Modeling Web Navigation using Grammatical Inference

Georgios Korfiatis
School of Electrical and Computer Engineering,
NTUA, Athens, 15780, Greece

Georgios Paliouras
Institute of Informatics and Telecommunications,
NCSR "Demokritos", Athens, 15310, Greece

Abstract

In this paper, a method that models user navigation on the Web, as opposed to a single Web site, is presented, aiming to assist the user by recommending pages. User modeling is done through data mining of Web usage logs, resulting in aggregate, rather than personal models. The proposed approach extends Grammatical Inference methods, by introducing an extra merging criterion, which examines the semantic similarity of automaton states. The experimental results showed that the method does indeed facilitate the modeling of Web navigation, which was not possible with the existing Web usage mining methods. However, a content-based recommendation model is shown to still outperform the proposed method, which suggests that the knowledge of the navigation sequence does not contribute to the recommendation process. This is due to the thematic cohesion of navigation sessions, in comparison to the large thematic diversity of Web usage data. Among three variants of the proposed method, the one based on Blue Fringe, that examines a larger space of possible merges, performs better.

Keywords: machine learning, Web mining, grammatical inference, information retrieval, user modeling.

1 Introduction

The lack of structure and the information overload of the Web make the navigation through it a difficult task. Thus, it is important to assist the user by offering personalized navigation services. At the level of a Web site, personalization consists in various functions, among which the customization of a Web page and the guidance of the user through the Web site by recommending links to possibly interesting pages. For the development of personalized Web site services, techniques from the domain of Data Mining have been employed, which exploit usage data collected in the log files of Web servers.

In contrast to that work, the method presented here intends to model user navigation in the entire Web, aiming at assisting the user by recommending links to interesting pages. For that purpose, usage data of the entire Web are needed. Such data exist, among other places, in log files of proxy servers of Internet Service Providers (ISPs). However, these data are characterized by great thematic diversity, since they reflect the users' navigation in the entire Web, rather than the closed world of a Web site. This fact makes it hard to employ the common techniques which are based only on usage data and are being used for the personalization of Web sites. Addressing this issue, this paper proposes that usage mining techniques can properly be adapted to take into account extra information about the content similarity of Web pages.

At the core of the navigation modeling method proposed in this paper, lie machine learning techniques. Especially, we adopt Grammatical Inference methods for the construction of the model. In this context, we consider that the user navigation can be represented as a regular grammar and more precisely as its probabilistic extension, which takes also into account the probability of a string occurring. Thus, Web pages are considered as symbols of a probabilistic regular grammar and page sequences (usage sessions) as strings of the respective language. The inferred model is then used for page recommendation. In order to select the pages to be recommended to a specific user, this process makes use of the recorded navigation sequence of the user. For the purposes of evaluation, we have exploited usage data from the log files of an ISP and employed a measure that assesses the utility of the list of recommended pages.

The rest of this paper is organized as follows. In section 2 we summarize the major research approaches in the field of navigation pattern discovery. Section 3 describes the Web navigation modeling method that we propose. Section 4 evaluates the method on real usage data from an ISP proxy server. Finally, the main conclusions and open issues are presented in section 5.

2 Related Work

Various Data Mining techniques have recently been developed that model the users' navigation in a Web site, mining Web usage data. A Web usage mining process consists generally of four successive stages (Pierrakos et al. 2003). At the first stage, the usage data, i.e. log of accesses to Web pages, are collected from the source. Then these data pass through a pre-processing stage. In particular, the users as well as the user sessions must be identified. A session is a sequence of pages that a user has visited in sequence. After that, follows the pattern discovery stage, where knowledge is extracted from the data, and at the final stage the extracted knowledge is evaluated and exploited. In the case of Web navigation, pattern discovery methods are employed that take also into account the element of time. These form the research domain of navigation pattern discovery. The proposed techniques can be divided into deterministic and stochastic ones. The former ones attempt to produce patterns that describe the navigational behavior of the user, while the latter construct probabilistic models of the user navigation. Such user models aim usually at recommending relevant links to a user or generally predicting the next pages to be requested.

An example of a deterministic method is presented in (Spiliopoulou et al. 1999), where the Web Utilization Miner (WUM) tool is being used for sequential pattern discovery. The system is based on a special tree-like index of user sessions and an SQL-like query language. The language supports predicates that can be used to specify the content, the structure and the statistics of navigation patterns. WUM provides interactive mining and thus the discovery process is a semi-automated one. In the method proposed in (Paliouras et al. 2000), user sessions are represented by the transitions between pages. By clustering these data, communities are being produced, which correspond to the navigational behavior of users. In (Halvey et al. 2005) clustering is employed to create predictive models that depend on different time periods and goals of the user. Another deterministic approach has been employed by the Clementine tool of SPSS, which uses the sequential pattern discovery algorithm CAPRI (Clementine A-Priori Intervals). This algorithm not only discovers frequent item-sets but also finds the order in which these items have been traversed. Similarly, an algorithm for frequent Web sequences is presented in (Nanopoulos et al. 2001). The algorithm considers the specific characteristics of Web user navigation, that is, the order of dependencies between Web page accesses, the interleaving of requests belonging to a pattern with random ones and the ordering of requests, in order to deduce the next page accesses of a client.

Most of the stochastic sequential pattern discovery methods make use of Markov models to predict the next link the user will choose. In (Bestavros 1995) Markov models are applied to Web usage data. In particular, a first-order hidden Markov model is employed in order to predict the subsequent link that a user might follow within a certain period of time. In (Sarukkai 2000) Markov chains are employed in order to model sequences of Web pages. A similar approach is followed in (Zhu 2001), where additionally the referrer information from the log file is exploited.

Albrecht et al. (1999) present an approach that exploits four different Markov models for predicting Web pages within a hybrid structure named *maxHybrid*. The first model predicts the next link to be followed, based only on the last page that has been requested and is called Time Markov Model. Accordingly, the Second-order Markov Model predicts subsequent links based on the last two pages that have been requested. The Space

Markov Model employs the referring page and finally the Linked Space-Time Markov Model predicts the next user's request by taking into account both the referring page and the last requested. Once a page has been requested, the four Markov models calculate the probability of the next page to be requested and the maxHybrid model chooses the model with the highest probability for prediction.

