A Customizable Behavior Model for Temporal Prediction of Web User Sequences

Enrique Frías-Martínez and Vijay Karamcheti

Courant Institute of Mathematical Sciences, New York University
715 Broadway, New York NY 10003 USA
{frias,vijayk}@cs.nyu.edu

Abstract. One of the important Internet challenges in coming years will be the introduction of intelligent services and the creation of a more personalized environment for users. A key prerequisite for such services is the modeling of user behavior and a natural starting place for this are Web logs. In this paper we propose a model for predicting sequences of user accesses which is distinguished by two elements: it is customizable and it reflects sequentiality. Customizable, in this context, means that the proposed model can be adapted to the characteristics of the server to more accurately capture its behavior. The concept of sequentiality in our model consists of three elements: (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 gap between the antecedent and the consequent in terms of the number of user clicks.

1 Introduction

Since Etzioni [12] first proposed the term "Web Mining" a lot of research has been done in the area of analyzing web servers logs to detect patterns and to find user characteristics. One topic that has received a lot of attention is modeling the behavior of web users, in other words, being able to predict the requests of users. Having such a model of user behavior has made possible the implementation of a great variety of intelligent services. Some examples of these services include: redesigning of web sites [24], personalization of 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. Clustering is a technique that can be used to group similar browsing patterns or to divide the web pages of a site into groups that are accessed together. Association rules detect frequent patterns among transactions. Despite their popularity, 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.

O.R. Zaïane et al. (Eds.): WEBKDD 2002, LNAI 2703, pp. 66–85, 2003.

In this paper we present a model that constructs sequential rules which, unlike clustering and association rules, capture the sequentiality and temporality in which web pages are visited. These rules are defined by an antecedent and a consequent. Both antecedent and consequent are subsequences of a session. In order to preserve sequentiality, the rules maintain the sequence of the click stream of the antecedent and of the consequent. The concept of temporality is reflected with a distance metric between the antecedent and the consequent measured by the number of user clicks to go from one to another.

Two features of our model distinguish it from previous work. The first feature is the concept of distance between the antecedent and the consequent. This concept 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 precisely when 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 rules's quality. 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 that shortcut is.

Second, our model addresses a shortcoming of current algorithms for prediction, which, traditionally, have not taken into account the characteristics of the specific web server they are trying to model. The prediction system we propose is customizable. This means that the prediction system can be adapted, depending on the characteristics of the server (number of pages, architecture of the server, number of links per page, etc.), in order to more accurately capture the behavior of its users. The model offers a balance between the storage space needed and the accuracy of the prediction system. The balance will be mainly determined by the application that is going to be developed using the prediction system.

The organization of the paper is as follows. Section 2 summarizes the motivation and prior work done in this area. Section 3 presents the Customizable Sequential Behavior Model. Section 4 implements some examples of the Customizable Sequential Behavior Model and analyzes the results. Finally, in Section 5 we conclude the article and discuss 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] and [14]. Several 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 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 that additionally preserves the order in which those web pages are visited. The model expresses those patterns using rules.

This ability to detect 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. 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.

Traditionally, the models and algorithms developed for user access prediction ([2],[4],[26], etc.) apply the same approach to all servers regardless of their characteristics. In our approach, we recognize that the ability of the model to capture

user behavior depends on the characteristics of the site. In order to capture efficiently user's behavior, the model we propose is customizable. This means that it can be adapted to the inherent characteristics (number of web pages, number of users, architecture of the site, etc.) of each server. This customization capability makes possible trade-offs between the number of rules and the prediction accuracy of the prediction system.

3 Customizable Sequential Behavior Model

In this section we present the algorithms and ideas behind the concept of Customizable Sequential Behavior Model. We begin with the preparation of the data and then we present the Clustering of Users. Next, the concept of Sequential Association Rule and the definition of Customizable Behavior Model is given.

3.1 Preparing the Data

The syntax of the log file that contains 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 remote logname of the user, (3) the access date and time, (4) the request method, (5) the URL of the page , (6) the protocol (HTTP 1.0, HTTP 1.1,etc.), (7) the return code and (8) the number of bytes transmitted. 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 of a user. Also, the W3C defines a user as a single individual that is accessing one or more servers from a browser.

