A Prediction Model for User Access Sequences

Enrique Frias-Martinez
Computer Science Department
New York University
715 Broadway, Room 719, NYC, NY, 10003, USA
Tel: +1 212 998 3493
frias@cs.nyu.edu

Vijay Karamcheti
Computer Science Department
New York University
715 Broadway, Room 704, NYC, NY, 10003, USA
Tel: +1 212 998 3496
vijayk@cs.nyu.edu

ABSTRACT

One of the important Internet challenges in coming years will be the introduction of intelligent services and a more personalized environment for users. Analysis of Web server logs has been used in recent years to model the behavior of web users in order to provide intelligent services. In this paper we propose a model for predicting sequences of user accesses which is distinguished by two elements: sequentiality and personalization.

The concept of sequentiality in our model possesses three characteristics: (1) preservation of the sequence of the click stream in the antecedent, (2) preservation of the sequence of the click stream in the consequent and (3) a measure of the time gap between the antecedent and the consequent using the number of user clicks. In order to improve its prediction ratio, the model includes a personalization scheme in which each frequent user of a web site has a personal prediction system.

The model has been defined as a black box that can be used as part of any intelligent service. As an example, we present a cache prefetching system based on the prediction model. The hit ratio of the cache is highly satisfactory.

Keywords: Web Usage Mining, Association Rules, Sequential and Temporal Models, Prefetching, Sequence Prediction.

1. INTRODUCTION

Since Etzioni [12] first proposed the term "Web Mining" a lot of research has been done in this area. One of the topics that has received a lot of attention is modeling web users's behavior, in other words, being able to predict the requests of that user. Having a model of user behavior has allowed the implementation of a great variety of intelligent services. Some examples of these services include redesigning of web sites [24], personalization for e-commerce sites [27], recommendation of pages [18], construction of web pages in real-time [23], adaptation of web pages for wireless devices [4], improvement of web search engines, and prefetching [11][19][30][31].

The main techniques traditionally used for modeling user's patterns are clustering and association rules. These two approaches produce systems which lack two important characteristics of Web user access: sequentiality and temporality. In this context sequentiality implies reflecting the order of the requests of the user, and temporality refers to being able to capture when the predicted actions are actually going to happen.

In this paper we present a model that constructs sequential association rules to capture the sequentiality and temporality in which web pages are visited.

In order to do this, the rules constructed by the model preserve the sequence of the click stream of the antecedent and the sequence of the click stream of the consequent. There is a third significant element in our model: the rules also reflect the distance between the antecedent and the consequent measured by the number of user clicks to go from one to another.

The concept of distance between the antecedent and the consequent is very important for the prediction system because it allows the rules to express not only what pages are going to be accessed but also when precisely they are going to be accessed. This is especially useful for prefetching applications or for recommendation systems. Additionally, the concept of distance can be used as a measure of the quality of the rule. For example if we want to redesign Web pages for wireless devices by finding shortcuts, the distance of a rule can be used to measure how useful is that shortcut.

Our results show that this approach yields good prediction accuracy when working with sites that have a simple architecture. However when working with sites with complex architectures the results are not as good. This occurs because access order is not a global property shared across all users. In these cases, in order to improve the accuracy of the sequential prediction system, we introduce a personalized prediction system for the set of frequent users of the site. It is arguable that this will require a lot of space. However, as we will see, the increment per user compared to the information that sites already have about each user is not significant and is even less significant if we consider that user satisfaction is the most important criteria for an e-commerce site.

The proposed model can be used as a black box for any of the applications previously mentioned. To illustrate such use, the paper presents a cache prefetching application of the Sequential Behavior Model. This example is presented from a new point of view: we are especially interested in studying how the accuracy of prediction affects the final hit ratio. Our study separates out the above from other effects.

The organization of the paper is as follows. Section 2 summarizes the motivation and prior work done in this area. Section 3 presents the Sequential Behavior Model. Section 4 implements some examples of the Sequential Behavior Model and analyzes the results. Section 5 presents the Personalized Sequential Behavior Model and analyzes the results. Section 6 presents an example of the model: a prefetching system. Finally, in Section 7 we conclude the article with the conclusions and a discussion of future work.

2. MOTIVATION AND RELATED WORK

The main techniques used for pattern discovery are clustering and association rules [25].

Clustering, applied in the context of web mining, is a technique that makes it possible to group similar browsing patterns or to divide the web pages of a site into groups that are accessed together. This information can be used in the recommendation process of a page [18] or by search engines to display related pages along with their results. Also, in this context, clustering has a distinguishable characteristic: it is done with non-numerical data. This implies that, usually, the clustering techniques applied are relational. This means that we have numerical values representing the degrees to which two objects of the data set are related. Some examples of relational clustering applied to web mining are [18][14]. Some authors have also considered the inherent fuzziness of the data presented in the web mining problem and have developed relational fuzzy clustering algorithms [15].

Association rule discovery aims at discovering all frequent patterns among transactions. The problem was originally introduced by Agrawal et al. [1] and is based on detecting frequent itemsets in a market basket. In the context of web usage mining, association rules refer to sets of pages that are accessed together. Usually these rules should have a minimum support and confidence to be valid. The Apriori algorithm [1] is widely accepted to solve this problem. Association rules can be used to re-structure a web site [24], to find shortcuts, an application especially useful for wireless devices [4], or to prefetch web pages to the local cache of a user to reduce the final latency [11].

The data used to obtain frequent patterns in a Web mining problem has a very important characteristic: it is sequential. The user accesses a set of pages in a given order and it is very important to capture this order in the final model obtained. Unfortunately, the two previous methods lack any kind of representation of this order. Clustering identifies groups of pages that are accessed together without storing any information about the sequence. Association rules indicate groups that are presented together.

Some authors have already dealt with the problem of capturing sequentiality in association rules for web mining. The approach taken in [9] considers sequences of each session to produce rules. [21] presents the PPM algorithm, which also preserves the order of access and basically uses a Markov prediction model. The main limitation of most of these approaches is that those algorithms only detect patterns that correspond to consecutive sequences.

In this paper we present a model that is able to detect patterns produced by non consecutive sequences and to preserve the order in which those web pages are visited. The model expresses those patterns using rules.

This ability of detecting patterns constructed with non consecutive sequences introduces the possibility of measuring the distance between the antecedent and the consequent of a rule. Some algorithms, like [19], are designed to detect non consecutive sequences, but there is no indication of the distance between them. In our model the distance between the antecedent and the consequent is measured in terms of the number of user clicks to go from one to the other. To date no model deals with the concept of distance between the antecedent and the consequent of a rule. This concept is very important for any application (such as a recommendation system) that attempts to infer characteristics of a web site because it provides information about when the pages are going to be visited. This concept is different from finding association rules that have explicit temporal information, as done in [3], or from looking for rules that give temporal relations between different sessions of the same user, as done in [17]. Our method gives a temporal relation within the same session between the antecedent and the consequent by measuring the distance between them.