Another stochastic approach to sequential pattern discovery from user sessions is presented in (Borges and Levene 2000). The sessions are being modeled by means of a so-called hypertext probabilistic grammar (HPG). In the context of HPGs, Web pages are represented by language symbols, links between Web pages are represented by production rules and sequences of Web pages by strings. A directed-graph breadth-first search algorithm is employed in order to identify strings that describe best the users' browsing behavior.

In (Karampatziakis et al. 2004) Grammatical Inference methods were first employed for the modeling of user navigation in a Web site. Representing Web pages as terminal symbols of a probabilistic regular grammar and the sequences of Web pages as strings of the respective language, the method constructs initially a probabilistic prefix tree automaton (PPTA) or a hypertext probabilistic automaton (HPA) from Web usage data. Then the Alergia and the Blue Fringe algorithms are applied on the initial automaton. These algorithms check the state compatibility with respect to the transition probabilities and perform state merges. The final graph can be used to recommend pages to the visitors of a Web site, by modeling the visitor's observed navigation with the learned grammar.

In (Jin et al. 2004), a model based on Probabilistic Latent Semantic Analysis (PLSA) is used for Web usage mining. In particular, this model computes the probabilities of relationships between Web users and tasks (users' objectives), as well as Web pages and tasks. Probabilistic inference is then used in order to discover a variety of usage patterns, enabling, for instance, the identification of users who perform the same task or the characterization of user groups with respect to the tasks they perform. PLSA is also used in (Pierrakos and Paliouras 2005), for the construction of personalized Web Directories, based on usage data from ISP logs. To our knowledge, this is the only effort that uses Web mining to provide personalization on the entire Web, as opposed to a single Web site.

3 Content-aware navigation user modeling with grammatical inference

The majority of the approaches mentioned above address navigation pattern discovery from usage data of a single Web site aiming at offering personalized services for this site. In contrast to these, this paper proposes a method

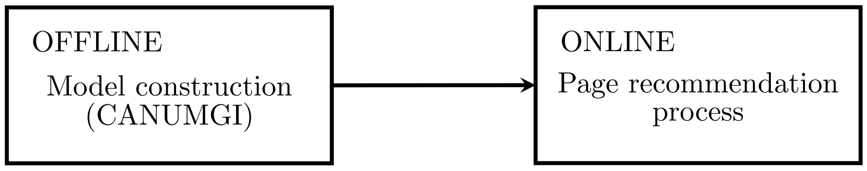

Figure 1: Interaction between the navigation modeling and page recommendation processes.

that models the navigation of the users through the entire Web. This is a much more challenging problem, due to the large volume and the thematic diversity of the Web compared to a single Web site.

The method proposed here to address this problem is called CANUMGI: Content-Aware Navigational User Modeling with Grammatical Inference. As its name suggests, it is based on Grammatical Inference methods, extending them to take into account the content of the Web pages, in addition to the pure usage information modeled in (Karampatziakis et al. 2004). For this purpose, the well-known Grammatical Inference methods Alergia (Carrasco and Oncina 1994) and Blue Fringe (Lang et al. 1998) are being modified, by introducing an extra merging criterion, which checks the content similarity of Web pages. Modeling takes place off-line and constructs a stochastic finite automaton (SFA). This automaton is then used on-line for the recommendation of links to users (Fig. 1).

In order to construct the user model, the following procedure is being employed. Web usage data, taken from proxy log files, e.g. of an ISP, are used as input to the process. These data consist of sequences of pages that the users-clients of the ISP have visited within a certain time span. These recorded page sequences are separated into user sessions. Furthermore, the pages are considered as symbols of a probabilistic regular grammar and the sequences of pages (user sessions) as strings of the respective language. Initially, the method uses the training strings (usage data) to construct a probabilistic prefix tree automaton (PPTA), such that each string (user session) corresponds to a path on the tree. If the original versions of Grammatical Inference methods, such as Alergia or Blue Fringe, were applied to this data, almost no generalization would take place. Due to the large volume of the Web which results in the thematic diversity of the usage data, since the compatibility between two states is determined by the transition similarity, most of the PPTA states would result to be incompatible and therefore almost no merges would be possible.

In order to overcome this problem, the compatibility metric is modified to take into account the semantic similarity of automaton states. This new measure requires the existence of additional information that describes the content of the pages in a state. For this purpose, we assume that each page is represented by a vector of characteristic keywords, which have been extracted from the pages in a pre-processing stage. Similarly, a state representing a cluster of pages, which can result from a series of state merges, is represented by the respective keyword vector $\underset{\sim}{x}$. Each keyword-component takes real values in the range $[0, 1]$ and expresses the proportion of pages of the cluster which contain the respective keyword. This vector can be considered to express the center of gravity of the pages that comprise the state (cluster). In fact, this is a TF (term frequency) representation with the assumption that keywords appear only once in (initial) pages. However, we decided against IDF (inverse document frequency) normalization, because we consider here clusters rather than plain documents. Moreover, since we seek a descriptive rather than discriminative cluster representation, penalizing keywords that appear in many clusters is not desired. Based on this representation, two states are similar with respect to their content, if the value of the cosine metric (Eq. 1) between the respective vectors is greater than a predefined threshold value.

$$\cos\left(\underset{\sim}{x}, \underset{\sim}{y}\right) = \frac{\sum_i x_i y_i}{\sqrt{\left(\sum_i x_i^2\right)\left(\sum_i y_i^2\right)}} \tag{1}$$

Therefore, the CANUMGI method combines the usage metric (transition similarity) and the content metric (metric of similarity between the vectors of two states) when calculating the compatibility of two states. In this work, three versions of the method have been implemented. The first version (CANUMGI-A) extends the Alergia algorithm, the second version (CANUMGI-B) extends the Blue Fringe algorithm and the third version (CANUMGI-C) employs a dimensionality reduction technique before applying the inductive methods.

