Navigation pattern discovery using grammatical inference

Nikolaos Karampatziakis[1], Georgios Paliouras[2], Dimitrios Pierrakos[2], and Panagiotis Stamatopoulos[1]

[1] Department of Informatics and Telecommunications, University of Athens, Greece
[2] Institute of Informatics and Telecommunications, NCSR "Demokritos", Greece

Abstract. We present a method for modeling user navigation on a web site using grammatical inference of stochastic regular grammars. With this method we achieve better models than the previously used first order Markov chains, in terms of predictive accuracy and utility of recommendations. In order to obtain comparable results, we apply the same grammatical inference algorithms on Markov chains, modeled as probabilistic automata. The automata induced in this way perform better than the original Markov chains, as models for user navigation, but they are considerably inferior to the automata induced by the traditional grammatical inference methods. The evaluation of our method was based on two web usage data sets from two very dissimilar web sites. It consisted in producing, for each user, a personalized list of recommendations and then measuring its recall and expected utility.

1 Introduction

In this work we are concerned with modeling the behavior of users on the web. We are specifically interested in discovering interesting navigation patterns by means of grammatical inference methods. When a user requests a page from a specific web site, this transaction is recorded in various places like the browser's history, the web server log file and intermediary devices such as proxies. These transactions are generally called web usage data and the research discipline whose purpose is the discovery of patterns in these data is called Web Usage Mining. In our scenario we examine web usage data in the form of a web server log file. From these data we try to discover interesting patterns by assuming that the user's navigation is governed by the rules of an unknown formal language. To infer the language's grammar, each sequence of web pages that a user requests during an interaction with the web site, is treated as a positive example of a string belonging to the unknown language.

The motivation in choosing a grammar to model the navigation behavior of the users of a web site lies in the sequential nature of browsing. The users usually begin from a page, spend some time reading it and then select the link which they think will satisfy their informational needs. The process is then repeated, producing a sequence of pages that each user has requested. Thus, the use of formal languages seemed natural since they are very well suited for modeling

sequences and also because of their strong theoretical foundation. For example, formal languages have already been used to model biological sequences with very good results [13]. To our knowledge, existing techniques towards our direction have not so far utilized grammatical inference methods to build the model of user navigation behavior. Popular approaches include Markov models, usually of first order [2–4, 8, 16, 17], or ad-hoc methods [18]. However, these approaches result either in difficult to interpret models or they do not have a sound theoretical background.

The rest of the paper is organized as follows. In the next section we summarize the major research directions in the field of user modeling and web usage mining. In section 3 we analyze the inference methods that were used and the necessary adaptations that were made for this specific problem. In section 4, we describe the framework in which the experiments were conducted and we interpret the results. Finally, in section 5 we draw our conclusions and suggest some directions for further work.

2 Related Work

Web Usage Mining is a relatively new research field inspired by the recent growth in the WWW and on-line information sources. Web usage mining, aims at discovering interesting usage patterns, by analyzing Web usage data such as those kept in the log file of a web server. The research work on Web Usage Mining is very extensive and covers aspects both related to traditional data mining issues as well as difficulties inherent in this specific area, such as the reliability of the usage data. Two thorough surveys on the field of Web Usage Mining can be found in [15] and [19].

An approach that has been employed for extracting patterns from web usage data is sequential pattern discovery. Markov chains have been extensively used for this purpose due to their ability to model sequential processes, such as browsing a web site. One of the earliest approaches was presented in [4]. In this work, a first order Markov model was employed in order to predict the subsequent link that a user might follow. A more elaborate approach is proposed in [2] in which the previous model is enhanced with the use of a hybrid prediction model. This model consults four simple Markov chains to select which document the user might request. In [17] first order Markov chains are used to accomplish four tasks: prediction of HTTP requests, adaptive navigation, automatic tour generation and locating personalized "hubs" and "authorities". In addition, the method presented in [8] exploits a mixture of first order Markov models, each one corresponding to the navigational behavior of users with similar behavior. This work is extended in [3] where an algorithm that recommends shortcut links to the user is presented. This can help users find the information they are looking for more quickly, which is especially interesting in the case of users with low bandwidth devices. Finally, in [16], the authors try to reduce the complexity of the previously used Markov model while maintaining a comparable predictive accuracy. This is achieved by calculating the most important navigation patterns and

removing everything else which is considered as noise. Generally, Markov chains produce simplistic models for the users navigation behavior, simply aggregating the information in the data, without involving any inference procedure.