The first step for preparing the data is the transformation of the set of logs into sessions. We have defined a compiler that transforms a set of log entries L,

$$\begin{gathered} L = \left\{L_1, ..., L_{|L|}\right\} \\ L_i = (IP_i, LOGNAME_i, TIME_i, METHOD_i, URL_i, \\ PROT_i, CODE_i, BYTES_i), \forall i / i = 1...|L|, \end{gathered} \tag{1}$$

into a set of sessions S,

$$S = \left\{S_1, ..., S_{|S|}\right\}, \tag{2}$$

where $|L|$ is the number of log entries in L and $|S|$ is the number of sessions of S. Each session is defined as a tuple *(USER,PAGES)*:

$$S_i = (USER_i, PAGES_i), PAGES_i = \{url_{i,1}, ..., url_{i,p_i}\}, i = 1... |S|, \tag{3}$$

where *USER* identifies the user of the session, *PAGES* the set of pages requested, and p_i is the number of pages requested by user $USER_i$ in session S_i.

```
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 USER_k = USER_i
        if (TIME_i − END_TIME(S_k)) < Δt then
          S_k = (USER_k, PAGES_k ∪ URL_i)
        else
          CLOSE SESSION (S_k)
          OPEN NEW SESSION(USER_i, (URL_i))
        end if
      else
        OPEN NEW SESSION(USER_i, (URL_i))
      end if
    end if
  end for
```

Fig. 1. Compiler that transforms web log data into a set of sessions.

Working with the CERN Common Log Format, a user *USER* is defined as,

$$USER_i = (IP_l, LOGNAME_{l,m}), 1 \leq l \leq d, 1 \leq m \leq h(l), \tag{4}$$

where d is the number of different *IPs* of the log, and *h(i), i=1,...,d*, the number of different *LOGNAMEs* of IP_i.

The definition of a user is highly dependent on the log format and on the information available. In order to generalize the algorithms and ideas presented we are going to work with a generic definition of user, *USER*.

The set of *URLs* that form a session satisfy the requirement that the time elapsed between two consecutive requests is smaller than 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. 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,

$$N = \frac{\sum_{i=1}^{|S|} p_i}{|S|}. \tag{5}$$

3.2 Clustering of Users and Sessions

The set of different users of a system is expressed by,

$$USERS = \left\{USER_1, ..., USER_{|USERS|}\right\}, \tag{6}$$

where $|USERS|$ is the total number of different users of the log. This set of users is going to be clustered according to a cluster policy function P in order to create a customizable model. The purpose of these clusters is to group the users that have the same behavior and to allow a trade-off between the size and the personalization capabilities provided by the model.

Given p the number of clusters defined by the function P, the set of clusters of users CU can be expressed as:

$$\begin{array}{l} P : USER_i \rightarrow \{P_1, ..., P_p\}, \forall i = 1, ..., |USERS| \\ CU = \{CU_1, ..., CU_p\} \\ CU_i = \{USER_x / P(USER_x) = P_i\}, i = 1, ..., p \text{ and } 1 \leq x \leq |USERS| \\ \bigcup_{i=1}^{p} CU_i = USERS \\ CU_i \cap CU_k = \emptyset, \forall i, k \text{ with } i \neq k \end{array} \tag{7}$$

For example, given the web server of a university department a possible clustering policy function is P={*Professor, Graduate_Students, Students, Others*}.

The classification of users is going to be used to cluster the set of sessions S. The set of clustered sessions CS, can be expressed as:

$$\begin{array}{l} CS = \{CS_1, ..., CS_p\} \\ CS_i = \cup S_k / USER_k \in CU_i, \forall i = 1, ..., p, \forall k = 1, ..., |S| \end{array} \tag{8}$$

Each CS_i groups the sessions of the users that are part of the same CU_i cluster.

3.3 Sequential Association Rules

The concept of Sequential Association Rule (SAR) is based on the notion of 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) relates an antecedent A to a consequent C in terms of the temporal distance between them. 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, the SAR is defined as follows:

$$\begin{array}{c} A \underset{n}{\rightarrow} C \\ A := url_j, ..., url_{|A|+j} \\ C := url_l, ..., url_{|C|+l} \\ l = |A| + j + n + 1, \ n \in \mathrm{N}^+ \end{array} \tag{9}$$