3.1 CANUMGI-A

This method follows the basic procedure of the Alergia algorithm (Alg. 1). Starting with the PPTA, it checks the compatibility of the states (Alg. 2). As in the original Alergia (Carrasco and Oncina 1994), the state similarity formula used in the usage part of the compatibility criterion results from the Hoeffding boundary (Hoeffding 1963), which expresses the confidence range for a Bernoulli variable with probability h and observed frequency $C(\#)$ out of C, with probability larger than $(1-\alpha)$:

$$\left|h - \frac{C(\#)}{C}\right| < \sqrt{\frac{1}{2C}\ln\frac{2}{\alpha}} \tag{2}$$

Two states q and q' are considered different with respect to their usage, if it holds:

$$\left|\frac{C(q,\#)}{C(q)} - \frac{C(q',\#)}{C(q')}\right| > \sqrt{\frac{1}{2}\ln\frac{2}{\alpha}}\left(\frac{1}{\sqrt{C(q)}} + \frac{1}{\sqrt{C(q')}}\right) \tag{3}$$

For each state q, $C(q)$ is the number of strings that arrive at this state and $C(q, \#)$ the number of strings that end at this state. Solving Eq. 3 with respect to the parameter α, we get the p-value of the statistic measure:

$$p = 2\exp\left(-2\left(\frac{C(q,\#)\,C(q') - C(q',\#)\,C(q)}{C(q')\sqrt{C(q)} + C(q)\sqrt{C(q')}}\right)\right) < \alpha \qquad (4)$$

Thus, the two states are considered different, if their p-value is less than the user-defined threshold α, taking values in the range $[0, 2]$.

Algorithm 1 The Alergia grammatical inference method

```
Input:  $S^+$: Set of positive examples
        $\alpha$: 1 - confidence level
Output: Stochastic DFA
1: $A$ = PPTA from $S^+$
2: for $j$ = successor(firststate($A$)) to laststate($A$) do
3:   for $i$ = firststate($A$) to $j$ do
4:     if compatible($q_i$, $q_j$, $\alpha$) then
5:       merge($A$, $q_i$, $q_j$)
6:       break inner loop
7:     end if
8:   end for
9: end for
```

Line 1 in Alg. 2 concerns the calculation of the content similarity (cosine metric). Two states are considered to be compatible with respect to the content metric, if their similarity value is greater than the respective predefined threshold. Finally, the two metrics are combined either by disjunction or conjunction, as shown in line 9. Considering here the min of p-values as a measure of usage compatibility corresponds to the classical Alergia demanding that the two states be similar with respect to all their transitions.

When merging two states, the resulting state represents a cluster of pages; the union of the sets of pages in the two states. For the purposes of content similarity, vector $\underset{\sim}{x}$ as defined above is implemented as follows: each state is represented by a content vector $\underset{\sim}{v}$ of integer values that express the number of pages in the state-cluster that contain the respective keyword. Using this vector $\underset{\sim}{v}$ and the number k of pages in the cluster, $\underset{\sim}{x}$ can be calculated as $\underset{\sim}{v}/k$. The merging procedure follows that of Alergia (Alg. 3). Lines 4 and 5 extend the basic procedure, in order to calculate the content vector of the new state. It should be noted that, when merging two states, their children that correspond to the same symbol are also being merged recursively.

Algorithm 2 Compatibility check in the CANUMGI-A method (Additions to classical Alergia highlighted)

```
compatible($q_i$, $q_j$, $\alpha$)
Input:  $q_i$, $q_j$: states
        $\alpha$: usage metric threshold
        $\theta$: content metric threshold
Output: True, if two states compatible
 1: calculate similarity($q_i$, $q_j$)
 2: calculate pvalue($C(q_i)$, $C(q_j)$, $C(q_i, \#)$, $C((q_j, \#))$
 3: for all $a \in \Sigma$ do
 4:    calculate pvalue($C(q_i)$, $C(q_j)$, $C(q_i, a)$, $C((q_j, a))$
 5:    if not compatible($\delta(q_i, a)$, $\delta(q_j, a)$, $\alpha$) then
 6:       return false
 7:    end if
 8: end for
 9: if similarity $> \theta$ or/and min(pvalues) $> \alpha$ then
10:    return true
11: end if
12: return false
```

Algorithm 3 Merging procedure in the CANUMGI-A method (Additions to classical Alergia highlighted)

```
merge($A$, $q_i$, $q_j$)
Input:  $A$: the PPTA
        $q_i$, $q_j$: states to be merged
 1: $A = A - \{q_i, q_j\}$
 2: $A = A \cup q'$
 3: ID of $q'$ = min(IDs of $q_i$, $q_j$)
 4: vector($q'$) = vector($q_i$) + vector($q_j$)
 5: $k' = k_i + k_j$
 6: $C(q') = C(q_i) + C(q_j)$
 7: $C(q', \#) = C(q_i, \#) + C(q_j, \#)$
 8: for all $a \in \Sigma$ do
 9:    $C(q', a) = C(q_i, a) + C(q_j, a)$
10:    merge($A$, $\delta(q_i, a)$, $\delta(q_j, a)$)
11: end for
```

3.2 CANUMGI-B

However, Alergia chooses state merges in a brittle manner. In particular, it chooses to merge at each iteration the first encountered compatible pair of states, ignoring the possibility of other more compatible pairs. Furthermore, a merged state takes precedence in the checking process and therefore it is probable that it merges again with another state. As a result, the algorithm favors the creation of a single state that contains the majority of the pages, destroying thus the structure of the graph. Blue Fringe, which is an enhancement of Alergia, treats this problem more effectively, since at each merging step, it evaluates many candidate state pairs and chooses the best pair in a greedy search manner. In particular, it maintains two sets of red (examined incompatible states) and blue (children of red) states and chooses to merge the red-blue pair that achieves the highest compatibility score. The second version of the CANUMGI method is based on Blue Fringe and it is examined together with the Alergia-based CANUMGI-A method for comparison reasons. The main procedure of CANUMGI-B is identical to that of the Blue Fringe algorithm (Alg. 4). The method that rates the merges (Alg. 5) is analogous to that of CANUMGI-A; the difference is that the new algorithm calculates an arithmetic instead of a boolean compatibility value. This value expresses how compatible the two states are, in both usage and content terms. The value that the method returns can be any function of the two metrics. We consider three functions (denoted as f in line 9 of Alg. 5) for that purpose here:

- max value
- min value
- weighted sum $(1-w) \cdot a + w \cdot b$, where $w \in [0,1]$

The first case corresponds to disjunction as used in CANUMGI-A and the second to conjunction. The third case offers a way to parameterize the weight we assign to each metric.

In order to apply one of these functions, it is important that the values of the two metrics are normalized so as to be comparable. For the purposes of normalization, we have computed the distributions of values for the two metrics for all pairs of states in the initial PPTA. The normalization process commences with the calculation of a normalization factor:

- For the two distributions, isolate the values that are higher than the respective threshold.
- Subtract from these values the respective threshold.
- Compute the respective mean values (*usage_mean* and *content_mean*).