Except from Markov models, a few other techniques have been used to discover sequential patterns in web usage data. In [18], a system called Web Utilization Miner is presented. This system provides an SQL-like language that can be used to set the constraints that the mined patterns should satisfy. These constraints can be highly elaborate since the language is very expressive. Unfortunately, setting the right constraints requires a human expert which makes the mining procedure semi-automatic. In [14] a method to discover interesting sequences using clustering is discussed. According to this method, the transitions between pages, which were performed during a user's single interaction with the site are clustered to produce models, which correspond to the navigational behavior of users. Finally, the Clementine tool of SPSS also employs sequential pattern discovery algorithm, known as CAPRI. CAPRI (Clementine A-Priori Intervals) is an association rule discovery algorithm that apart from discovering relationships between items, also finds the order in which these items have been traversed. However, all these methods are generally ad-hoc and they usually do not have any theoretical foundations.

3 Grammatical inference methods for sequence mining

In our work, we model user navigation on a web site as being governed by an unknown stochastic regular language L. Each interaction of a user with the web site can equivalently be expressed as a sequence of web pages the user requests from the web site. These sequences are represented as strings belonging to L. Thus, each web page that appears in the web server logs is represented by a unique terminal symbol. A web site visitor is then implicitly providing a string belonging to L by following the links on the web pages until the desired information is found or the user leaves the web site.

In order to infer the language L, from the usage data, we applied the widely used ALERGIA algorithm using the Hoeffding statistical test [9], and a slight modification of it, using the proportions test [11]. Our contribution lies primarily in the initial hypothesis with which these algorithms work. We tried starting from the standard probabilistic prefix tree automaton (PPTA) as well as the so called hypertext probabilistic automaton (HPA) [6]. The latter type of automaton has been previously used in web usage mining to locate the most frequently accessed paths in a web site [5] but not within the paradigm of grammatical inference.

The hypertext probabilistic automaton (HPA) was proposed as a compact way to model the users' interaction with a web site. This interaction is initially recorded in the web server's log file. Then, the sequence of web pages that each user requests within a certain period of time, which is called a user session, is extracted from it (see table 1) . Given a set of user sessions, a HPA is constructed in the following way: For each page A_i that appears in the sessions a unique state is created in the HPA. If a user requests the page A_j right after the page A_i,

Table 1. Sample user session set.

Session	Page Sequence
1	$A_1 \rightarrow A_2 \rightarrow A_3 \rightarrow A_4$
2	$A_1 \rightarrow A_5 \rightarrow A_3 \rightarrow A_4 \rightarrow A_1$
3	$A_5 \rightarrow A_2 \rightarrow A_4 \rightarrow A_6$
4	$A_5 \rightarrow A_2 \rightarrow A_3$
5	$A_5 \rightarrow A_2 \rightarrow A_3 \rightarrow A_6$
6	$A_4 \rightarrow A_1 \rightarrow A_5 \rightarrow A_3$

a transition is created in the HPA from A_i to A_j, labeled with the symbol A_j. Furthermore there is a starting state S and transitions from S to the states that correspond to pages which appear first in at least one user session. Each transition's probability, as well as each state's probability to be an accepting one, is estimated straightforwardly from frequency data of the given set of user sessions. For example, the HPA which models the sessions of table 1, is shown in figure 1. The numbers inside each state denote the number of requests of each page and the number of times this page was last in the session set. The number on each transition $A_i \rightarrow A_j$ denotes the number of times the users followed a link from A_i towards A_j. From this figure and the above description, it becomes apparent that the HPA is equivalent to first order Markov chains, which have been previously used for modeling the user behavior on the web. In the rest of this work we will refer to HPAs and Markov chains interchangeably.