The model also presents a local approach to the problem by introducing a personalized behavioral model for each frequent user of a web site.

3. SEQUENTIAL BEHAVIOR MODEL

This section presents the development of the proposed model. First we formalize the preparation of the data. Next the concept of Sequential Association Rule is introduced and then the definition of Sequential Behavior Model is given.

3.1 Preparing the Data

The syntax of the log file that contain all requests that a site has processed is specified in the CERN Common Log Format [7]. Basically an entry consists of (1) the user's IP address, (2) the access date and time, (3) the request method (GET, POST ...), (4) the URL of the page accessed, (5) the protocol (HTTP 1.0, HTTP 1.1,...), (6) the return code and (7) the number of bytes transmitted.

This set of logs contain enough information to reveal the set of sessions served by the site. The W3C Web Characterization Activity (WCA) [29] defines a user session as the click-stream of page views for a single user across the entire Web. In our context, the information we really want to obtain is a server session, defined as the click-stream in a user session for a particular Web server. A click-stream is defined as a sequential series of page view requests. Although the W3C defines a user as a single individual that is accessing one or more servers from a browser, in our problem we will identify an IP as an user, bearing in mind that a single IP can be used by a group users. Our system defines a compiler that transforms a set of logs L expressed as,

$$L = \left\{ L_1, ..., L_{|L|} \right\}$$

$$L_i = (IP_i, TIME_i, METHOD_i, URL_i, PROT_i, CODE_i, BYTES_i),$$
$$\forall i / i = 1...|L|$$

Into a set of sessions S,

$$S = \left\{ S_1, ..., S_{|S|} \right\}$$

Where $|L|$ is the number of logs of L and $|S|$ is the number of sessions of S. Each session of the set of sessions is defined by a tuple *(IP,PAGES)*, where *IP* identifies the user of the session and *PAGES* the set of pages requested,

$$S_i = (IP_i, PAGES_i)$$

$$PAGES_i = \left\{ url_{i,1}, ..., url_{i,p_i} \right\}, \forall i / i = 1...|S|$$

Where p_i is the number of pages requested by user IP_i in session S_i. The set of URLs that form a session satisfy the requirement that the time elapsed between two consecutive requests is smaller than a given Δt. The value we have used is 30 minutes, based on the results of [6] and [25]. Figure 1 presents the algorithm of the compiler.

```
Input : L, Δt, |L|
Output : S, |S|
function Compiler(L, Δt, |L|)
  for each L_i of L
    if METHOD_i is GET and URL_i is WEB_PAGE then
      if ∃ S_k ∈ OPEN SESSIONS with IP_k = IP_i then
        if (TIME_i − END_TIME(S_k)) < Δt then
          S_k = (IP_k, PAGES_k ∪ URL_i)
        else
          CLOSE SESSION (S_k)
          OPEN NEW SESSION(IP_i, (URL_i))
        end if
      else
        OPEN NEW SESSION(IP_i, (URL_i))
      end if
  end for
```

Figure 1. Algorithm of the compiler implemented

The filters implemented by our algorithm delete all entry logs that do not refer to a URL or that indicate an error. Also, sessions of length one or sessions three times as long as the average length of the set of sessions S are erased. This is done to eliminate the noise that random accesses or search engines would introduce to the model. Finally, the average length of the set of sessions S is defined as,

$$N = \frac{\sum_{i=1}^{|S|} p_i}{|S|}$$

3.2 Sequential Association Rules

The concept of Sequential Association Rule (SAR) is based on the concept of an N-Gram. In the context of web mining, an N-Gram of a session S_i is defined as any subset of N consecutive URLs of that session.

A Sequential Association Rule (SAR), given $|A|$ the length of the sequence of URLs of the antecedent, $|C|$ the length of the sequence of URLs of the consequent, and n the distance between the antecedent and the consequent, is defined as:

$$A \xrightarrow[n]{} C$$
$$A := url_j, ..., url_{|A|+j}$$
$$C := url_l, ..., url_{|C|+l}$$
$$l = |A| + j + n + 1, \quad n \in \mathbb{N}^+$$

A SAR expresses the following relation: if the last click stream of length $|A|$ of a user is A then in n clicks the set of URLs C will be requested.

Each SAR is constructed from an N-Gram obtained from a session. This means that for the SAR of the definition some session S_k of S contains an N-Gram, with $N=|A|+|C|+n$, that satisfies:

$$S_k = (..., url_{k,j}, ..., url_{k,j+|A|}, url_{k,j+|A|+1}, ..., url_{k,j+|A|+n}, url_{k,l}, ..., url_{k,l+|C|}, ...), \text{ with } l = j + |A| + n + 1$$

Each SAR has a degree of support and confidence associated with it. The support of a rule is defined as the fraction of strings in the set of sessions of S where the rule successfully applies. The support of a rule is given by:

$$\theta(A?_1...?_n\, C) = \frac{|S_i \in S / (A?_1...?_n\, C) \in S_i|}{|S|}$$

Where $?_1...?_n$ represents the set of any n pages in the session, and $A?_1...?_nC \in S_i$ is defined as $A?_1...?_nC$ is an N-Gram of S_i. This is the way to model the distance between the antecedent and the consequent when obtaining the support. The confidence of a rule is defined as the fraction of times for which if the antecedent A is satisfied, the consequent C is also true in n clicks. The confidence of the rule is defined as:

$$\sigma(A?_1...?_n\, C) = \frac{\theta(A?_1...?_n\, C)}{\theta(A)}$$

$SR_{|A|,n,|C|}$,for a given a set of sessions S, is defined as a set of tuples *(SAR,Counter)*, where each tuple has a SAR with an antecedent of length $|A|$, a consequent with length $|C|$ and a distance n between the antecedent and the consequent. The *Counter* of each tuple indicates the number of times that the correspondent rule occurs in S. Figure 2 shows the algorithm used to obtain $SR_{|A|,n,|C|}$.

```
Input : |A|, n, |C|, S
Output : SR_{|A|,n,|C|}
function Obtain_SR(|A|, n, |C|, S)
  SR_{|A|,n,|C|} = ∅
  for each S_i of S
    for k = 1 to p_i
      if k + |A| + n + |C| ≤ p_i
        SAR = url_{i,k}, ..., url_{i,k+|A|} →(n) url_{i,k+|A|+n}, ..., url_{i,k+|A|+n+|C|}
        if (SAR,?) ∈ SR_{|A|,n,|C|} then
          Counter((SAR,?)) = Counter((SAR,?)) + 1
        else
          SR_{|A|,n,|C|} = SR_{|A|,n,|C|} ∪ (SAR, 1)
        end if
      end if
    end for
```

Figure 2. Algorithm to obtain SR $_{|A|,n,|C|}$

In order to preserve only the relevant information, only those *SARs* which have support and confidence bigger than a given threshold are considered. With θ' the threshold of the support and σ' the threshold of the confidence, we will talk about the set of rules $SR_{|A|,n,|C|,\theta',\sigma'}$ as the rules of $SR_{|A|,n,|C|}$ with a support bigger that θ' and a confidence bigger than σ'.