Algorithm 4 The Blue Fringe grammatical inference method

```
Input: A PPTA
Output: A deterministic SFA
 1: set the initial state of the PPTA red
 2: set the children of the initial state blue
 3: set the other states white
 4: while exists a blue state do
 5:    score all red/blue merges
 6:    if exist blue states incompatible with all red states then
 7:       turn the nearest to the initial state among those states to red
 8:       set the white children of this state blue
 9:    else
10:       merge the red/blue pair with the highest score
11:       set the white children of the new state blue
12:    end if
13: end while
```

Algorithm 5 Merge score in the CANUMGI-B method (Additions to Blue Fringe highlighted)

```
mergeScore($q_i$, $q_j$, $\alpha$, $\theta$, $nf$)
Input:  $q_i$, $q_j$: states
        $\alpha$: usage metric threshold
        $\theta$: content metric threshold
        $nf$: normalization factor
Output: Real value
 1: calculate similarity($q_i$, $q_j$)
 2: calculate pvalue($C(q_i)$, $C(q_j)$, $C(q_i, \#)$, $C((q_j, \#))$
 3: for all $a \in \Sigma$ do
 4:    calculate pvalue($C(q_i)$, $C(q_j)$, $C(q_i, a)$, $C((q_j, a))$
 5:    if mergeScore($\delta(q_i, a)$, $\delta(q_j, a)$, $\alpha$, $\theta$, $nf$) <= 0 then
 6:       return 0
 7:    end if
 8: end for
 9: return f(similarity$-\theta$, $nf \cdot$(min(pvalues)$-\alpha$)))
```

- Compute the normalization factor:

$$nf = content\_mean/usage\_mean \qquad (5)$$

While checking the compatibility of a pair of states, the usage metric value (UMV) and content metric value (CMV) are computed for the pair. The two values are made comparable as follows:

- $CMV - content\_threshold$
- $nf \cdot (UMV - usage\_threshold)$

Finally state merging in CANUMGI-B is done as in Blue Fringe (Alg. 6). Similar to Alg. 3 for CANUMGI-A, lines 3 and 4 calculate the content vector of the merged state.

Algorithm 6 Merging procedure in the CANUMGI-B method (Additions to Blue Fringe highlighted)

```
merge($A$, $q_i$, $q_j$)
Input:  $A$: The PPTA
        $q_i$, $q_j$: states to be merged
 1: $A = A - \{q_i, q_j\}$
 2: $A = A \cup q'$
 3: vector($q'$) = vector($q_i$) + vector($q_j$)
 4: $k' = k_i + k_j$
 5: color of $q'$ = RED
 6: if $q_i$ is RED and $q_j$ is WHITE then
 7:     paint the white children of $q'$ blue
 8: end if
 9: $C(q') = C(q_i) + C(q_j)$
10: $C(q', \#) = C(q_i, \#) + C(q_j, \#)$
11: for all $a \in \Sigma$ do
12:     $C(q', a) = C(q_i, a) + C(q_j, a)$
13:     merge($A$, $\delta(q_i, a)$, $\delta(q_j, a)$)
14: end for
```

3.3 CANUMGI-C

As already mentioned above, the large thematic diversity of the Web usage data is a major problem of the navigation pattern discovery procedure. The third version of CANUMGI attempts to cope with this problem by employing a dimensionality reduction pre-processing step. In particular, the pages that belong to the training set are initially being clustered based on their content. For this purpose, the partitioning method k-means is employed here, but any other crisp clustering method could also take its place. CANUMGI-C is motivated by the intuition that the pages of a certain category appear in similar navigation patterns. Thus, it constructs the PPTA from the usage data, using as basic states the page clusters. The transitions on the PPTA also correspond to transitions between thematic categories, rather than Web pages. Based on this reduced PPTA, the induction of the target automaton can be done with any of the other two versions (CANUMGI-A and CANUMGI-B).

3.4 Using the grammatical user model for personalized navigation

The model constructed with the CANUMGI method can be used to recommend pages to users. The process of choosing the pages to be recommended is divided into two stages. In the first stage, the users' observed navigation thus far is used to arrive at a state of the automaton. The state transition process (Alg. 7) is less strict than it is for the common DFAs, for it is unlikely that an explicit transition between two specific pages will be observed frequently, due to the large volume of the Web. Thus, if there is no explicit transition to a state containing the observed page, the transition to the most similar child-state, based on the page's content, is followed. A threshold is set on this similarity, so that, if no child-state is similar enough, then the method returns to the start state of the automaton. In the case of dimensionality reduction (CANUMGI-C), the process is slightly different. Here the transitions represent page clusters instead of separate pages. Thus Web pages must be classified first to the existing clusters before applying the normal state transition process. The classification of pages is done as shown in Alg. 8.

In the second stage, the pages to be recommended are chosen from the child-states of the state reached in stage 1. These pages are selected according to an evaluation criterion, which is based on two assumptions: (a) a high-probability transition leads to a cluster of pages that are more suitable for recommendation to a user, and (b) in order to choose pages from a certain state (cluster), we assume that a page closer to the center of gravity is better for recommendation. In particular, two ways to choose pages have been examined in order to construct a recommendation list of predefined size n:

- Coarse approach: Find the child-state with the highest transition probability. Choose the pages which are closer to the center of gravity of the cluster, using the cosine metric. If the pages of the first state are not sufficient, continue with the next most-probable states until a list of length n is constructed.

- Fine approach: Sort all pages w of all children c of the current state

Algorithm 7 State transition process of the page recommendation process

```
transition($A$, $q$, $p$, $thres$)
Input:  $A$: the automaton
        $q$: current state
        $p$: page / input symbol
        $thres$: threshold
 1: if a transition for $p$ exists then
 2:    goto the target state of the transition
 3: else
 4:    find the state-child of $q$ which is more similar to $p$ with respect to
       content
 5:    if similarity value $> thres$ then
 6:       goto this state
 7:    else
 8:       goto the start state of the automaton
 9:    end if
10: end if
```

Algorithm 8 Page classification of the page recommendation process in the case of CANUMGI-C

```
classifyPage($p$, $C$)
Input:  $p$: the page to be examined
        $C$: the existing clustering of pages
Output: a cluster
 1: if $p$ already exists in a cluster of $C$ then
 2:    return this cluster
 3: else
 4:    compute the cosine metric between page $p$ and all clusters of $C$
 5:    return the cluster with the highest value
 6: end if
```

based on the product, p (Eq. 6), of the probability of the transition to state c times the value of cosine metric between the page w and the center of gravity of c. Choose the best n pages.

$$p = transProb\,(c) \cdot similarity\,(w, c) \tag{6}$$

It should be noted that the model can only recommend pages that have already appeared in the training set. The recommendation set gets richer the more users are involved in the system and the longer the examined period of time is. Therefore, the process is a dynamic one.