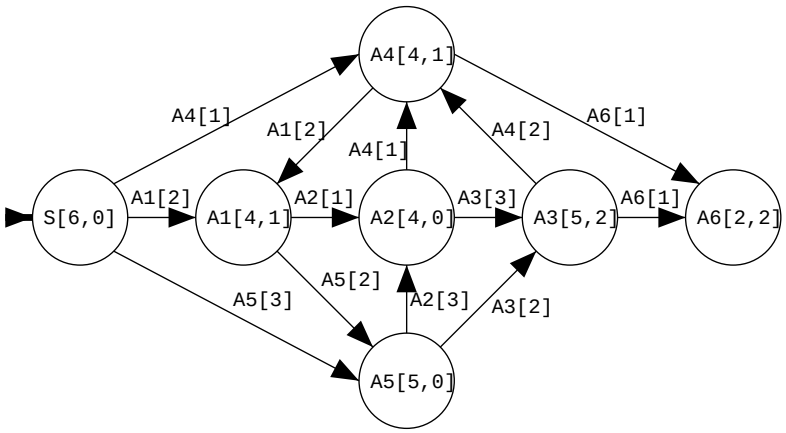

Fig. 1. The HPA that models the sessions in table 1.

The HPA approach reduces the space in which the search for the target automaton is conducted. Furthermore, the HPA can be derived from a PPTA, by merging together the states that correspond to the same web pages. As a result, the HPA is a generalization of the corresponding PPTA. This means that

a large portion of the original search space, containing more general automata than the PPTA but less general than the HPA, is excluded from the search when using the HPA. Additionally, some automata that are more general than the HPA may also be excluded from the new search space. In the next section the overall quality of the hypotheses in this smaller search space will be illustrated.

Therefore, it is interesting to compare the effect of grammatical inference starting from a HPA, rather than a PPTA. However, applying the existing stochastic regular grammar inference algorithms, which were designed for the PPTA, on an arbitrary automaton is not always feasible. This is due to assumptions about the structure of the automaton in each step of the inference procedure. For example, the ALERGIA algorithm, shown in table 2, avoids the possibility of infinite recursion by specifying a sequence for the merging operations which ensures that one of the states is always the root of a subtree of the PPTA. As a result, ALERGIA cannot be applied as is to an arbitrary automaton. However, the HPA has an interesting property that allows the algorithm to be applied despite the automaton's structure. As can be seen in figure 1, each symbol A_i labeling a transition fully defines the destination state, independently of the current state. The first consequence of this property is that state merging cannot result in a non-deterministic automaton. Non-determinism can in principal occur when merging two states which have transitions with the same symbol s towards different states. The merged state will then be non-deterministic because there will be two states to which s can lead. In order to remove non-determinism, the states causing it are repeatedly merged together. In the case of the HPA, transitions with the same symbol always lead to the same state, so non-determinism cannot occur. The other effect of the above property is that the compatibility test is greatly simplified. To test for the compatibility of two states it is no longer necessary to test the compatibility of all subsequent states, since these states are now identical and thus guaranteed to be compatible.

4 Experiments

Two data sets were used for the evaluation of our method. The first set was the MSWeb[3] data and the second was usage data from the web site "Information Retrieval in Chemistry"[4], hosted at NCSR "Demokritos". The data used in the experimental evaluation of our work were already split into sessions and some amount of "denoising", for example from bot sessions, had already taken place [14]. This prevented any experimentation with different session timeouts. Recent work that addresses the issue of session timeout can be found in [12]. In order to distinguish between different sessions of the same user, the sessions of the second data set were produced by constraining the time between two consecutive requests from the same IP to sixty minutes. If the time between two consecutive requests exceeded sixty minutes, the second request was considered as the beginning of a new session. The MSWeb data were grouped according to the identity

[3] Publically available from http://www.ics.uci.edu/∼mlearn/MLRepository.html
[4] http://macedonia.chem.demokritos.gr

Table 2. The original ALERGIA algorithm. $C(k)$ denotes the number of times state k was visited while parsing the user sessions. $C(k, \#)$ denotes the number of times state k was last in the user sessions. $C(k, x)$ is the number of times that the terminal symbol x was encountered in the sample when the automaton was in state k and $\delta(k, x)$ denotes the state in which the automaton goes when it is in state k and x in encountered.