$SR_{|A|,n,|C|,\theta',\sigma'}$ is defined as a set of elements $(SAR,\theta_{SAR},\sigma_{SAR})$, where SAR is a Sequential Association Rule, and θ_{SAR} and σ_{SAR} the support and confidence associated with it, satisfying $\theta_{SAR}>\theta'$ and $\sigma_{SAR}>\sigma'$. $SR_{|A|,n,|C|}$ contains enough information to obtain θ_{SAR} and σ_{SAR}. The set of $SARs$ of $SR_{|A|,n,|C|,\theta',\sigma'}$ is a subset of $SR_{|A|,n,|C|}$.

In order to optimize the storage and access to the rules that define $SR_{|A|,n,|C|,\theta',\sigma'}$, we group the rules with the same antecedent, storing for each set of rules the consequent and the degree of support and confidence associated with it. The structure of $SR_{|A|,n,|C|,\theta',\sigma'}$ is:

$$SR_{|A|,n,|C|,\theta',\sigma'}(A)=\begin{cases}((C_{1,1},\theta_{1,1},\sigma_{1,1}),...,(C_{1,l_1},\theta_{1,l_1},\sigma_{1,l_1})), if\ \mathrm{A}=A_1\\ \cdots\\ ((C_{k,1},\theta_{k,1},\sigma_{k,1}),...,(C_{k,l_k},\theta_{k,l_k},\sigma_{k,l_k})), if\ \mathrm{A}=A_k\\ \varnothing,\ \text{otherwise}\end{cases}$$

Where A is a click stream of length $|A|$, A_i, $i=1...k$, with $|A_i|=|A|$, is the set of antecedents of the rules of $SR_{|A|,n,|C|,\theta',\sigma'}$, $C_{i,j}$, $i=1...k$, $j=1...l_i$, with $|C_{i,j}|=|C|$, the set of consequents for each antecedent A_i, and l_i the number of consequents of each antecedent.

Figure 3 presents an example of the concepts previously introduced.

$$\begin{aligned}S=&\{(IP_1,\{url_1,url_2,url_4,url_5\})\\&(IP_2,\{url_1,url_3,url_2,url_4,url_5\}),\\&(IP_3,\{url_1,url_2,url_5\}),\\&(IP,\{url_1,url_2,url_5,url_4\})\}\\SR_{1,1,1}=&\{(url_1\rightarrow url_2,3),\\&(url_2\rightarrow url_4,2),\\&(url_4\rightarrow url_5,3),\\&(url_1\rightarrow url_3,1),\\&(url_3\rightarrow url_2,1),\\&(url_5\rightarrow url_4,1)\}\\\mathrm{SR}_{1,2,1}=&\{(url_1\rightarrow url_4,1),\\&(url_2\rightarrow url_5,2),\\&(url_1\rightarrow url_2,1),\\&(url_3\rightarrow url_4,1),\\&(url_1\rightarrow url_5,2),\\&(url_2\rightarrow url_4,1)\}\\SR_{2,1,1}=&\{(url_1,url_2\rightarrow url_4,1),(url_2,url_4\rightarrow url_5,1),\\&(url_1,url_3\rightarrow url_2,1),(url_3,url_2\rightarrow url_4,1),\\&(url_2,url_4\rightarrow url_5,2),(url_1,url_2\rightarrow url_5,1),\\&(url_2,url_5\rightarrow url_4,1)\}\end{aligned}$$

Figure 3. Example of application

3.3 Sequential Behavior Model

A Sequential Behavior Model is defined by a tuple $\{RU,\Phi\}$ where RU is a set of rules and Φ is the decision policy function. RU can be defined by any $SR_{|A|,n,|C|,\theta',\sigma'}$ or any subset or union of more than one SR. The function Φ is defined as:

$$\Phi((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i}))=\{C_{i,j},...,C_{i,b}\}$$
$$with\ 1\le i\le k,\ 1\le j\le l_i, 1\le b\le l_i$$

Where the independent variable of Φ is the set of consequents obtained from RU for a given antecedent (click stream), and the dependent variable is the consequent or set of consequents predicted. Some examples of policy functions include:

$$\Phi((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i}))=\{C_{i,j}\}$$
$$with\ \sigma_{i,j}\ge\sigma_{i,m}\forall m/m=1...l_i, m\ne j$$

$$\Phi((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i}))=\{C_{i,j}\}$$
$$with\ \theta_{i,j}\ge\theta_{i,m}\forall m/m=1...l_i, m\ne j$$

$$\Phi((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i}))=\{C_{i,j},C_{i,b}\}$$
$$with\ \sigma_{i,j}\ge\sigma_{i,b}\ge\sigma_{i,m}\forall m/m=1...l_i, m\ne j, m\ne b$$

The first example predicts the consequent with the biggest confidence, the second one predicts the one with the biggest support, and the third one picks the two consequents with the highest confidence. The policy function can be defined depending on the specific application and on the characteristics of the web log used.

The tuple (RU,Φ) defines the on-line execution of the prediction system. Given A the click stream of the last $|A|$ pages requested by the user, the set of predicted pages will be given by:

$$\Phi(RU(A))$$

3.4 Sequential Behavior Model as a Markov Chain

The problem of rule mining has also been presented from a Markov chain point of view. A process is a Markov Chain process with stationary transitions and countable space state [13] if that process moves through a countable set I of states, and at each stage, it decides where to go by a random mechanism which depends only on the current state and not on the previous history or even on the time.

The rule mining problem can be seen as a stochastic process which moves through a countable set of states $url_1, ..., url_p$, where p is the number of pages of that web server. In each state url_t, the next page visited will depend only on the actual page, and not on the previous history. Following these ideas, [2] presents a Markov model based approach for prediction and [4] uses a Markov model for finding shortcuts. [26] compares N-gram prediction models for different sizes of N, which are equivalent to constructing an $(N-1)$ order Markov Model. Each of these investigations consider only subsequences with consecutive accesses within transactions and do not present a formal model for taking into account the time gap between the antecedent and the consequent.