4 Experimental Evaluation

4.1 Evaluation process and measure

For the experimental evaluation of the new method, usage data from the log files of an ISP have been used. In a pre-processing stage, data cleaning has been performed and the remaining 12932 recorded Web page accesses have been separated to 1468 user sessions. For this purpose, distinct users were identified by their IP addresses and a silence threshold of 60 minutes was used for separating different sessions. Moreover, the keywords that characterize the 7214 unique pages have been selected. The page content information is denoted by a keyword vector of size 5086. Following common practice in machine learning evaluation, the set of user sessions was split randomly into two parts: the first one (983 sessions) was used for training the models and the second one (485 sessions) for testing them. In each test session, all pages but the last one are observed and the hidden last page is compared to the recommended pages.

Evaluating the proposed method has been far from straightforward. Standard measures, such as perplexity, would not be sufficient here. In particular, a measure that checks the coverage of the constructed graph would not work in this case, due to the diversity of the input data. Moreover, we would like to emphasize on the utility of the recommended pages, rather than the graph structure in itself. For this purpose, the notion of expected utility was employed. This is proposed in the collaborative filtering literature (Breese et al. 1998) for the evaluation of a recommendation list and has been used by many researchers for this purpose. The measure estimates how useful a list of ranked items can be to a user. This estimate is provided by the probability of viewing each item of the recommendation list times its utility. In particular, the expected utility formula defined in (Breese et al. 1998) was here modified to incorporate the content similarity of each recommended page a_j to a target page w (Eq. 7). The probability to choose a certain recommendation is still assumed to decrease exponentially with its position on the recommendation list.

$$EU_w = \sum_{j=0}^{n-1} \frac{similarity\,(w, a_j)}{2^{j/h}} \tag{7}$$

In the above formula, the predefined parameter h corresponds the viewing half-life, namely the ordinal number of the page in the recommendation list which has 50% chance to be requested, and is set to 5, following common practice (Breese et al. 1998). Finally, for a set of user sessions with the respective recommendation lists, the expected utility is computed as follows:

$$EU = 100 \frac{\sum_w EU_w}{\sum_w EU_w^{max}} \tag{8}$$

where EU_w^{max} is the maximum utility that can be achieved for a certain list and the respective target page:

$$EU_w^{max} = \sum_{j=0}^{n-1} \frac{1}{2^{j/h}} \tag{9}$$

For the purposes of comparison, a simple content-based recommendation model has also been constructed. This model recommends the pages that are on average more similar to the n last observed pages of a user session for a given n. Similar to the case of CANUMGI, the performance of this content-only model is assessed by means of the expected utility measure for all sessions of the testing sample.

4.2 Specification of parameters

Using the ISP data presented above, we evaluated the proposed navigation modeling and recommendation methods on the test set, along several dimensions determined by respective parameters of the methods. The results of this evaluation are discussed briefly in the following paragraphs, starting from the end of the recommendation process and moving backwards to the navigation modeling method.

4.2.1 Approaches to page selection

In the page selection phase of the recommendation process, we examine two different approaches to selecting pages for the recommendation list. The experiments have shown a clear advantage of the coarse approach of choosing pages only from the best child-state, rather than the fine one that scores all pages of all children. The former achieved consistently 10 to 25 percentage points higher expected utility than the latter. This is due to the fact that most of the states of the induced automaton have a self-transition of high probability (higher than 50%). That is, page clusters tend to be constructed, that represent long sequences of page transitions. As a result, the "best" child of a state, in terms of transition probability, is itself and the pages that are close to its center of gravity are usually quite close to the hidden page. The results presented in the rest of this section follow the coarse page selection approach.

4.2.2 Transition process threshold

In the state transition phase of the recommendation process, a threshold is used to determine whether a transition will be followed or the process will return to the start state. Fig. 2 shows the effect of this parameter on the expected utility. Curves E1 and E2 correspond to two representative experiments with CANUMGI-B and curve E3 with CANUMGI-C (using Blue Fringe for induction). These three cases have been chosen as the best results among the many parameter configurations that were tested. In all cases it is observed that the expected utility increases as the transition threshold increases, up to a threshold value in the range of 0.3 to 0.5 and then starts to decrease. A higher threshold value leads to more "restarts" in the test sessions and therefore transitions to totally dissimilar states are avoided. However, if the threshold is set to a too high value, then the process degenerates to a situation where only the start state and its children are used in the recommendation process. Finally, since a higher threshold value means more "restarts", the fact that the graphs with a threshold value around 0.3, which is already quite high (i.e., it causes several restarts), give the best results suggests that the knowledge of only the last few visits of a user is more useful than the knowledge of the user's long-term behavior.

4.2.3 Usage metric threshold

In order to select an appropriate threshold value for the usage metric (which can take values from 0.0 to 2.0), the distribution of values for this metric for all pairs of states in the initial PPTA was examined. It turned out that there is a big spike (94% of cases) at the value 1.21306, which corresponds to the case where the two states that are compared appear in only one user session, and they have a single, distinct transition, as in Fig. 3. These cases that correspond to extremely rare and dissimilar patterns, clearly should not lead to a merge. Therefore, the threshold must be set higher than this value, but not too high, because the cases that remain are relatively few and no merging due to usage similarity would be possible. For that reason, the usage metric threshold is set to the value 1.22. In the case of dimensionality reduction (CANUMGI-C), the distribution depends also on the number of the initial clusters, in practice though, the distribution exhibits the same features as for the other methods and thus the same threshold value is set.