ALERGIA(S, α)
  $A = \text{PPTA}(S)$
  for $j = 2$ to $|A|$ do
    for $i = 1$ to $j - 1$ do
      if COMPATIBLE(i, j, α)
        MERGE(i, j)
        DETERMINIZE(A)

COMPATIBLE(i, j, α)
  if DIFFERENT$(C(i), C(j), C(i, \#), C(j, \#), \alpha)$
    return false
  for each terminal t do
    if DIFFERENT$(C(i), C(j), C(i, t), C(j, t), \alpha)$
      return false
    if not COMPATIBLE$(\delta(i, t), \delta(j, t), \alpha)$
      return false
  return true

DIFFERENT$(n_i, n_j, f_i, f_j, \alpha)$
  return $\left|\frac{f_i}{n_i} - \frac{f_j}{n_j}\right| > \sqrt{\frac{1}{2}\ln\frac{2}{\alpha}}\left(\frac{1}{\sqrt{n_i}} + \frac{1}{\sqrt{n_j}}\right)$

of the user, rather than simply the IP address. As a result the constraint on the time between consecutive requests was not required. The MSWeb data set contained around 38000 sessions while the chemistry data set contained 4500 sessions. For the experiments, two thirds of the sessions were used to infer the automaton and the rest were used for its evaluation.

The tasks in which the inferred automata were assessed, were designed to capture their ability as predictors of the user's behavior and their ability as good recommenders for the next page a user would request. After the training phase, we hid the last page of each session in the test set and gave the automaton the rest of the pages in each session. That would lead the automaton in a specific state from which we could then construct a list of recommendations based on the most probable transitions from that state. A successful prediction occurs when the hidden page is among the most probable pages to request from the current state.

For the assessment of a list of recommendations we used the metric of expected utility [7]. The expected utility of a recommended page is simply the probability of requesting the page times the page's utility. As in [7], the utility v_{aj} of a page j for a user a is one if j belongs in the pages of the session and zero otherwise. The probability of requesting the ith page in a list was set to decay exponentially with i. Thus the expected utility of a list of pages (sorted

in decreasing order of transition probability) is:

$$R_a = \sum_{j=0}^{l} \frac{v_{aj}}{2^{j/h}}$$

where l is the ordinal number of the last page in the list and h is the viewing halflife. Similar to other exponentially decaying phenomena, the viewing halflife is the ordinal number of the page which has a 50-50 chance to be requested. In our experiments, following [7], we set this chance to occur in the fifth page. Finally, for a set of user sessions with the corresponding recommendation lists, the expected utility is:

$$R = 100 \frac{\sum_a R_a}{\sum_a R_a^{max}}$$

where R_a^{max} is the maximum achievable utility if the top of the list contained only pages with v_{aj} equal to one. For example, if a recommended list of items for a user a is the one shown in the second row of table 3 then the expected utility is:

$$R_a = \frac{1}{2^{0/4}} + \frac{1}{2^{2/4}} + \frac{1}{2^{5/4}} + \frac{1}{2^{6/4}} = 2.48$$

where only items A, C, E and F belong in the user session (item utility one). Furthermore, if we assume that there is one more item H whose utility equals one then the maximum achievable utility is:

$$R_a^{max} = \frac{1}{2^{0/4}} + \frac{1}{2^{1/4}} + \frac{1}{2^{2/4}} + \frac{1}{2^{3/4}} + \frac{1}{2^{4/4}} = 3.64$$

since H also contributes to the maximum achievable utility.

ordinal	0	1	2	3	4	5	6
item	*A*	*B*	*C*	*D*	*E*	*F*	*G*
utility	1	0	1	0	0	1	1

Table 3. A sample recommendation list.

The predictive accuracy of an inferred automaton is measured using the recall of the list it recommends. The recall of an information retrieval system is defined as the proportion of the totally available relevant documents retrieved by it. In this case, the hidden page is considered to be a relevant one. To calculate the recall, we compute the percentage of the user sessions in which the hidden page belongs in the recommended list as a function of the length of the list. The graph of this function gives a hint for the appropriate size of the list of pages that should be recommended to the user.

In figures 2 and 3 the recall of the best automata is drawn. The legend shows for each curve the initial hypothesis that was used, the algorithm that

was applied and the chosen value of α. With the term "prop test" we mean the ALERGIA algorithm with the statistical test proposed in [11]. On the other hand the term "alergia" refers to the original algorithm using the Hoeffding test.

