Web Access Log Mining with Soft Sequential Patterns

Céline Fiot, Anne Laurent, Maguelonne Teisseire

▶ **To cite this version:**

Céline Fiot, Anne Laurent, Maguelonne Teisseire. Web Access Log Mining with Soft Sequential Patterns. Applied Artificial Intelligence, Aug 2006, Genova, Italy. pp.519-524, 10.1142/9789812774118_0074 . lirmm-00095914

HAL Id: lirmm-00095914
https://hal-lirmm.ccsd.cnrs.fr/lirmm-00095914v1

Submitted on 7 Oct 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

WEB ACCESS LOG MINING WITH SOFT SEQUENTIAL PATTERNS

C. FIOT, A. LAURENT AND M. TEISSEIRE
LIRMM - CNRS - UMII,
161 rue Ada, 34392 Montpellier, France
E-mail: {fiot,laurent,teisseire}@lirmm.fr

Web access logs contain time-stamped numerical data interesting to mine for numerous applications such as customer targeting, automatic updating of commercial websites or web server dimensioning. In this context, the algorithms for sequential patterns mining are well adapted to the temporal aspect of the data. However, they do not allow processing numerical information frequently found in web data. In previous works we defined fuzzy sequential pattern mining framework to cope with the numerical representation problem. Thus, in this paper, we apply these algorithms to web mining and assess them through different experiments showing the relevancy of this work in the context of web access log mining.

1. Introduction

The quantity of data from the World Wide Web is growing dramatically: requested URLs, number of requests or session duration, etc. are gathered automatically by web servers and stored in access log files. Analysing these data can provide useful information for performance enhancement or customer targeting. In this context, many works have been proposed to mine usage patterns and user profiles [1, 2, 3, 4]. Particularly, [5] provides knowledge from database of visited page sequences. However this method, based on sequential pattern mining, cannot be used to mine numerical data contained in these log files, such as number of requests for the same page, transfer rates, number of downloaded kilobytes or duration of sessions. Few works have been carried out to process such numerical data and most of them are restricted to association rules [6, 7, 8]. However, association rule based approaches do not take the sequencing of data into account. Sequential patterns are thus more adapted to time-stamped data. Nevertheless numerical data must be pre-processed into a binary representation to be mined by existing approaches, which necessarily leads to a loss of information. In order to cope with this problem, we propose here

to apply an efficient fuzzy approach for sequential pattern mining to mine time-stamped numerical data from web access logs. This approach, defined in our previous works [9], is based on the definition of *fuzzy* intervals. Obtained patterns are of the type "60% of users visiting *a lot* the Disneyland website and *a few* Eiffel Tower pages visit later *a lot of* traveling websites". These patterns are characterized by their support, which is the percentage of users who have followed this rule. Three approaches are proposed to mine such rules: SpeedyFuzzy, MiniFuzzy and TotallyFuzzy, differing by the support computation. The end-user is allowed to choose between the speed of result extraction and the accuracy of the obtained frequent patterns. Implementation of these solutions is based on an efficient and original data scanning method, which extends the PSP algorithm proposed in [5]. Experiments were carried out on synthetic datasets and on real-world data. They highlight the feasibility and robustness of a fuzzy approach. We present here the experiments carried out on web access logs.
This paper gives an overview of our algorithms and focuses on the processing of web log data. Section 2 presents an introduction to sequential patterns and fuzzy sequential patterns. Section 3 presents the experiments to analyse web access logs. Section 4 concludes on the different perspectives associated to this work.

2. From Crisp to Fuzzy Sequential Patterns

In this section, we briefly describe the basic concepts of sequential patterns and fuzzy sequential patterns.