The Sequential Behavior Model proposed can also be seen from a Markov model point of view. For example $SR_{1,1,1}$ is equivalent to a first-order Markov model. In general a $SR_{n,1,1}$ is equivalent to an n-order Markov model, where each state will be identified with the antecedent of length $|A|$ of each rule. The confidence of each rule, which by definition will depend only on the $|A|$ elements of the antecedent, and not on the previous URLs, will be equivalent to the probability of the transition in the equivalent Markov model.

4. EXAMPLES OF SEQUENTIAL BEHAVIOR MODELS

4.1 Characteristics of the logs used

We have selected three sets of logs in order to cover different types of servers, ranging from small sites with few users to complex commercial sites with a large number of users.

The first log is from the Computer Science Department (CS) of the Polytechnic University of Madrid [10]. The training set is for September 2001. The test set has been defined using log data for 1st October 2001. This is an example of a small site, with a simple tree architecture and a small number of visitors.

The second log is from the NASA Kennedy Space Center server [20]. It contains 3,461,612 requests collected over the months of July and August 1995. This is an example of a medium site server, with a complex architecture and a medium number of users. The training set considered was July 1995, and the test set has been defined using log data for the 1st of August 1995.

The third site is ClarkNet [8], a commercial Internet site provider, which contains 3,328,587 requests over a period of two weeks. This is an example of a large commercial site, with a highly complex architecture and a high number of visitors. The training set has been defined from 28 August 1995 to 9 Sep 1995, and the test set is the log data for 10th Sep 1995.

These training sets are the inputs to the algorithms and filters presented in Section 3.1. Table 1 presents some characteristics of the training sets including the processing time of the algorithm presented in Table 1 for each log. Although the filtering process eliminates a lot of sessions, we keep the relevant part of the requests. The processing time is given for an implementation in LISP of the compiler, running with Linux, on a Pentium III 450MHz.

Table 1. Training Log Characteristics.

	CS	NASA	CLARKNET
Size	27M	160M	308.6M
Dates	September 2001	July 1995	28 Aug to 9 Sep, 1995
Processing Time	12 min	4 h 35 min	8 h 50 min
# of sessions before filtering	3,499	124,666	224,935
# of sessions after filtering	839	57,875	83,011
# of requests before filtering	6,244	352,844	511,536
# of requests after filtering	3,504	215,223	283,844

The characteristics of the test logs defined for each one of the training sets are given in Table 2. The processing time indicates the time needed to obtain the set of sessions from the logs using the algorithm presented in Figure 1. The number of sessions and number of pages requested shown in Table 2 are post-filtering.

Table 2. Test Log Characteristics.

	CS	NASA	CLARKNET
Size	170K	6.8M	19M
Date	Oct. 1st 2001	Aug. 1st 1995	Sep. 10th 1995
# of sessions	53	2753	5725
# of Pages requested	138	9521	18233
Average length of Session	2.6	3.4	3.9
Processing Time	25 sec.	2 min 40 sec	6 min 10 sec

4.2 Implementation and Result Analysis

For each log we have obtained $SR_{1,1,1}$. After that a threshold of 1% for the support and of 5% for the confidence has been applied, obtaining $SR_{1,1,1,1,5}$. Other values of support and confidence have been tested but the best prediction rate is obtained with 1% and 5%. The characteristics of these *SRs* are presented in Table 3. The processing time indicates the time needed to process each one of the training sets using the algorithm presented in Figure 2.

Table 3. Characteristics of the SR obtained.

	CS	NASA	CLARKNET
Processing Time	50 sec	3 min 20 sec	7 min 40 sec
# of SAR before applying thresholds	392	15,644	49,245
# of SAR after applying thresholds	94	116	166

Figure 4 presents the accuracy of the prediction system for CS, NASA and ClarkNet logs, with $SR_{1,1,1,1,5}$ and different Φ functions. The results are obtained comparing the actual next page of the session with the page or set of pages predicted by the system. The first set of columns represents the results for the function Φ defined as:

$$\Phi_1((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i})) = \{C_{i,j}\} \quad with\ \sigma_{i,j} \geq \sigma_{i,m} \forall m/m = 1..l_i, m \neq j \tag{1}$$

In other words, only the consequent with the highest confidence is given as the prediction. The rest of the functions Φ defined for each set of columns are, in order:

$$\Phi_2((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i})) = \{C_{i,j},C_{i,b}\} \quad with\ \sigma_{i,j} \geq \sigma_{i,b} \geq \sigma_{i,m} \forall m/m = 1...l_i, m \neq j, m \neq b \tag{2}$$

$$\Phi_3((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i})) = \{C_{i,j},C_{i,b},C_{i,h}\} \quad with\ \sigma_{i,j} \geq \sigma_{i,b} \geq \sigma_{i,h} \geq \sigma_{i,m} \quad \forall m/m = 1...l_i, m \neq j, m \neq b, m \neq h \tag{3}$$

$$\Phi_4((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i})) = \{C_{i,1},...,C_{i,l_i}\} \tag{4}$$

Function Φ_2 gives as the set of predicted pages the two consequents with higher confidence, Φ_3 the set of three consequents with higher confidence, and Φ_4 all the predictions that have a support and a confidence bigger than the thresholds used.

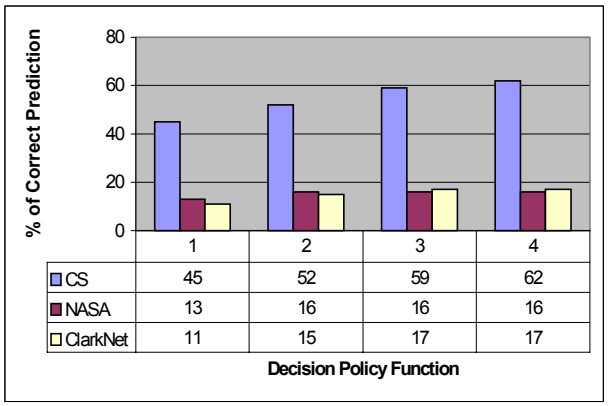

Figure 4. Results of the model for each set of logs.