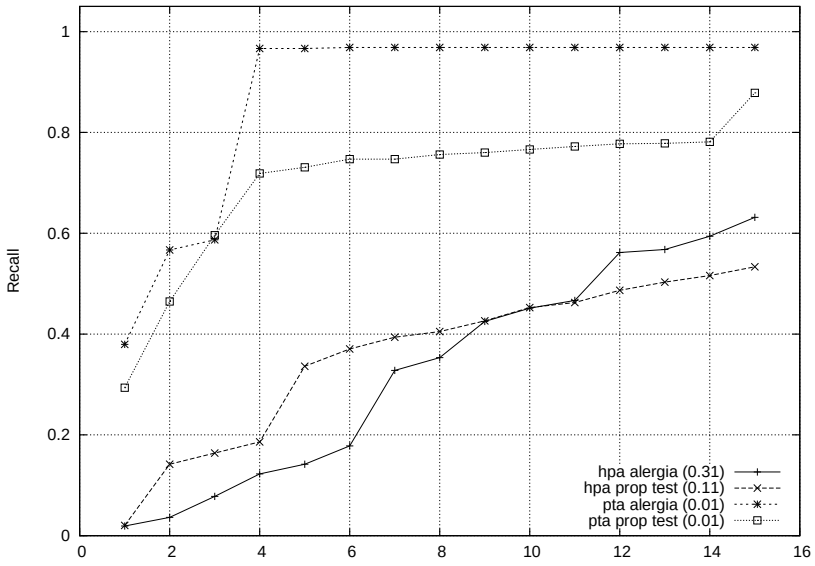

Fig. 2. Recall of some automata induced from the chemistry data set. The x axis represents the length of the recommendation list.

From these graphs, the models that were induced with the PPTA as initial hypothesis are clearly superior over the models that were induced from a Markov chain. The graph shows that traditional grammatical inference methods can produce models with more than 90% recall in the top 4 pages of the recommendation list. On the other hand, the automata induced from Markov chains have poor recall which increases steadily with the length of the list. This implies that their recommendations are actually guesses of the user behavior.

The expected utility of automata induced with different values of the α parameter is shown in figures 4 and 5. By inspecting the recall graphs, it was decided to set the maximum length of the produced recommendation list to ten items. As α increases, fewer states are merged and the automaton is less general. This means that the expected utility should decrease when α increases. Again the automata induced from Markov chains yield inferior models compared to the automata induced from the conventional algorithms. Regarding the automata inferred from the Markov chains, their expected utility seems to be greater from what the initial Markov chain would produce. This conclusion is drawn because for α equal to one we get the initial hypothesis, since no merging operations will be done, and also because the expected utility is decreasing as α increases. How-

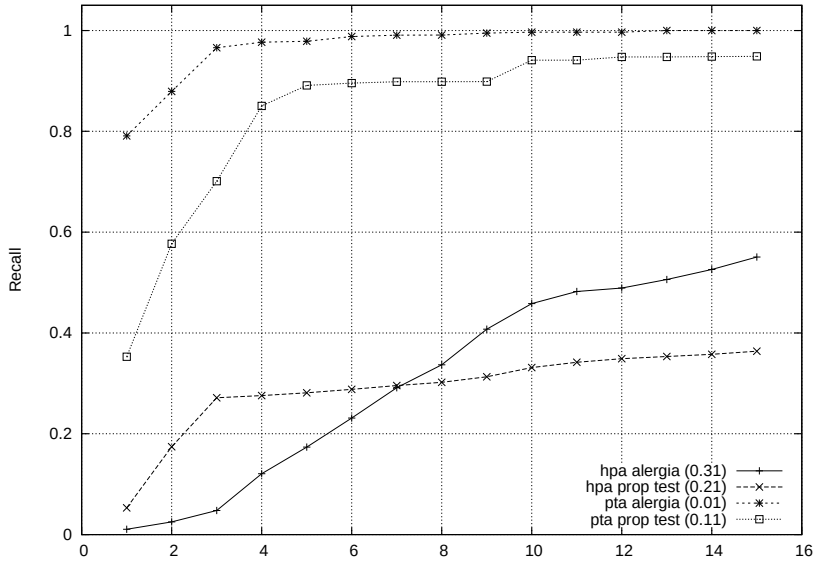

Fig. 3. Recall of some automata induced from the MSWeb data set. The x axis represents the length of the recommendation list.

ever, the expected utility in the case of these automata is much lower than that of the automata inferred from PPTAs and it also very sensitive to the changes of α. The latter observation is evidence that good models are excluded from the search space.