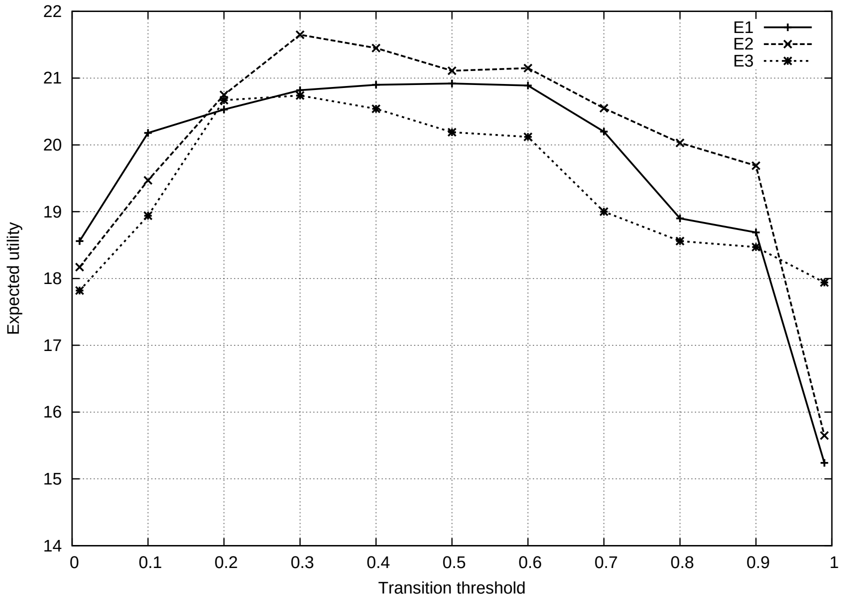

Figure 2: Study of the transition threshold for CANUMGI-B (E1, E2) and CANUMGI-C (E3). E1 with content metric threshold 0.01 and criteria combination parameter (weight for usage metric) 0.7, E2 with 0.1 and 0.4 respectively, E3 with 0.1 and 0.5 respectively and initial clustering into 2500 clusters.

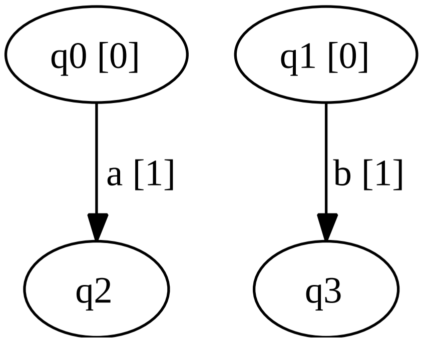

Figure 3: An example of two states (q_0 and q_1) that appear in only one session each and lead to different states in different ways.

It should be noted that the choice of threshold value in the above-described manner is no more than an educated guess and thus suboptimal, since the distribution continuously changes due to the merges.

4.2.4 Content metric threshold

In order to select an appropriate threshold for the content metric (range 0.0 to 1.0), the distribution of its values for all pairs of PPTA states was examined, as in the case of the usage metric. Due to the diversity of the usage data, most of the pairs of states are rather dissimilar with respect to the page content and therefore there is a big spike at 0 (around 80%). However, it is speculated that during the model construction process the distribution is getting smoother, for, in the case of measuring the similarity of a pair of page clusters (instead of a pair of simple pages as in the PPTA), the probability that they have at least one common keyword is higher, resulting in a non-zero value of the cosine metric. Fig. 4 shows how the content metric threshold affects the size of the final automaton constructed by CANUMGI-B, as well as the corresponding expected utility. The size of the final automaton increases as the threshold increases; this is expected, since the criterion becomes in this way stricter and thus the number of possible merges decreases. Accordingly, the expected utility seems to decrease for threshold values above 0.1, because the graph is getting so big that it becomes unsuitable for a sequential pattern discovery process. The best results are obtained by automata whose size is about an order of magnitude smaller than the initial PPTA (which is 7758 states in this case). Finally, it seems that the stricter threshold value 0.1 leads to better results than the value 0.01, as it avoids the merging of states that exhibit minimal similarity.

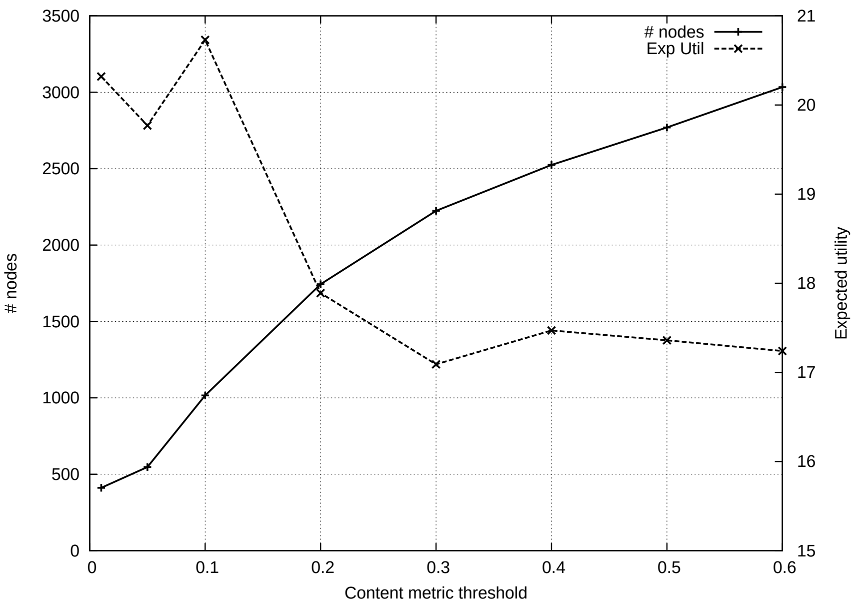

Figure 4: Study of the content metric threshold for CANUMGI-B.

4.2.5 Combination of usage and content criteria

In the case of CANUMGI-A, based on Alergia, the two metrics can be combined either by conjunction or disjunction. As expected, an automaton constructed by a conjunctive compatibility metric has a significantly greater size, since fewer states are considered compatible and thus mergeable. For instance, in the case of content threshold value 0.1, the induced automaton using conjunction has 2563 states, while the one using disjunction only 9 states.

In the case of CANUMGI-B, the two metrics are numeric and are thus combined arithmetically, rather than logically. As already mentioned above, the two metrics are either combined with the min and max functions, which have the same effect as conjunction and disjunction in CANUMGI-A, or they are combined with the weighted sum function. Fig. 5 shows how the increase in the contribution of the usage criterion (usage weight) affects the size and the expected utility of the induced automaton. It is observed that the size of the induced automaton increases as the weight of the usage criterion increases. This means that as the contribution of the content metric increases, more merges become possible, which is due to the fact that, for the given content similarity thresholds, more pairs of pages are found compatible due to content rather than usage similarity. Regarding the expected utility, it turns out that the best performance is obtained for a usage weight in the range around 0.5.