As can be seen in Figure 4 the percentage of correct predictions for each system, in the worst case, is 45% for CS, 13% for NASA and 15% for ClarkNet. The result for CS is satisfactory in the sense that the prediction system obtained allows us to implement efficient intelligent services, as can be seen in Section 6. Nevertheless, the results for NASA and ClarkNet are not satisfactory. The cause for that difference in the prediction system is that the sites are actually very different. In CS we have a site with a well defined tree architecture, a low number of visitors and a low number of links per page (11 on average). This allows a simple system like $SR_{1,1,1,1,5}$ to capture the behavior of the users of that site. In other words, the users of CS site have a clear behavior pattern that can be satisfactorily captured with a Sequential Behavior Model.

The other two sites have a large number of pages, a higher number of links per page and a higher number of visitors. More importantly they have highly interconnected architectures. Based on these results we speculate that in those sites, user behaviors cannot be captured properly with a Sequential Behavior Model because access order is not a global property in complex sites. In the same sense, we speculate that a Sequential Behavior Model can efficiently capture the behavior of a site which has clear patterns, which occurs mainly in sites that have a tree architecture and a low number of visitors.

The inability of the proposed model to efficiently capture a global behavior in sites with complex structures and high numbers of visitors, suggests that a more local approach to the behavior should be taken. This variation of the model is presented in Section 5.

4.3 Other Sequential Behavior Models

For the CS and NASA logs we have also obtained other Sequential Behavior Models and have tested the accuracy of their predictions.

The SRs collected ranged from $SR_{1,1,1,5}$ to $SR_{N,1,1,1,5}$, where N is the integer part of the average length of the sessions of each log. This means that for CS we have obtained $SR_{1,1,1,1,5}$, $SR_{2,1,1,1,5}$ and for NASA $SR_{1,1,1,1,5}$, $SR_{2,1,1,1,5}$ and $SR_{3,1,1,1,5}$.

The improvement of the accuracy of the prediction for these systems, compared to the results of $SR_{1,1,1,1,5}$ presented in the previous section is at best 5%. This means that all of the systems contain basically the same information.

As we noted in Section 3.3 the set of rules of RU can include more than one SR. We tested the results of a prediction system with a RU defined as:

$$RU = \bigcup_{i=1}^{N} SR_{i,1,1,1,5}$$

In other words, RU is the union of all the SRs up to the average length of the session. Using as the decision policy function Φ_1, the accuracy of this Sequential Behavior Model is only 6% bigger as compared to $SR_{1,1,1,1,5}$. This implies that the information contained overlaps significantly.

Although in this specific case the idea of using more that one SR has not produced good results, more generally, it offers the option of creating more complex RUs in order to achieve a higher prediction accuracy.

5. IMPROVING THE PERFORMANCE OF THE SBM MODEL: PERSONALIZATION

As we found out in the previous section, the Sequential Behavior Model proposed in Section 3 is not capable of capturing the behavior of complex, highly interconnected sites. A more local approach is needed.

One method which could improve the model is personalization. The intuition here is that there is a set of users in each web site who are responsible for the best part of the load. These users pollute the behavior model of the rest of the users. We propose to have a personalized set of rules SR(user) for each frequent user of a site.

The main criticism that arises from this idea is that much space is needed to store the set of personalized sequential association rules of each user. Nevertheless, e-sites already have a lot of information about each user, ranging from name, address, or credit cards numbers to layouts and preferences, as can be seen in [5],[16] and [28].

The inclusion of a personal set of association rules, as will be shown, will not cause a large increase in needed storage. Of course not every user that accesses a site will automatically have a personal set of association rules. A policy for creation and destruction should be defined: a personal set of rules can be created for a user if that user accesses the site a minimum number of times in a given period of time, and can be destroyed if the user does not maintain a given rate of visits .

This new solution does not necessarily apply to every site. In order to be applied it should be demonstrated that the greatest part of the load of the site is produced by a core group of users. Those will be the users for which the system will define a personal set of association rules. Table 4 shows the characteristics of our training logs.

Table 4. Some characteristics of the Training Logs.

	NASA	CLARKNET
# of Sessions	124,666	224,935
# of Users	37,821	58,035
# of Users with just one visit	30,875	49,085

As can be seen in Table 4, in the case of the NASA log, 18% of the users are responsible of 75% of the visits. Similar behavior is also observed for the ClarkNet log, where 15% of the users are responsible for 78% of the visits. These sites posses the perfect characteristics for the implementation of a Personalized Sequential Behavior Model (PSBM).

5.1 Personalized Sequential Behavior Model

The set of sessions *S* is going to be clustered using *IP*. Given $|IP_D|$ the number of different *IPs* of *S* that have a ratio of visits in the training log bigger than or equal to the ratio defined to create a personal set of rules, and $|IP_k|$ the number of sessions of *S* that satisfy $IP=IP_k$, *S* can be expressed as:

$$S = \left\{ S_{IP_1}, ..., S_{IP_{|IP_D|}} \right\}$$
$$S_{IP_k} = \left\{ (IP_k, PAGES_1), ..., (IP_k, PAGES_{|IP_k|}) \right\}, \quad with\ k = 1...|IP_D|$$

The set of sessions that correspond to IPs that do not have enough visits to be considered for the construction of a PSBM are filtered out of the log.

In this context we define $SR(IP)_{|A|,n,|C|,\theta',\sigma'}$ as the set of sequential association rules that describe the behavior of user *IP*, using rules with a length $|A|$ of the antecedent, length $|C|$ of the consequents, a distance between the antecedent and the consequent of *n*, a minimum support of θ' and a minimum confidence of σ'.

$SR(IP)_{|A|,n,|C|,\theta',\sigma'}$ is obtained applying the algorithm presented in Figure 2 to each one of the clusters of S, and applying to the rules obtained the filters for the support and the confidence.

$$SR(IP_i) = Obtain\_SR(|A|, n, |C|, S_{IP_i}), \forall i = 1, ..., |IP_D|$$

The support of the rules of each $SR(IP_i)$ is obtained as:

$$\theta(A ?_1 ... ?_n C) = \frac{\left| S_l \in S_{IP_i} / (A ?_1 ... ?_n C) \in S_l \right|}{\left| S_{IP_i} \right|}$$

The model allows the construction of different $SR(IP)_{|A|,n,|C|,\theta',\sigma'}$ for different IPs. In our study, we will consider that all of the SRs constructed have the same $|A|$, *n*, $|C|$, θ', and σ' values.