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

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 a given CS_i contains an N-Gram, with N=$|A| + |C|$+n, that satisfies,

```
Input : |A|, n, |C|, CS_i
Output : SR(CS_i)_{|A|,n,|C|}
function Obtain_SR( |A|, n, |C|, CS_i)
  SR(CS_i)_{|A|,n,|C|} = ∅
  for each S_k ofCS_i
    for j = 1 to p_k
      if j + |A| + n + |C| ≤ p_k
        SAR = url_{k,j}, ..., url_{k,j+|A|} →_n url_{k,j+|A|+n}, ..., url_{k,j+|A|+n+|C|}
        if (SAR,Counter) ∈ SR(CS_i)_{|A|,n,|C|} then
          Counter = Counter + 1
        else
          SR(CS_i)_{|A|,n,|C|} = SR(CS_i)_{|A|,n,|C|} ∪ (SAR, 1)
        end if
      end if
    end for
  end for
```

Fig. 2. Pseudo-code of the algorithm developed to obtain $SR(CS_i)_{|A|,n,|C|}$.

$$S_k = (..., url_{k,j}, ..., url_{k,j+|A|}, url_{k,j+|A|+1}, ..., url_{k,j+|A|+n}, url_{k,l}, ..., url_{k,l+|C|}, ...), \text{ with } l = j + |A| + n + 1 \ . \tag{10}$$

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 CS_i where the rule successfully applies. The support of a rule is given by,

$$\theta(A?_1...?_nC) = \frac{|S_k \in CS_i/(A?_1...?_nC) \in S_k|}{|CS_i|}, \tag{11}$$

where $?_1...?_n$ represents the set of any n pages in the session, and $A?_1...?_nC \in S_k$ is defined as an N-Gram of S_k. 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,

$$\sigma(A?_1...?_nC) = \frac{\theta(A?_1...?_nC)}{\theta(A)}. \tag{12}$$

$SR(CS_i)_{|A|,n,|C|}$, for the set of sessions grouped by CS_i, 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 $SR(CS_i)_{|A|,n,|C|}$.

In order to preserve 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 $SR(CS_i)_{|A|,n,|C|,\theta',\sigma'}$ as the rules of $SR(CS_i)_{|A|,n,|C|}$with a support bigger that θ' and a confidence bigger than σ'.

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

In order to optimize the storage and access to the rules that define the set of rules $SR(CS_i)_{|A|,n,|C|,\theta',\sigma'}$, 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 $SR(CS_i)_{|A|,n,|C|,\theta',\sigma'}$ is:

$$SR(CS_i)_{|A|,n,|C|,\theta',\sigma'}(A) = \begin{cases} ((C_{1,1},\theta_{1,1},\sigma_{1,1}),...,(C_{1,l_1},\theta_{1,l_1},\sigma_{1,l_1})), \ A = A_1 \\ ... \\ ((C_{k,1},\theta_{k,1},\sigma_{k,1}),...,(C_{k,l_k},\theta_{k,l_k},\sigma_{k,l_k})), \ A = A_k \\ \emptyset, \ \text{Otherwise} \end{cases} \tag{13}$$

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 $SR(CS_i)_{|A|,n,|C|,\theta',\sigma'}$, $C_{i,j}$, i=1...k, j=1...l_k , with $|C_{i,j}| = |C|$, is the set of consequents for each antecedent A_i, and l_k is the number of consequents of each antecedent.

3.4 Sequential Behavior Model

The concept of Sequential Behavior Model (SBM) defines a prediction system based on the Sequential Association Rules introduced in the previous section. 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 is defined as:

$$RU(USER, A) = \begin{cases} SR(CS_1)_{|A|,n,|C|,\theta',\sigma'}(A), \text{if USER} \in CU_1 \\ ... \\ SR(CS_p)_{|A|,n,|C|,\theta',\sigma'}(A), \text{if USER} \in CU_p \end{cases} \tag{14}$$

The function Φ is defined as:

$$\begin{gathered} \Phi((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i})) = \{C_{i,j},...,C_{i,b}\} \\ with\ 1 \leq \text{i} \leq \text{k},\ 1 \leq j \leq l_k, 1 \leq b \leq l_k, \end{gathered} \tag{15}$$

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

$$\begin{gathered} \Phi((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i})) = \{C_{i,j}\} \\ with\ \sigma_{i,j} \geq \sigma_{i,m} \forall m/m = 1...l_i, m \neq j, \end{gathered} \tag{16}$$

$$\begin{gathered} \Phi((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i})) = \{C_{i,j}\} \\ with\ \theta_{i,j} \geq \theta_{i,m} \forall m/m = 1...l_i, m \neq j, \end{gathered} \tag{17}$$

$$\begin{array}{l} \Phi((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i})) = \{C_{i,j},C_{i,b}\} \\ with\ \sigma_{i,j} \geq \sigma_{i,b} \geq \sigma_{i,m} \forall m/m = 1...l_i, m \neq j, m \neq b. \end{array} \tag{18}$$