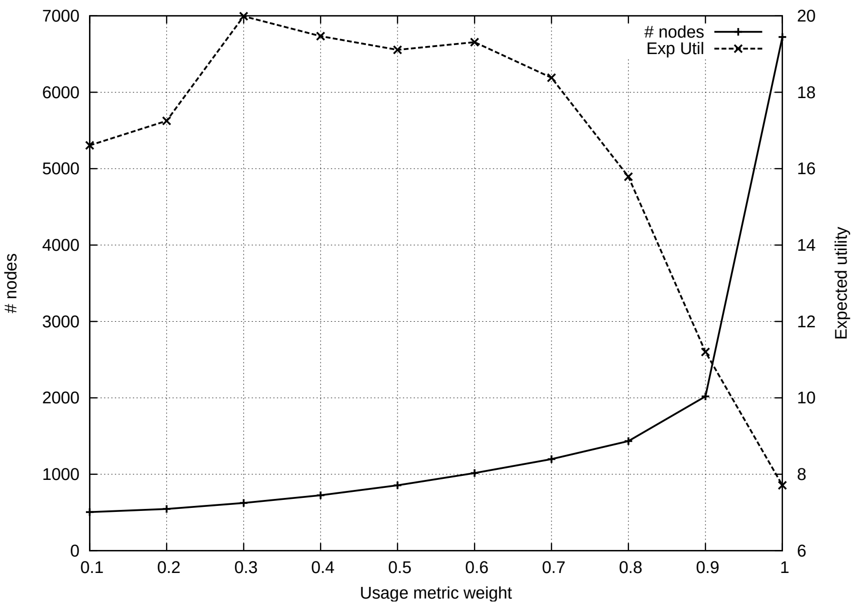

Figure 5: Study of criteria combination. Results of the CANUMGI-B method, for content metric threshold 0.1

4.2.6 Dimensionality reduction

In the case of CANUMGI-C, the number of clusters that are initially created affects significantly the performance of the induced model. Experiments with various clusterings have shown that a high level of dimensionality reduction into a small number of clusters reduces the discriminating ability of the model, instead of strengthening the process by highlighting latent similarities. This is intensified by the nature of the usage data that causes clustering to create one big cluster with many pages as well as many clusters with only one page (singletons). As a result, a high number of initial clusters (around 2500) results in higher expected utility values. Nevertheless, none of the experiments with dimensionality reduction exhibited better performance than the one obtained without any reduction.

4.3 Comparison of the different methods

Figure 6 summarizes the best results achieved by the methods that we examined. Out of the three proposed variants of the CANUMGI method, CANUMGI-A performed worst. The best expected utility value obtained by this method was 8.57, by setting the content metric threshold to 0.05, using disjunctive combination of the two metrics and setting the transition threshold to 0.2. The low performance of the method is due to the structure of the induced automaton, owing to the way Alergia chooses the states to

Method	**max Exp. Utility**
CANUMGI-A	8.57
CANUMGI-B	21.65
CANUMGI-C	20.74
Content-based model	24.85

Figure 6: Best results of the examined methods

merge.

The results of the CANUMGI-B method were much better. The induced automaton is more balanced and its performance is in all cases better than that of CANUMGI-A. The best expected utility value was 21.65 and was obtained by setting the content metric threshold to 0.1, the weight of the usage criterion to 0.4 and the state transition threshold to about 0.3.

Regarding the CANUMGI-C method, which employs the dimensionality reduction pre-processing step, we used only Blue Fringe as the induction method and not Alergia, due to the bad results of CANUMGI-A which result from the merging strategy of Alergia. However, CANUMGI-C did not perform as well as CANUMGI-B. Particularly when the pages are initially clustered in a small number of clusters, the expected utility is very low. However, for a larger number of initial clusters, the results were comparable to those of CANUMGI-B. It should be noted though that the more initial clusters the more CANUMGI-C (using Blue Fringe) degenerates to CANUMGI-B. The best expected utility value was 20.74 and was obtained by setting the content metric threshold to 0.1, the usage metric weight to 0.5, the state transition threshold to 0.3 and starting with 2500 initial clusters.

In order to check the statistical significance of the difference between the best results of CANUMGI-B and CANUMGI-C we did a 10-fold cross validation, considering the parameter values that yield the best results for each case above. The average expected utility achieved for CANUMGI-B was 21.03, while for CANUMGI-C was 20.17. Standard deviation was 4.5 and 4.8 respectively. The t-test value computed was 0.053, thus the difference of the two average values is not cosidered significant. However, since no significant fluctuations around the best values are observed for different parameter values nor any peaks, we are quite confident as far as the performance of the methods and their ranking are concerned.

Finally, none of the three CANUMGI variants obtained higher expected utility than that of the model based on content similarity only. The content-only model obtained expected utility of 24.85, which was achieved in the case of comparing with only the last observed page.

5 Conclusions and open issues

We have presented a method that mines Web usage data to produce models of Web user navigation and then use these models to recommend potentially interesting Web pages to the users. The proposed method, called CANUMGI (Content-Aware Navigational User Modeling with Grammatical Inference), extends the Grammatical Inference methods Alergia and Blue Fringe by introducing content-similarity ideas from Information Retrieval. In this manner, it is able to address the problem of thematic diversity of Web usage data, which makes common usage mining methods inapplicable. Furthermore, we attempted employing a dimensionality reduction preprocessing step, in order to enhance the thematic coherence of the induced model. The methods were evaluated on real usage data, collected at the proxy server of an Internet Service Provider. The evaluation was based on a measure of expected utility for the user, introduced in collaborative filtering literature. Finally, the proposed methods were compared against a model that recommends Web pages, based solely on content similarity.

The use of the CANUMGI method on real data has shown that the method does indeed allow the modeling of user behavior on the Web, which was not possible with the existing Web usage mining methods, including the ones based on Grammatical Inference. However, none of the three versions of CANUMGI succeeded in performing better than a content-based recommendation model. This probably means that the knowledge of the order in which a user visits Web pages does not contribute to the page recommendation process, contrary to intuition based on research for navigation modeling in a single Web site. The great diversity of the Web usage data is the main cause of this phenomenon. Indeed, it was observed that state merges during the inductive process are actually being based more on content similarity than on usage similarity, resulting in the construction of large clusters of pages that are similar in content. It seems in general that the navigation of a user through the Web, within a particular session, is mainly restricted to a set of pages of a single thematic category, while it is difficult for the few transitions to other thematic categories to be predicted. This fact is being experimentally confirmed by the structure of the induced graph, where self-transitions have much higher probability than inter-state transitions, while the latter are also governed by a degree of randomness.