Let $\mathcal{T}$ be a set of object records where each record consists of three information elements: an object-id, a time-stamp and a set of items. An **itemset**, $(i_1 i_2 \dots i_k)$, is a non-empty, unordered set of items taken from $I = \{i_1, i_2, ..., i_m\}$. A **sequence** s is a non-empty ordered list of itemsets, denoted by $< s_1 s_2 ... s_p >$. For instance, let us consider the purchase sequence $s =<$ (1) (2 3) (4) (5) $>$. Then all items of the sequence were bought separately, except products 2 and 3 which were purchased together. Roughly speaking, the **support** of a sequence is the percentage of objects who have s in their records. In order to decide whether a sequence is frequent or not, a minimum support value ($minSupp$) is specified by the user. A sequence s is said to be frequent if the condition $support(s) \geq minSupp$ holds. Given a database of object records the problem of sequential pattern mining is to find all maximal frequent sequences [10].

In this context, fuzzy sequential patterns have been defined to handle quantitative attributes. T.-P. Hong et al. [11] and Y.-C. Hu et al. [12] have proposed approaches to mine fuzzy sequential patterns without providing experimental evaluations. We thus here consider the complete approach from [9]. This work proposes three algorithms, SpeedyFuzzy, MiniFuzzy and TotallyFuzzy, defining three levels of fuzzification depending on the adopted counting method for fuzzy set cardinality as detailed later.

In this framework, we consider the fuzzy extension of item, itemset and sequence notions. The quantity universe of each attribute is discretized into fuzzy subsets, as explained in next paragraph, taking the dataset Table 1 as an example. A **fuzzy item** is the association of one item and one corresponding fuzzy set. For instance, $[/php/tutor.htm, lot]$ is a fuzzy item where *lot* is a fuzzy set defined by a membership function on the access quantity universe of the item */php/tutor.htm*. A **fuzzy itemset** is a set of fuzzy items, e.g. $([/php/tutor.htm, lot][/php/functions.php, little])$. One fuzzy itemset only contains one fuzzy set related to one single item. For example, the fuzzy itemset $([/php/tutor.htm, lot][/php/tutor.htm, little])$ is not a valid fuzzy itemset. A **fuzzy sequence** $S =< s_1 \cdots s_g >$ is a sequence consisting of fuzzy itemsets, e.g. $< ([/php/tutor.htm, lot][/php/faq.phps, little])([/php/functions.php, lot]) >$.

Table 1. Access grouped by IPs and ordered by their timestamp (empty cells indicate that the page has not been accessed)

Cust./IP	Date	/php/tutor.htm	/php/eg.htm	/php/functions.php	/php/faq.php
C1	d1	2			
82.228.151.52	d2	1	3	1	
	d3	4		1	
	d4			1	
	d5			2	2
	d1	2			
86.197.153.12	d2			2	
	d3		4	1	
	d4	3			
	d1				3
82.226.199.47	d2	3	1		
	d3				5
	d4			2	
82.226.199.48	d1		2		

First, the quantitative database is converted into a membership degree database, as shown on Figure 1. These partitions (e.g. *"little", "medium", "lot"*) are automatically built by dividing the universe of quantities into intervals. Each interval groups the same proportion of users. It is then fuzzified in order to enhance generalization.

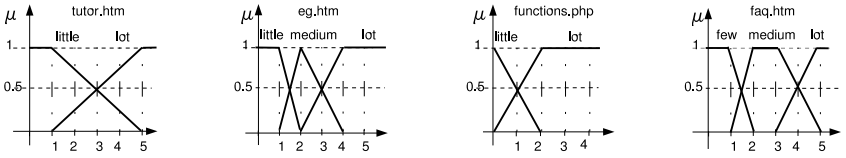


Figure 1. Membership functions for the fuzzy sets of the database attributes

From these membership functions we get the membership degrees for each record and for each fuzzy set. For instance, at d2, C1 has browsed */php/tutor.htm* once, 3 times */php/eg.htm* and once */php/functions.php*. Figure 1 shows that 1 access to */php/tutor.htm* is converted into a degree equals to 1 for the fuzzy set *little*, 3 for */php/eg.htm* belongs to both sets *little* and *lot* with a degree of 0.5. Table 2 describes these values for customer 1.