Similar to the SBM introduced in Section 3, a Personalized Sequential Behavior Model is defined by a tuple (RU, Φ), where *RU* is defined as:

$$RU(IP, A) = \begin{cases} SR(IP_1)_{|A|,n,|C|,\theta',\sigma'}(A), \text{if IP=IP}_1 \\ ... \\ SR(IP_{|IP|})_{|A|,n,|C|,\theta',\sigma'}(A), \text{if IP=IP}_{|IP|} \\ SR_{|A|,n,|C|,\theta',\sigma'}(A), \text{ otherwise} \end{cases}$$

The *RU* of a *PSBM* is represented as the union of all the personal SRs plus the global $SR_{|A|,n,|C|,\theta',\sigma'}$. This global $SR_{|A|,n,|C|,\theta',\sigma'}$ will be used to model the behavior of the users that do not have a personal SR

The decision policy function Φ can be defined as a vector:

$$\Phi = (\Phi_1, \Phi_{IP_1}, ..., \Phi_{IP_{|IP|}})$$

Where Φ_1 defines the policy for the global SR, and the remaining functions define the policy for each one of the personal SRs. In our study we are going to consider a common decision policy function for all the personal SRs.

The tuple (RU, Φ) defines the on-line execution of the prediction system. Given *A* the click stream of the last $|A|$ pages requested by user *IP*, the set of predicted pages will be given by:

$$\Phi(RU(IP, A))$$

5.2 Implementation and Results Analysis

We have developed a *PSBM* for the NASA and ClarkNet logs. The *RU* implemented for both cases is shown below:

$$RU(IP, A) = \begin{cases} SR(IP_1)_{1,1,1,1,5}(A), \text{if IP=IP}_1 \\ ... \\ SR(IP_{|IP_D|})_{1,1,1,1,5}(A), \text{if IP=IP}_{|IP_D|} \\ SR_{1,1,1,1,5}(A), \text{ otherwise} \end{cases}$$

Where $|IP_D|$=6,494 for the NASA log, and $|IP_D|$=8,950 for ClarkNet. For the construction of *RU* we decided that any user that has more than two visits in the period of the training sets is a frequent user and should have a personal *SR*. Table 5 presents the characteristics of the personal SRs constructed.

Table 5. Characteristics of the Personal SR constructed.

	NASA	CLARKNET
# of personal SR	6946	8950
Average # of rules per SR	10.3	9.7
Total Processing Time	1 min 4 sec	2 min 30 sec

Figure 5 presents the prediction accuracy of the set of PSBM constructed for the NASA log and the ClarkNet log. The test logs in this case have been modified to contain only the set of visits of the users that have a personal SR. This allows us to obtain the prediction accuracy of the set of personal SRs.

Each set of columns presents the percentage of pages correctly predicted for each test log using two different policy functions.

Φ_1 gives the prediction accuracy using the consequent with the highest confidence of each *SR(IP)*. In both cases, NASA and ClarkNet, the system achieves at least a 44% correct prediction. Φ_2 considers the two consequents with highest confidence, and in that case the correct prediction is over 50%.

The global rate of correct prediction will be given by the prediction rate of the global SR and by the prediction rate of the set of personal SRs. In the case of the NASA logs, 75% of the visits are generated by users that have a personal SR. In the rest of the load, 25%, the prediction accuracy will be given by the global $SR_{1,1,1,1,5}$. Using Φ_1 as the decision policy function gives a total prediction accuracy of 0.36 (0.75*0.44+0.25*0.13). Using Φ_2 the prediction accuracy goes up to 0.43 (0.75*0.53+0.25*0.16).In the case of the ClarkNet log, when using Φ_1 the final accuracy is 0.33 and when using Φ_2 it is 0.42.

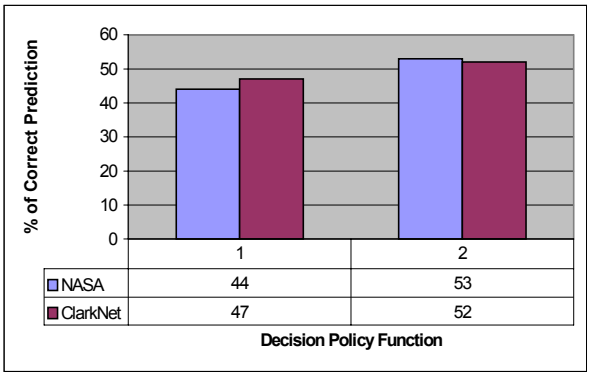

Figure 5. Prediction Accuracy for the NASA log.

These results prove that the personalization introduced in the prediction system produces satisfactory prediction rates with an acceptable increase in memory usage. This increase in memory is less significant when one takes into account that most sites already have a lot information about each user for other personalization purposes.

6. APPLICATION EXAMPLE: A WEB PREFETCHING SYSTEM

Several researchers have previously studied the use of a prediction system for intelligent prefetching. For example [30][31] constructs a case-base reasoning system using a window-based algorithm that finds pages that are visited together, [19] presents a new algorithm, WM_0, which is used to implement a web prefetching algorithm, and [11] presents a system in which clients initiate the prefetching of hyperlinks using the information that the server disperses to all clients.

To illustrate how a Sequential Behavior Model can help improve the generation and delivery of web content, we employ it to implement an intelligent prefetching system. The purpose of this example is also to more directly correlate prediction to performance by obtaining a breakdown of the cache behavior to understand when benefits arise because of the prediction system and when the prediction system is wrong but the page happen to be in the cache.

Figure 6 presents the general architecture of our approach. Each session of the test log has a cache associated with it. At the begining the cache is empty and we assume that we are working with an unlimited cache size. For each page of each session the system first looks if that page is in the local cache. If the page is not in the cache, an http request is send to the web server. At the same time, the cache sends the click stream to the web server. With this data the intelligent web server sends the requested web page to the cache and also, applying the Sequential Behavior Model to the click stream, the predicted page. If the prediction is correct, the next time the user sends an http request, the page will already be in the cache. Therefore, the latency perceived will be much smaller. Even in that situation, the cache will again send the click stream to the server, which applies the SBM to send the next page to be visited.

Figures 7 and 8 present the results of the CS and NASA test logs using the global Sequential Behavioral Model $SR_{1,1,1,1,5}$. Figure 9 present the results for the NASA log using the Personalized Sequential Behavior Model constructed with $SR(IP)_{1,1,1,1,5}$ for each frequent user and $SR_{1,1,1,1,5}$ as the global prediction system.