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 *USER*, the set of predicted pages will be given by,

$$\Phi(RU(USER,A)). \tag{19}$$

3.5 Classes of Sequential Behavior Models

The web servers found in Internet have very different sets of characteristics, differing in the number of users, the number of web pages that the server has, the architecture of the web server, the number of links per page, etc.

The prediction systems developed to date ([2],[4],[11],[26], etc.) do not consider the different characteristics of the web servers in order to more accurately model their behavior. We recognize that the same model cannot be used to generate a prediction system for a university department server and to capture the behavior of an e-commerce site, because the two sites exhibit vastly different behavior. In order to solve this problem, our Sequential Behavior Model is customizable. This means that it can be adapted to the characteristics of the server to more accurately capture its behavior. The Model presents three different classes: global, personalized and cluster.

Global Sequential Behavior Model (GSBM). The GSBM considers only a single cluster of users. This means that the system will also consider just one cluster of the set of sessions. Formally,

$$\begin{array}{l} p = 1 \\ CU = \{CU_1\} \\ P = \{P_1\}\,; P_1 = \text{"Any user of the system"} \\ CS = \{CS_1\} = S \end{array} \tag{20}$$

The Global Sequential Behavior Model will be defined by the tuple *(RU,Φ)*, where RU is expressed as:

$$RU(USER,A) = \left\{SR(CS_1)_{|A|,n,|C|,\theta',\sigma'}(A), \text{if } USER \in CU_1\right. . \tag{21}$$

The storage capacity needed by a GSBM is generally small because it has only one set of rules. On the other hand, the personalization capabilities offered are usually limited.

Clustered Sequential Behavior Model (CSBM). The Clustered Model considers a set of p clusters. These clusters should be designed to group the set of users that have common behavior. Each set of users will end up having a set of rules that describe their behavior in *RU*.

This profile produces bigger models than the Global approach but also increases the personalization capabilities. The size and the prediction accuracy will depend on the number of clusters that the system has defined.

Personalized Sequential Behavior Model (PSBM). The Personalized Model is a particular example of the Clustered Model. In this case, each one of the users of a system is placed in its own cluster, which means that each user has its own set of rules to describe his/her behavior. The storage space needed by this profile is, typically, bigger than the other two but it also offers more personalization capabilities.

This profile is very interesting considering that an increasing number of sites identify its users (using password or cookies) to provide them with personalized services. Having a model that describes the behavior of each user will make it possible to provide a higher degree of adaptation and personalization.

The main criticism that arises from this idea is that too much space is needed to store each user's set of personalized sequential association rules. Nevertheless, e-sites already posses 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 in section 4.5, will not cause a large increase in the amount of needed storage.

Additionally, it is not necessary for all the users of a server that implements a Personalized Sequential Behavior Model to have a personal set of rules. A more useful approach considers that, given the set of users of a server, a smaller subset is typically responsible for the largest part of the system's load. Taking this characteristic into account, each member of this set of frequent users will have a personal cluster. The non-frequent users will be grouped in another cluster and will share a set of rules. Formally, being g the number of frequent users in the system,

$$\begin{aligned} &p = g + 1 \\ &CU = \{CU_1, ..., CU_g, CU_{g+1}\} \\ &P = \{P_1, ..., P_g, P_{g+1}\} \\ &CS = \{CS_1, ..., CS_g, CS_{g+1}\} \end{aligned} \tag{22}$$

The number of clusters is $g+1$, where the last cluster groups the non-frequent users of the system. The clustering policy is defined as P_1=*Frequent User #1*, ...,P_g=*Frequent User #g*, P_{g+1}=*All non-frequent users*. CS_1 groups the set of sessions of the frequent user #1 and CS_{g+1} groups all the sessions of the non-frequent users.