Concerning the three variants of the proposed method, it turned out that CANUMGI-B, based on the Blue Fringe algorithm, performs better than CANUMGI-A, and close to the content-based system. This is due to the fact that Alergia, on which CANUMGI-A is based, chooses the states to merge in a very greedy manner. As a result, the creation of a single state-cluster of too many Web pages is favored and thus the structure of the automaton is destroyed. In contrast to that, Blue Fringe chooses merges in a more intelligent way. Finally, the dimensionality reduction technique, which is used in CANUMGI-C, did not contribute further to the page recommendation process, meaning that the merging taking place in CANUMGI provides a more flexible clustering of the pages, than using k-means for pre-processing.

During the experimental evaluation process, various parameters of the method were examined, regarding the metric thresholds, the combination of the metrics, the page recommendation process as well as the dimensionality reduction technique. It turned out that many parameters can be predefined, while for the rest of them the range of possible values can be restricted significantly, based on an analysis of the training data.

By studying the effect of the transition process threshold, it turned out that the knowledge of only the last few visits of a user is more useful than the knowledge of the user's long-term behavior. This confirms the main conclusion expressed above, but also suggests that a better modeling, which makes use of the usage data more selectively, while combining them with a default thematic model based on the content of the pages, might provide better results. Especially, we could attempt a smarter use of clustering for thematic categorization, as done in (Pierrakos and Paliouras 2005), where Web directories are used as a basis for this characterization.

Further work would also require the reconsideration of design choices made here, such as the choice of similarity measure. Cosine has been chosen as a simple and effective content similarity measure, but there are many alternative ones that could replace it. Furthermore, in a Semantic Web scenario, feature vectors could be replaced by meta-data annotations that would allow the use of a semantic similarity measure. As far as evaluation is concerned, it would also be interesting to perform an experiment with end-users in order to find out whether the recommended pages are actually useful to humans browsing the Web.

Finally, the main conclusion of the paper, regarding the thematic bias of Web user navigation could be strengthened by testing it against other sequential pattern discovery methods, such as Hidden Markov Models, which would need to be extended accordingly to deal with the problem of thematic diversity of the data.

Acknowledgements

We would like to thank Stasinos Konstantopoulos and Nikos Karampatziakis for useful discussions, as well as Prof. Timoleon Sellis for his support.

References

Albrecht, D. W., I. Zukerman, and A. E. Nicholson (1999). Pre-sending documents on the WWW: A comparative study. In *Proceedings of the*

6th International Joint Conference on Artificial Intelligence (IJCAI), Stockholm, pp. 1274–1279.

Bestavros, A. (1995). Using speculation to reduce server load and service time on the WWW. In *Proceedings of the 4th ACM International Conference on Information and Knowledge Management*, Baltimore, MD, pp. 403–410.

Borges, J. and M. Levene (2000). Data mining of user navigation patterns. In *WEBKDD '99: Revised Papers from the International Workshop on Web Usage Analysis and User Profiling*, London, UK, pp. 92–111. Springer-Verlag.

Breese, J. S., D. Heckerman, and C. Kadie (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pp. 43–52.

Carrasco, R. and J. Oncina (1994). Learning stochastic regular grammars by means of a state merging method. In *Proc. 2nd International Colloquium on Grammatical Inference - ICGI '94*, Volume 862, pp. 139–150. Springer-Verlag.

Halvey, M., M. T. Keane, and B. Smyth (2005). Birds of a feather surf together: Using clustering methods to improve navigation prediction from internet log files. In P. Perner and A. Imiya (Eds.), *MLDM*, Volume 3587 of *Lecture Notes in Computer Science*, pp. 174–183. Springer.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association 58*(301), 13–30.

Jin, X., Y. Zhou, and B. Mobasher (2004). Web usage mining based on probabilistic latent semantic analysis. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, pp. 197–205. ACM Press.

Karampatziakis, N., G. Paliouras, D. Pierrakos, and P. Stamatopoulos (2004). Navigation pattern discovery using grammatical inference. In Y. S. G. Paliouras (Ed.), *ICGI*, Volume 3264 of *LNCS*, pp. 187–198. Springer Verlag.

Lang, K. J., B. A. Pearlmutter, and R. A. Price (1998). Results of the abbadingo one dfa learning competition and a new evidence-driven state merging algorithm. In *ICGI '98: Proceedings of the 4th International Colloquium on Grammatical Inference*, London, UK, pp. 1–12. Springer-Verlag.

Nanopoulos, A., D. Katsaros, and Y. Manolopoulos (2001). Effective prediction of web-user accesses: a data mining approach. In *Proceedings* *of the WEBKDD Workshop, San Fransisco, CA.*

Paliouras, G., C. Papatheodorou, V. Karkaletsis, and C. D. Spyropoulos (2000). Clustering the users of large web sites into communities. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 719–726. Morgan Kaufmann Publishers Inc.

Pierrakos, D. and G. Paliouras (2005). Exploiting probabilistic latent information for the construction of community web directories. In L. Ardissono, P. Brna, and A. Mitrovic (Eds.), *User Modeling*, Volume 3538 of *Lecture Notes in Computer Science*, pp. 89–98. Springer.

Pierrakos, D., G. Paliouras, C. Papatheodorou, and C. D. Spyropoulos (2003). Web usage mining as a tool for personalization: A survey. *User Modeling and User-Adapted Interaction 13*(4), 311–372.

Sarukkai, R. R. (2000). Link prediction and path analysis using markov chains. In *Proceedings of the 9th international World Wide Web conference on Computer networks*, Amsterdam, The Netherlands, pp. 377–386. North-Holland Publishing Co.

Spiliopoulou, M., L. C. Faulstich, and K. Winkler (1999). A data miner analyzing the navigational behaviour of web users. In *Proceedings of the Workshop on Machine Learning in User Modeling of the ACAI99, Chania, Greece*, pp. 54–64.

Zhu, J. (2001). Using markov chains for structural link prediction in adaptive web sites. In *UM '01: Proceedings of the 8th International Conference on User Modeling 2001*, London, UK, pp. 298–302. Springer-Verlag.