5 Conclusions

We evaluated two models for the navigation behavior of users in a web site. Both can be seen as stochastic regular grammars, but one is inferred from the traditional prefix tree automaton while the other from the first order Markov chain that has been previously used to model this navigation behavior. In this work we present the results produced by the well known ALERGIA algorithm suitably adapted to work with Markov chains as initial hypotheses. Two statistical tests were used to decide if two states of the automaton should be merged, the Hoeffding test and the proportions test. The difference between the two tests is that the latter is stricter than the former, leading to fewer state merging operations and larger automata. This has two effects. On one side the Hoeffding test leads to automata with better generalization ability which is reflected in the higher recall and expected utility that it achieves. On the other side, the proportions test is less likely to perform a bad merging, due to the inevitable presence of noise in the data. When two incompatible states are merged, the resulting automaton can be very different from the target. This can lead to results that are

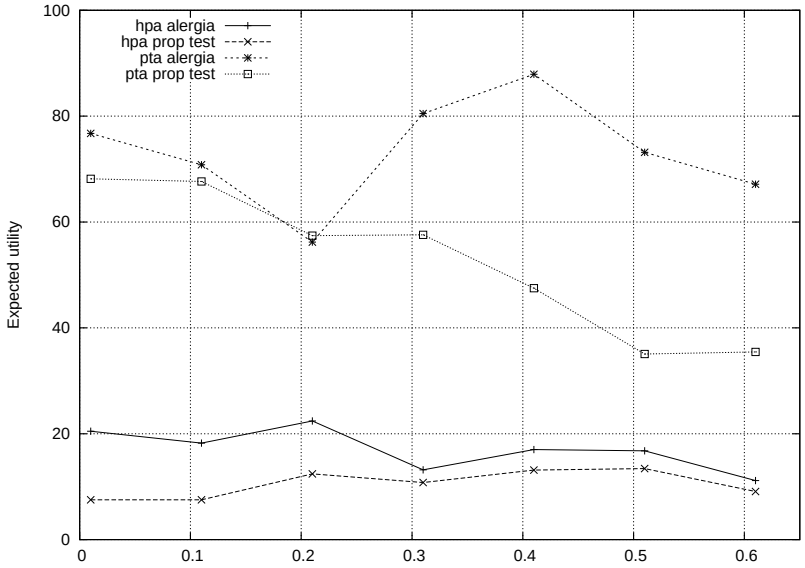

Fig. 4. The expected utility as a function of α for the chemistry data set. The maximum length of the recommendation list was set to 10.

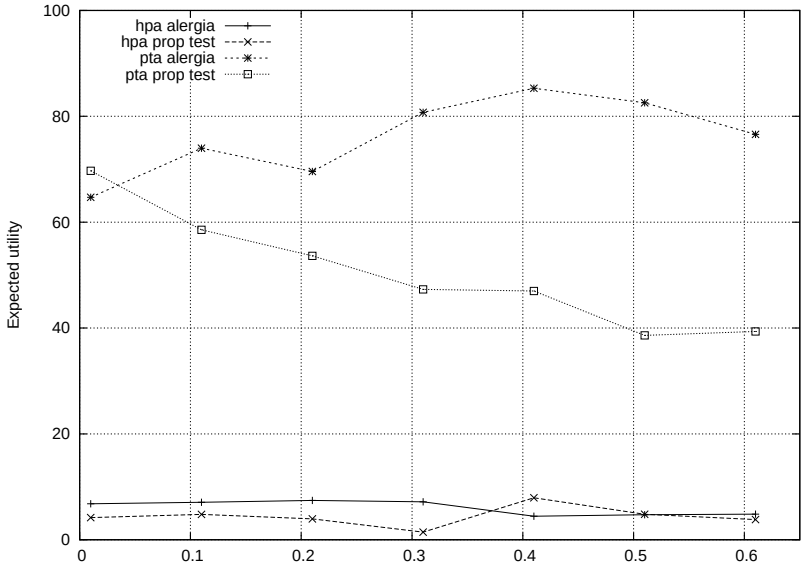

Fig. 5. The expected utility as a function of α for the MSWeb data set. The maximum length of the recommendation list was set to 10.

very sensitive to the value of α. This is the reason why the expected utility in figures 4 and 5 is predictably decreasing in the case of the proportions test. In the case of the Hoeffding test however, the curve is very sensitive to the value of α.

We have used data from two very different web sites. In both data sets the automata that were induced from the prefix tree automaton achieved better performance with respect to the metrics of recall and expected utility. This suggests that, even though stochastic regular languages have not been previously used in modeling the user navigation on the web, they are better suited for this task than the previously used Markov chains. This is attributed to the expressiveness of stochastic regular languages.