The Personalized Sequential Behavior Model can be defined by the tuple *(RU,Φ)*, where RU is expressed as:

$$RU(USER, A) = \begin{cases} SR(CS_1)_{|A|,n,|C|,\theta',\sigma'}(A), if \text{ USER is Frequent User 1} \\ ... \\ SR(CS_g)_{|A|,n,|C|,\theta',\sigma'}(A), if \text{ USER is Frequent User g} \\ SR(CS_{g+1})_{|A|,n,|C|,\theta',\sigma'}(A), if \text{ USER} \in \text{CU}_{\text{g+1}} \end{cases} \quad (23)$$

3.6 General Rules for Application of the Classes.

This set of classes give the designer of the prediction system the capability of choosing an appropriately trade-off between the memory needed by the set of rules and the personalization capabilities provided. Each one of the profiles is useful in different contexts:

• The Global Sequential Behavior Model is likely able to accurately capture the user behavior of a simple site. We consider a simple site as one that has a small number of users and/or a small number of web pages and/or a simple architecture (normally a tree architecture). A typical example of this kind of site is the web server of a university department.

• The Clustered Sequential Behavior Model is capable of modeling the behavior of more complex systems. A complex system is defined as a system with a high number of users and/or a high number of web pages and/or a highly interconnected structure. The definition of the clustering policy function P is application dependent.

• The Personalized Sequential Behavior Model is recommended for highly complex sites and/or for sites that need individual information for each user.

4 Examples of Sequential Behavior Models

This section evaluates the performance of different Sequential Behavior Models for different web logs.

4.1 Characteristics of the Logs Used

e have selected three representative 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 the month of September 2001. The test set has been defined using log data for a single day, 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 for the month of 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. Table 1 presents some characteristics of the training sets including the processing time of the algorithm presented in Fig. 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 CLISP of the compiler, running with Linux, on a Pentium III 450MHz machine.

Table 1. Training Log Characteristics.

	CS	NASA	CLARKNET
Size	27M	160M	308.6M
Dates	Sep. 2001	Jul. 1995	28 Aug.-9 Sep. 1995
Processing Time	12 min	4 h 35 min	8 h 50 min
# of sessions before filter	3,499	124,666	224,935
# of sessions after filter	839	57,875	83,011
# of HTML requests before filtering	6,244	352,844	511,536
# of HTML 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 Fig. 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 of Global Sequential Behavior Models

We have obtained the Global Sequential Behavior Model of each log as defined by $SR(CS_1)_{1,1,1}$. After that, a threshold of 1% for the support and of 5% for the confidence has been applied, obtaining $SR(CS_1)_{1,1,1,1,5}$. Other values of support and confidence were 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 Fig. 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 of SR before applying thresholds	392	15,644	49,245
# of SAR of SR after applying thresholds	94	116	166

Fig. 3 presents the percentage of correct prediction for CS, NASA and ClarkNet logs, with RU defined by $SR(CS_1)_{1,1,1,1,5}$ and different Φ functions. The percentage of correct prediction is 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:

$$\begin{aligned}&\Phi_1((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i}))=\{C_{i,j}\}\\&with\ \sigma_{i,j}\geq\sigma_{i,m}\forall m/m=1...l_i,m\neq j.\end{aligned}\tag{24}$$

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

$$\begin{aligned}&\Phi_2((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i}))=\{C_{i,j},C_{i,b}\}\\&with\ \sigma_{i,j}\geq\sigma_{i,b}\geq\sigma_{i,m}\forall m/m=1...l_i,m\neq j,m\neq b,\end{aligned}\tag{25}$$

$$\begin{aligned}&\Phi_3((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i}))=\\&\{C_{i,j},C_{i,b},C_{i,h}\}\ with\ \sigma_{i,j}\geq\sigma_{i,b}\geq\sigma_{i,h}\geq\sigma_{i,m}\\&\forall m/m=1...l_i,m\neq j,m\neq b,m\neq h,\end{aligned}\tag{26}$$

$$\Phi_4((C_{i,1},\theta_{i,1},\sigma_{i,1}),...,(C_{i,l_i},\theta_{i,l_i},\sigma_{i,l_i}))=\{C_{i,1},...,C_{i,l_i}\}.\tag{27}$$

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

The inability of the GSBM, as shown in Fig. 3, to efficiently capture global behavior in sites with complex structures, like NASA and ClarkNet, suggests that a more local approach should be taken.