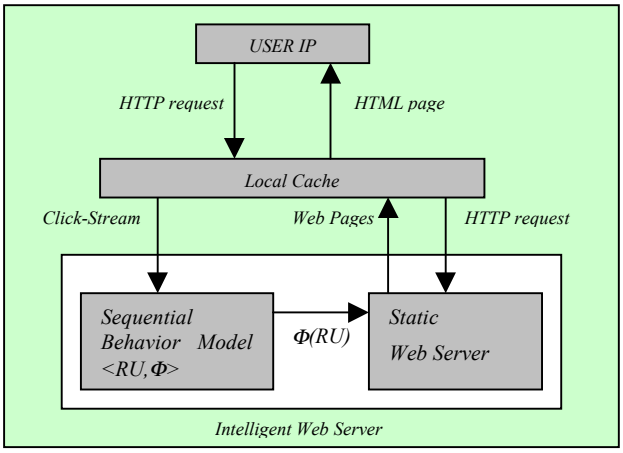

Figure 6. Architecture of the prefetching system

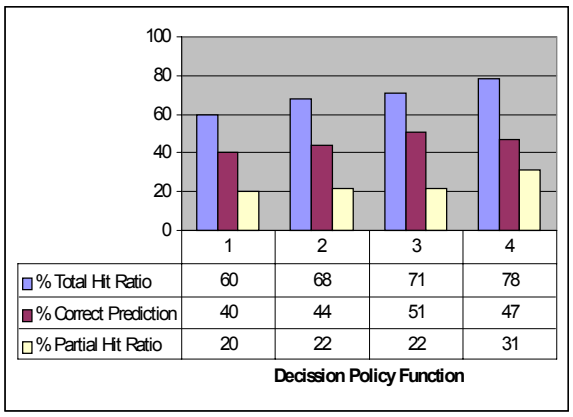

Figure 7. Results of the Prefetching system with CS.

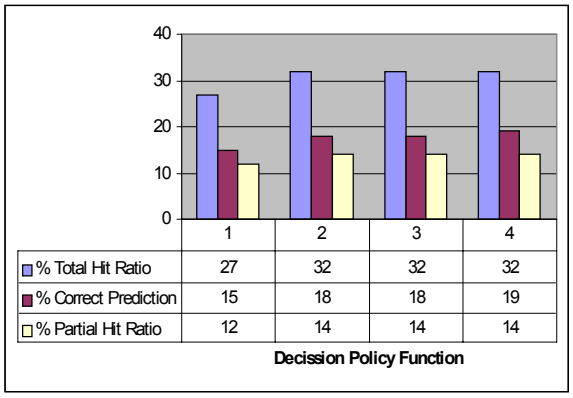

Figure 8. Results of NASA prefetching system based on SBM.

The figures present for different decision policy functions the total cache hit ratio (% Total Hit Ratio), where this ratio is divided into two parts: the number of times when the hit was caused because the prediction was correct (% Correct Prediction), and the number of times that the prediction was incorrect but the page happened to be in the cache (% Partial Hit Ratio). The functions Φ implemented are identical to the functions presented in Section 4.

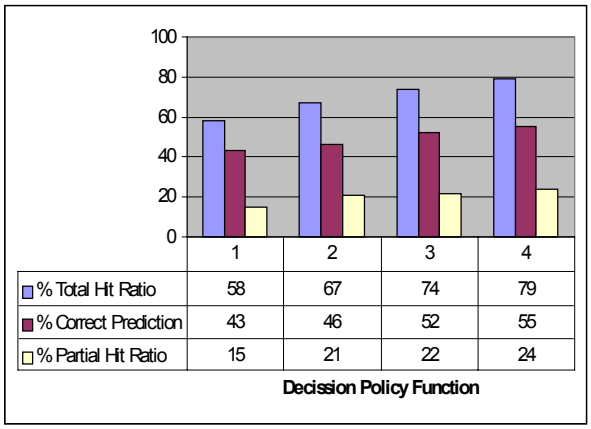

Figure 9. Results of NASA prefetching system based on PSBM

6.1 Result Analysis

Examining the results of the prefetching application developed we can see that, as expected, the greater the accuracy of the prediction system, the greater the total hit ratio of the cache is. We can also see that the Partial Hit Ratio (the number of times that the prediction system predicted incorrectly but the page happened to be in the cache) is responsible for at least 25% of the hits in the cache. We conclude that in these kinds of applications the inherent properties of the cache are as important as the prediction system in determining the total hit ratio.

This application also shows that using a PSBM in a highly complex site allows to obtain the same results as when using a global Sequential Behavior Model in a simple site. This proves that PSBM is able to efficiently capture the behavior of complex systems.

Although we have demonstrated the utility of our prediction model for caching static content, such content is reasonably well supported using current-day solutions such as caching architectures or content distribution. In fact prediction-based architectures have greater utility for serving dynamic and personalized web content, which is growing in popularity and in fact accounts for a large fraction of current day Internet traffic [22]. In this context, the Model can be used to predict the next page and use the prediction to pregenerate the page in order to reduce latency.

7. CONCLUSIONS AND FUTURE WORK

We have considered the problem of modeling the behavior of a Web user. A method to construct a Sequential Behavior Model has been proposed. This method is able to capture the inherent sequentiality of web visits. The model is constructed using a set of Sequential Association Rules which reflect the order in the set of URLs of the antecedent and the consequent, and also the distance between the antecedent and the consequent measured in the number *n* of clicks between the two click-streams. This distance allows to design systems that not only predict which pages are going to be visited but also when they are going to be visited. To the best of our knowledge, our model is the only one to reflect this distance metric.

The Sequential Behavior Model proposed is able to very efficiently capture the behavior of sites with a well-defined architecture and with a small set of users. For more complex sites we extend the Model with a personalization scheme where each frequent user has a personal set of rules. One criticism of this solution could be that it needs space to store the models. However we show that taking into account that most of the sites already have a lot of information about each user, the extra space needed is not significant.

The Model has been designed as a black box to be used in any application that is based in a prediction system. As an example of this implementation we have developed a prefetching system. We show that the Model achieves a high cache hit ratio both with the global and the personal approach. Also, we show that in this kind of application the inherent properties of the cache are as important as the prediction system in determining the total hit ratio.

The paper presents two schemes which represent two extremes of the Model. In one of them we only care about global behavior which produces a very small set of rules. This scheme is useful for sites with a simple architecture. The other solution only cares about local information, clustering the set of sessions according to its IP. This solution is useful for highly interconnected sites. We plan to generalize the model by defining intermediate solutions to find an appropriate trade off between space and prediction accuracy. These intermediate solutions will consist in clustering the sessions that correspond to IPs that share a common prefix. Another intermediate solution will consist of clustering the sessions according to their similarity taking into account the access order. With that set of clusters, a given IP will have its set of sessions belong to a set of clusters and the behavior of that IP can be inferred by mixing those clusters.

The filtering of records implemented does not remove the pages that are caused by backtraking clicks. This will happen when users realize that they have made wrong navigation choices and therefore choose to return to previously visited pages. Future versions of the filter will avoid this problem. With this new filter the results of the experiments may be better.