In addition to further empirical evaluation, we are interested in utilizing the inferred automata in different ways. The merged states are virtually sets of web pages that could be viewed as a clustering of the pages of a web site. Preliminary experiments with the Msweb data set are promising since they have shown the emergence of meaningful clusters. We are also looking into different criteria with which the states could be merged. Since merging the web pages according to their transition probabilities seems at first awkward, we are trying to adopt criteria that would provide a more intuitive meaning to a merged state. Finally, we plan to compare our results with those of well known sequential pattern discovery algorithms [1, 10].

Acknowledgments

We would like to thank the members of the team "Information Retrieval in Chemistry" of NCSR "Demokritos" (E. Varveri, A. Varveris and P. Telonis) for providing the chemistry data.

References

1. R. Agrawal and R. Srikant. *Mining sequential patterns*. In Proceedings of the Eleventh International Conference on Data Engineering, 1995.
2. D. Albrecht, I. Zukerman and A. Nicholson. *Pre-sending documents on the WWW: A comparative study*. Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, pp. 1274-1279, 1999.
3. C. Anderson, P. Domingos, and D. Weld. *Adaptive Web Navigation for Wireless Devices*. Proceedings of the 17th International Joint Conference on Artificial Intelligence, 2001.
4. A. Bestavros. *Using speculation to reduce server load and service time on the www*. Proceedings of the fourth ACM International Conference on Information and Knowledge Management, 403-410, 1995.
5. J. Borges and M. Levene. *Data Mining of User Navigation Patterns*. Lecture Notes in Computer Science, Vol. 1836, pp. 92-111, 1999.
6. J. Borges. *A Data Mining Model to Capture User Web Navigation Patterns*. PhD dissertation, 2000.

7. J. S. Breese, D. Heckerman and C. Kadie. *Empirical Analysis of Predictive Algorithms for Collaborative Filtering*. Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, 1998.
8. I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. *Visualization of Navigation Patterns on a Web Site Using Model Based Clustering*. Technical Report MSR-TR-00-18, 2000.
9. R. Carrasco and J. Oncina. *Learning Regular Grammars by Means of a State Merging Method*. Proceedings of the ICGI, 1994.
10. R. Cooley, B. Mobasher, and J. Srivastava. *Web Mining: Information and Pattern Discovery on the World Wide Web*. In Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence, 1997.
11. A. Habrard, M. Bernard and M. Sebban. *Improvement of the State Merging Rule on Noisy Data in Probabilistic Grammatical Inference*. Proceedings of the Fourteenth European Conference on Machine Learning (ECML 2003), 2003.
12. R. Kohavi and R. Parekh. *Ten supplementary analyses to Improve E-commerce Web Sites*. In Proceedings of the Fifth WEBKDD workshop, 2003.
13. S.H. Muggleton, C. H. Bryant and A. Srinivasan. *Learning Chomsky-like grammars for biological sequence families*. In Proceedings of the Seventeenth International Conference on Machine Learning, 2000.
14. G. Paliouras, C. Papatheodorou, V. Karkaletsis and C. D. Spyropoulos. *Clustering the Users of Large Web Sites into Communities*. Proceedings of International Conference on Machine Learning (ICML), 2000.
15. D. Pierrakos, G. Paliouras, C. Papatheodorou and C. D. Spyropoulos. *Web Usage Mining as a Tool for Personalization: a survey*. User Modeling and User-Adapted Interaction Journal, vol. 13, issue 4, pp. 311-372, 2003.
16. J. Pitkow and P. Pirolli. *Mining longest repeating subsequences to predict WWW surfing*. Proceedings of the 1999 USENIX Annual Technical Conference, 1999.
17. R. Sarukkai. *Link Prediction and Path Analysis Using Markov Chains*. Proceedings of the 9th World Wide Web Conference, 2000.
18. M. Spiliopoulou, L. C. Faulstich, and K. Wilkler. *A data miner analyzing the navigational behavior of Web users*. In Proceedings of the Workshop on Machine Learning in User Modeling of the ACAI99, 1999.
19. J. Srivastava, R. Cooley, M. Deshpande, and P. T. Tan. *Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data*. SIGKDD Explorations, 1(2), pp. 12-23, 2000.