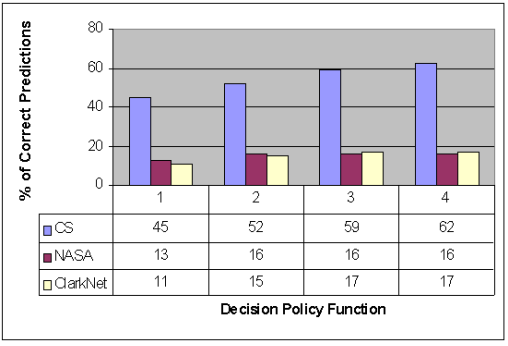

Fig. 3. Results of the model for each set of logs.

4.3 Implementation of Clustered Sequential Behavior Models Based on IP

In this section we present two CSBM examples for which the clustering policy function is defined using the IP address. In the first example, IP/16, the set of users that have the same 16 first bits of their IP equal are going to be part of the same cluster. In the second one, IP/24, the clusters will be obtained using the 24 first bits, working at the subnet level. The objective of this policy is to cluster the users that share a common behavior. This is based on the observation that the users that come from the same subnet have a greater probability of having similar behavior than two users that have completely different IPs. Table 4 presents the characteristics of the clusters for IP/16 .

Table 4. Characteristics of the clusters of IP/16.

	CS	NASA	CLARKNET
# of different IPs	840	37,821	58,035
# of Clusters	236	6,667	8,510
Processing Time to obtain Clusters	2 sec.	2 min. 10 sec.	3 min 40 sec.
Average # of IP per cluster	3.5	5.6	6.8
Processing Time of $SR(CS_i)_{1,1,1,1,5}$,$i=1,\ldots,p$	15 sec.	40 sec.	1 min. 30 sec.
Average # of SAR per cluster	4.6	6.3	7.2

For each cluster we have obtained $SR(CS_i)_{1,1,1,1,5}$, $i=1,\ldots,p$, with $p = 236$ for the CS log, $p = 6,667$ for the NASA log, and $p = 8,510$ for the ClarkNet log. Table 5 presents the characteristics of the clusters for IP/24.

For each cluster we have obtained $SR(CS_i)_{1,1,1,1,5}$, $i=1,\ldots,p$, with $p = 415$ for the CS log, $p = 23,183$ for the NASA log, and $p = 36,257$ for the ClarkNet log. Fig. 4 presents the accuracy of the prediction system for IP/16 and IP/24 for each log using the decision policy functions Φ_1 and Φ_2 previously defined.

Table 5. Characteristics of the clusters of IP/24.

	CS	NASA	CLARKNET
# of different IPs	840	37,821	58,035
# of Clusters	415	23,183	36,257
Processing Time needed to obtain the Clusters	2 sec.	2 min. 40 sec.	6 min. 40 sec.
Average # of IP per cluster	2.0	1.6	1.6
Processing Time for $SR(CS_i)_{1,1,1,1,5}, i=1,\ldots,p$	20 sec.	55 sec.	2 min. 10 sec.
Average # of SAR per cluster	2.9	6.2	5.2

In order to generate an efficient CSBM the training log should contain elements of all the possible clusters that the set of visitors of the system may form. This allows the prediction system to give a prediction for any possible visitor. If the training log lacks some of these users, the prediction system will not be able to give a recommendation for them.

In the testing logs defined for CS, NASA and ClarkNet only 2%, 5% and 6% of the sessions correspond to a user that is not included in any cluster. For those users, the system's prediction is obviously wrong. When the percentage of sessions that are not part of any of the clusters generated with the training log is significant compared with the total amount of visits, a GSBM can be applied for those users. This solution is similar to the idea presented in PSBM, where a cluster groups all non-frequent users of the system.

4.4 Implementation of Personalized Sequential Behavior Models

As presented in section 3.4.3 PSBMs are especially suitable for systems where a core of users is responsible for the best part of the load. As it will be shown, both NASA and ClarkNet logs have this characteristic. For these logs, we define a frequent user as the one that has more than two sessions in the training log. Also, because the logs lack from any kind of information about each individual

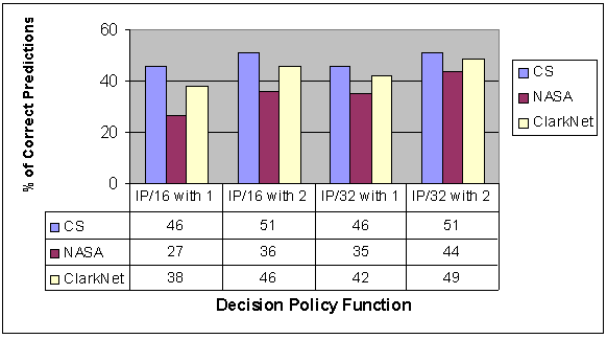

Fig. 4. Results of the CSBM with IP/16 and IP/24.

user, we will identify an IP as an user, bearing in mind that a single IP can be used by a group of users.