We plan to apply our prediction model to the NYU HOME page. NYU Home is a site that allows its users, the NYU community, to personalize their channel contents, ranging from e-mail, to news or weather forecasts. The site is slower that sites with static content because the pages have to be generated on real-time. As can be seen, this page has the ideal characteristics for the implementation of a PSBM: it has a large set of frequent users and the system already has personal information about each user. We will decrease the latency time for each user by pregenerating the pages using PSBM.

Also a deeper study of the impact on the parameters of the model in the prediction rate is needed. We are especially interested in studying the results depending on the parameter *n*, the distance in clicks between the antecedent and the consequent. The possibility of knowing not only what pages are going to be requested but also when are going to be requested can improve the efficiency of a lot of services (prefetching, recommendation systems, etc.). It will be also interesting to study the parameter C, the consequent of the rules and to find the optimum |C| for a given system. Having consequents with length bigger that one is useful for some applications such as construction of web pages in real time or prefetching.

8. ACKNOWLEDGMENTS

This work has been partially supported by the Spanish Department of Research grant EX2001-46775835.

9. REFERENCES

[1] Agrawal, R., Imielinski, T., Swami, A., Mining Association Rules between Sets of Items in Large Databases in Proceedings of the 1993 ACM SIGMOD Conference, Washington DC, USA, 1993

[2] Albrecht, D., Zukerman, I., Nicholson, A., Pre-sending documents on the WWW: A comparative study. IJCAI99-Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, 1999

[3] Ale, J.M., Rossi, G.H., AN Approach to Discovering Temporal Association Rules, in Proceedings of SAC'00, Marh 19-21, Como, Italy, 2000, pp. 294-300

[4] Anderson, C. R., Domingos, P., Weld, D.S., Adaptive Web Navigation for Wireless Devices, in Proceedings of the Seventeenth International Joint Conference on Artifical Intelligence (IJCAI-01), 2001

[5] Belkin, N., Helping People Find What They Don't Know, in Communications of the ACM, August 2000, Vol. 43, No. 8, pp.58-61

[6] Catledge, L.,Pitkow, J., Characterizing browsing behaviors on the world wide web, in Computer Networks and ISDN Systems, 27(6), 1995

[7] CERN Common Log Format, http://www.w3.org/Daemon/User/Config/Logging.html#common-logfile-format

[8] ClarkNet Internet Provider Log, http://www.web-caching.com/traces-logs.html

[9] Chen, M.S., Park, J.S., Yu, P.S., Efficient Data Mining for Path Traversal Patterns, in IEEE Transactions on Knowledge and Data Engineering, Vol. 10, No. 2, pp. 209-221, Apr. 1998

[10] Departamento de Tecnologia Fotonica Log (Univerisdad Politecnica de Madrid), http://www.dtf.fi.upm.es

[11] Duchamp, D., Prefetching Hyperlinks, In Proc. Second USENIX Symp. on Internet Technologies and Systems, USENIX, Boulder, CO, October 1999, pp. 127-138

[12] Etzioni, O., The World Wide Web: Quagmire or gold mine, in Communications of the ACM, Volume 39, No. 11, 1996, pp.65-68

[13] Freedman, D., Markov Chains, Holden-Day Series in Probability and Statistics, 1971

[14] Joshi, A., Joshi, K., Krishnapuram, R., On mining Web Access Logs, in Proceedings of the ACM-SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2000, pp. 63-69

[15] Krishnapuram, R., Joshi, A., Nasraoui, O., Yi, L., Low-Complexity Fuzzy Relational Clustering Algorithms for Web Mining, in IEEE Transactions on Fuzzy Systems, August 2001, Vol. 9 (4), pp. 595-608

[16] Manber, U., Patel, A., Robinson, J., Experience with Personaliztion on Yahoo!, in Communications of the ACM, August 2000, Vol. 43, No. 8, pp.35-39

[17] Masseglia, F., Poncelet, P., Cicchetti, R., An efficient algorithm for Web usage mining, in Networking and Information Systems Journal, Vol. X-nº X, 2000, pp. 1-X

[18] Mobasher, B., Cooley, R., Automatic Personalization Based on Web Usage Mining, in Communications of the ACM, vol. 43, num. 8, 2000, pp.142-151

[19] Nanopoulos, A., Katsaros, D., Manolopoulos, Y., Effective Prediction of Web-user Accesses: A Data Mining Approach, Proceeding of the WEBKDD 2001 Workshop, San Francisco, CA, 2001

[20] NASA Kennedy Space Center Log, http://www.web-caching.com/traces-logs.html

[21] Palpanas, T., Mendelzon, A., Web Prefetching using Partial Match Prediction, in Proceedings of the 4th Web Catching Workshop, Mar. 1999

[22] Shi, W., Wright, R., Collins, E., Karamcheti, V., Workload Characterization of a Personalized Web Site and Its Implications for Dynamic Content Caching, Technical Report TR2002-826, Department of Computer Science, New York Univeristy, April, 2002.

[23] Spiliopoulou, M., Pohle, C., Faulstich, L., Improving the Effectiveness of a Web Site with Web Usage Mining, in Proceedings of WEBKDD99, 1999, pp. 142-162

[24] Srikant, R., Yang, Y., Mining Web Logs to Improve Website Organization in Proceedings of WWW10, Hong Kong, May 1-5, 2001, pp. 430-437

[25] Srivastava, J., Cooley, R., Deshpande, M., Tan, P., Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data, in SIGKDD Explorations, ACM SIGKDD, Jan. 2000

[26] Su, Z., Yang, Q., Zhang, H., A prediction system for multimedia pre-fetching on the Internet, in Proceedings of the ACM Multimedia Conference 2000, ACM, October 2000

[27] VanderMeer, D., Dutta, K., Datta, A., Enabling Scalable Online Personalization on the Web, in Proceedings of EC'00, October 17-20, 2000, Minneapolis, Minnesota, pp. 185-196

[28] Wells, N., Wolfers, J., Finance with a Personalized Touch, in Communications of the ACM, August 2000, Vol. 43, No. 8, pp.31-34

[29] World wide web committee web usage characterization activity. http://www.w3.org/WCA

[30] Yang, Q., Tian-Yi, I., Zhang, H., Minin High-Queality Cases for Hypertext Prediction and Prefetching, in D.W. Aha and I. Watson (Eds.): ICCBR 2002, LNAI 2080, Springer-Verlag Berlin Heidelberg, pp. 744-755, 2001

[31] Yang, Q., Zhang, H., Li, I., Lu, Y., Mining Web Logs to Improve Web Caching and Prefetching in N. Zhong et al (Eds.): WI 2001, LNAI 2198, Springer-Verlag Berlin Heindelberg, pp. 483-492, 2001