produce statistically significant performance improvement, as shown in Table 3.

The results presented in Table 3 are the best average prediction performances (averaged over users) obtained when different time decay values α are applied. As we can see, the maximum is typically achieved around $\alpha = 0.2$, which means that only 20% of the previous cumulative score is used when the scores are updated at the end of the current trail. Thus, to achieve the best performance with the current PP-Co algorithm it is important to apply aggressive time decay. This implies that the co-occurrence statistics with pages from the previous trails does not play a significant role in supporting users in page revisitation. In other words, people may revisit sites periodically, as captured by the starts-of-trails predictions, but may not necessarily visit the same URLs. This seems plausible considering Web activities such as online shopping, news reading, and similar.

The PP-Seq algorithm, i.e., the prediction of pages based on the fixed 3-step navigation sequences alone performs poorly. This is expected as such sequences are very rare compared to other types of revisitations.

Predictions using Association Rules

As in the case of starts-of-trail predictions, we tried to use association rules to predict revisitation of pages. We treat each trail as a collection of items visited during the trail. Again, the resulting sets of rules failed to recommend any pages in over 90 % of times. It might be possible to address this problem by allowing rules with a minimal support of 1 rather than 2 or more.

5. SUMMARY

In this paper we demonstrated the use of structured history model in analyzing the patterns of Web revisitation. We assume that the patterns in revisitation of Web trails are the reflection of the patterns in the user's online activities. As we are concerned with the generic support for navigation, the typology of online activities is not considered but rather their manifestation in the navigation pattern. We demonstrate that our algorithms, which take into account statistics about Web trails and constituent pages in the navigation sessions outperform simpler approaches that do not utilize the structure. We also demonstrate that the standard association rules method does not produce useful predictions on the relatively small data sets that we are working with. On the other hand, we provide evidence that incorporating time decay does provide statistically significant improvement of the algorithms for prediction of pages.

One natural extension of the presented work is to introduce more detailed typing of the visits and pages in the navigation history and use them in the prediction algorithms. For example, we can add information about navigational hubs, i.e., branching points in Web navigation, or specific types of pages, such as results of form submissions, such as logins, online searching, and similar.

The second important aspect is to investigate how the presence and usage of recommendations affect the user behaviour and therefore the future predictions of the algorithm. For example, if the user clicks on a page level recommendation (e.g., results of PP-Co) it might be appropriate to consider that page as a start of a new trail since sometimes users do navigate to a page from where they can start an activity (e.g., navigating from a home page of bank to a login page for account management). However, it is important to understand implications that would have on the algorithms.

Finally, it is important to evaluate the ways of combining Web trail and page level predictions to support users in their online activities, in their natural setting. In [18] we report on an initial study that focused on the usefulness and appeal of the initial SmartFavourites prototype. Detailed discussion of this study is beyond the scope of this paper. Direct comparisons with the experimental evaluation is not possible based on that data and therefore it has not been presented in any detail here.

6. REFERENCES

[1] Abrams, D., Baecker, R. and Chignell, M. Information Archiving with Bookmarks: Personal Web Space Construction and Organization. In Proceedings of CHI'98 (1998), 41-48.

[2] Agrawal, R. & Srikant, R. Fast Algorithms for Mining Association Rules. In Proceedings of 20th International Conference on Very Large Data Bases (1994).

[3] Agrawal, R. and Srikant, R. Mining Sequential Patterns. In Proceedings of the 11th International Conference on Data Engineering (1995).

[4] Catledge, L., and Pitkow, J. Characterizing browsing strategies in the World Wide Web. Computer Networks and ISDN Systems 27(6) (1995), 1065-1073.

[5] Cockburn, A. and McKenzie, B. What Do Web Users Do? An Empirical Analysis of Web Use. International Journal of Human-Computer Studies 54 (2001), 903-922.

[6] Craw, D. & Smith, B. DB_Habits: comparing minimal knowledge and knowledge-based approaches to pattern recognition in the domain of user-computer interactions. In R. Beale & J. Finlay, Eds. Neural Networks and Pattern Recognition in Human-Computer Interaction. Chichester: Ellis Horwood (1992), 33-61.

[7] Joachims, T., Freitag, D., and Mitchell, T. WebWatcher: A tour guide for the world wide web. In Proceedings of the 15th International Conference on Artificial Intelligence (1997).

[8] Lieberman, H. Letizia: An Agent that Assists Web Browsing. Proceedings o the International Joint Conference on Artificial Intelligence (1995).

[9] Maarek, Y. and Ben Shaul, I. Automatically Organizing Bookmarks per Contents. In Proceedings of the Fifth International World Wide Web Conference (1996).

[10] Milic-Frayling, N., Jones, R., Rodden, K., Smyth, G., Blackwell, A. and Sommerer, R. SmartBack: Supporting Users in Back Navigation. In Proceedings of the Thirteenth World Wide Web Conference (2004).

[11] Milic-Frayling, N., Sommerer, R., and Rodden, K. WebScout: Support for revisitation of Web pages within the navigation session. In the Proceedings of IEEE/WIC International Conference on Web Intelligence (2003), 689-693.

[12] Mobasher, B. & Cooley, R. Automatic Personalization Based on Web Usage Mining, in Communications of the ACM 43(8) (2000), 142-151.

[13] Ngu, D.S.W. & Wu, X. SiteHelper: A Localized Agent that Helps Incremental Exploration of the World Wide Web. In Proceedings of the 6th International World Wide Web Conference (1997).

[14] Pitkow, J. and Pirolli, P. Mining Longest Repeating Subsequences to Predict World Wide Web Surfing. In Proceedings of USITS'99: The 2nd USENIX Symposium on Internet Technologies & Systems (1999).

[15] Srivastava, J., Cooley, R., Deshpande, M., and Tan, P-N. Web Usage Mining: Discovery and Application of Usage Patterns from Web Data. In SIGKDD Explorations (2000).

[16] Srikant, R. & Agrawal, R. Mining Generalized Association Rules. In Proceeding of the 21st International Conference on Very Large Data Bases, Zurich, Switzerland (1995).

[17] Tauscher, L. and Greenberg, S. How people revisit web pages: empirical findings and implications for the design of history systems. International Journal of Human Computer Studies 47(1) (1997), 97-137.

[18] Milic-Frayling, N., Jones, R., Rodden, K., Smyth, G. and Frayling, A. Designing for Web Revisitation: Exploiting Structure from User Interaction and Navigation. Microsoft Technical Report MSR-TR-2004-97. September 2004