Table 6 shows the characteristics of our training logs, focusing this time on the data needed to identify the set of frequent users.

Table 6. 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 session	30,875	49,085

As can be seen in Table 6, in the case of the NASA log, 18% of the users are responsible for 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 posseses the perfect characteristics for the implementation of a PSBM. For each frequent user we obtain $SR(CS_i)_{1,1,1,1,5}$, $i=1,\ldots,g$ where $g = 6,946$ for the NASA log, and $g = 8,950$ for ClarkNet. Table 7 presents the characteristics of the *RUs* constructed.

Table 7. Characteristics of the Personal SR constructed.

	NASA	CLARKNET
g	6946	8950
Average # of rules per $SR(CS_i)$, $i=1,\ldots,g$	10.3	9.7
Total Processing Time	1 min 4 sec	2 min 30 sec

Fig. 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 frequent users. This allows us to obtain the prediction accuracy of the set of personal *SRs*, $SR(CS_i)$, $i=1,\ldots,g$.

Each set of columns of Fig. 5 presents the percentage of pages correctly predicted for each test log using two different policy functions. Using Φ_1, in both NASA and ClarkNet, the prediction system achieves at least a 44% correct prediction. With Φ_2, the correct prediction rate is over 50% for both logs.

The final prediction rate will be given by the prediction rate of $SR(CS_{g+1})$, that models the non-frequent users, 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 frequent users. For the rest of the load, 25%, the prediction accuracy will be given by $SR(CS_{g+1})$. 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 $\Phi_1$the final accuracy is 0.33 and when using Φ_2 it is 0.42.

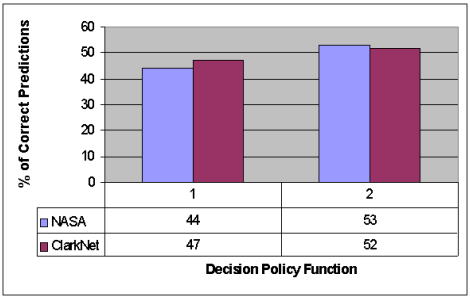

Fig. 5. Prediction Accuracy for the NASA log.

4.5 Results Analysis

As can be seen in Fig. 3, the percentage of correct prediction, using the GSBM for each log is, in the worst case, 45% for CS, 13% for NASA and 15% for ClarkNet.

The result for CS is satisfactory in the sense that the prediction system obtained would allow us to implement efficient intelligent services. Nevertheless, the results for NASA and ClarkNet are not satisfactory. The cause of this 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. The other two sites have a larger number of pages, a higher number of links per page and a higher number of visitors. More importantly, they have a highly interconnected architecture. Based on these results we speculate that in those sites, user behaviors cannot be captured properly with a Global Sequential Behavior Model because access order is not a global property in complex sites.

The results of the CSBM based on IP (Fig. 4) prove that a more local approach makes possible to obtain a better prediction rate in complex sites. In this case, and using IP/16 as the clustering policy function, the NASA prediction rate, in the worst case, goes up to 27% (14% more that the GSBM) and ClarkNet up to 36% (20% more that the GSBM). Using IP/24 allows a more local approach than IP/16, and that implies an increment in the prediction rate: 35% for NASA (7% more that IP/16 and 21% more that GSBM), and 42% for ClarkNet (4% more that IP/16 and 24% more that GSBM). Nevertheless, a more local approach in the case of a simple site, like CS, does not necessarily mean a better prediction rate. As we can see, the prediction rate for CS stays basically the same for both IP/16 and IP/24. With these results we speculate that a Global 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.

Although the CSBM increases the prediction rate of complex sites, this prediction rate is not necessarily enough for any intelligent application that is going to be implemented using the proposed model. For those cases, a PSBM can be used.

Using a PSBM, NASA prediction rate goes up to 44% (29% more that the GSBM and 17% more that the CSBM with IP/16), and ClarkNet goes up to 47% (36% more that the GSBM and 9% more that the CSBM with IP/16). And this is for the worst case, because when using more complex decision policy functions the successful prediction rate goes up to 53% for NASA and 52% for ClarkNet.

These results show that PSBM produces for highly complex sites the same successful prediction rates as the GSBM produces for simple sites. The PSBM is able to efficiently capture the behavior of complex highly interconnected sites 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.

The results also prove that our model can be adapted to the different characteristics of the site that is going to be modeled. In other words, that the model is customizable.

5 Conclusions and Future Work

We have considered the problem of modeling web user behavior. For that purpose, a Customizable Sequential Behavior Model has been designed.

One of the distinguishable characteristics of our model is that it is customizable. The necessity of a customizable model comes from the fact that Internet servers have very different characteristics. Our model can be configured to the characteristics of the server in order to obtain the maximum possible efficiency. Our Customizable Model allows a trade-off between the number of rules and the prediction accuracy.

Also, the model 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 of 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 makes it possible 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 that is customizable and that incorporates a distance metric in its rules.

Despite the encouraging results described in Section 4, a deeper study of the impact of the parameters of the model on the prediction rate is needed. We are especially interested in studying how prediction accuracy depends 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 they are going to be requested can improve the efficiency of a lot of services (prefetching, recommendation systems, etc.). It will also be interesting to study the parameter C, especially study the optimum value $|C|$ for a given set of sessions. Having consequents with $|C| > 1$ is useful for some applications such as the construction of web pages in real time or prefetching.

We also plan to apply our prediction model to the NYU Home web 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 than sites serving static content because the pages have to be generated in real-time. As can be seen, this server 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. A possible use of the prediction model we have described here is to decrease the user perceived latency at such sites by pregenerating the pages using a PSBM.

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. In: 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) 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, Vol. 43, No. 8, (2000) 58–61
6. Catledge, L.,Pitkow, J.: Characterizing browsing behaviors on the world wide web. In Computer Networks and ISDN Systems, Vol. 27, No. 6 (1995)
7. CERN Log Format, http://www.w3.org/Daemon/User/Config/Logging.html
8. ClarkNet Internet Provider Log, http://www.web-caching.com/traces-logs.html
9. Chen, M.S., Park, J.S.: Efficient Data Mining for Path Traversal Patterns. In IEEE Transactions on Knowledge and Data Engineering, Vol. 10, No. 2 (1998) 209–221
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 (1999) 127–138
12. Etzioni, O.: The World Wide Web: Quagmire or gold mine. In: Communications of the ACM, Volume 39, No. 11 (1996) 65–68
13. Freedman, D.: Markov Chains. In: Holden-Day Series in Probability(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) 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, Vol. 9 (4), (2001) 595–608
16. Manber, U., Patel, A., Robinson, J.: Experience with Personaliztion on Yahoo!. In: Communications of the ACM, Vol. 43, No. 8 (2000) 35–39
17. Masseglia, F., Poncelet, P., Cicchetti, R.: An efficient algorithm for Web usage mining. In Networking and Information Systems Journal, Vol. X,n^o X (2000) 1-X
18. Mobasher, B., Cooley, R.: Automatic Personalization Based on Web Usage Mining. In Communications of the ACM, Vol 43:8 (2000)142-151
19. Nanopoulos, A., Katsaros, D., Manolopoulos, Y.: Effective Prediction of Web-user Accesses: A Data Mining Approach. In: 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 (1999)
22. Shi, W., Wright, R., Collins, E., Karamcheti, V.: Workload Characterization of a Personalized Web Site and Its Implications for Dynamic Content Caching. In: Proceedings of the 7th International Conference on Web Content Caching and Distribution (WCW'02), Boulder, CO (2002)
23. Spiliopoulou, M., Pohle, C., Faulstich, L.: Improving the Effectiveness of a Web Site with Web Usage Mining. In: Proceedings of WEBKDD99 (1999) 142–162
24. Srikant, R., Yang, Y.: Mining Web Logs to Improve Website Organization. In: Proceedings of WWW10, Hong Kong (2001) 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 (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 (2000)
27. VanderMeer, D., Dutta, K., Datta, A.: Enabling Scalable Online Personalization on the Web. In Proceedings of EC'00 (2000) 185–196
28. Wells, N., Wolfers, J.: Finance with a Personalized Touch. In: Communications of the ACM, Vol. 43:8 (2000) 31–34
29. WWW 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 (2001) 744–755
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 (2001) 483–492