Table 2. Membership degrees for customer 1

	Items							
	/php/tutor.htm		/php/eg.htm		/php/functions.php		/php/faq.php	
D.	li.	L.	m.	L.	li.	L.	m.	L.
d1	0.75	0.25						
d2	1		0.5	0.5	0.5	0.5		
d3	0.25	**0.75**			0.5	0.5		
d4					0.5	**0.5**		
d5						1	1	

The support of a fuzzy itemset is computed as the proportion of objects supporting it. However the cardinality of a fuzzy set depends on the counting method. We transpose here three of these techniques in the framework of fuzzy sequential patterns and we propose three definitions for the fuzzy support of an itemset (X, A):

• SpeedyFuzzy counts objects recording, for each item of the itemset, a membership degree not null at least once ($S_{SF}(c, (X, A))$, in Table 3).

• MiniFuzzy is based on a thresholded count, incrementing the number of objects supporting the fuzzy itemset when each of its item has a membership degree greater than a specified threshold in the data sequence of the object ($S_{MF}(c, (X, A))$, in Table 3).

• TotallyFuzzy carries out a thresholded Σ-count. The support ($S_{TF}(c, (X, A))$, in Table 3) is computed as a weighted sum of the membership degrees greater than a given threshold. In this approach the importance of each fuzzy itemset in the data sequence is taken into account in

the support computation, using generalized t-norm and t-conorm operators $\overline{\top}$ and $\underline{\perp}$. As for MiniFuzzy, a minimum relevancy is required to consider the itemset as being supported. It is specified by the minimum relevancy threshold ω.

Support	Value	Condition to hold $\exists t \in \mathcal{T}_c$|
$S_{SF}(o,(X,A))$	1	$\forall [x,a] \in (X,A),\ \mu_a\big(t[x]\big) > 0$
$S_{MF}(o,(X,A))$	1	$\forall [x,a] \in (X,A),\ \mu_a\big(t[x]\big) > \omega$
$S_{TF}(o,(X,A))$	$\underline{\perp}_{j=1}^{\lvert\mathcal{R}_o\rvert} \overline{\top}_{[x,a]\in(X,A)} \big[\mu_a\big(t_j[x]\big)\big]$	$\forall [x,a] \in (X,A),\ \mu_a\big(t[x]\big) > \omega$

The **support of a fuzzy sequence** is computed as the ratio of the number of objects supporting this fuzzy sequence compared to the total number of objects in database ($|\mathcal{O}|$):

$$FSupp_{(X,A)} = \frac{\sum_{o\in\mathcal{O}} \Big[S(o,gS)\Big]\Big]}{|\mathcal{O}|} \tag{1}$$

where the support degree $S(o,gS)$ indicates wether the object o supports the fuzzy sequence gS, i.e. each itemset of the sequence in the same order. This support degree is computed algorithms detailed in [9].

Example 2.1. In this example, we consider the membership database for customer 1, given in Table 2 and assess the support of the sequence $< ([/php/tutor.htm, lot])([/php/functions.php, lot]) >$. With SpeedyFuzzy, we consider the items underlined into account. With MiniFuzzy (ω=0.49), customer 1 supports the sequence, we take the items boldfaced into account. With TotallyFuzzy (ω=0.49), customer 1 supports the sequence, the best occurrence of the sequence is kept, twice underlined.

Table 4. Extracted sequential patterns with *minsupp*=55%

SpeedyFuzzy	<([/php/tutor.htm, little])([/php/functions.php, lot])>	75%
	<([/php/tutor.htm, lot])([/php/functions.php, lot])>	75%
MiniFuzzy	<([/php/tutor.htm, little])([/php/functions.php, lot])>	75%
	<([/php/functions.php, lot])>	75%
TotallyFuzzy	<([/php/tutor.htm, little])([/php/functions.php, lot])>	69%
	<([/php/functions.php, lot])>	56%

Table 4 shows the sequential patterns respectively extracted by SpeedyFuzzy, MiniFuzzy and TotallyFuzzy. Note that the frequent items are the

same for all counting methods. The difference is in the number and length of the sequences. For a same *minSupp*, the number and length of the mined patterns are indeed greater with MiniFuzzy or SpeedyFuzzy than with TotallyFuzzy. This is due to the thresholded Σ-count. This reduction in the number of patterns can be used for a database containing very high amount of frequent patterns to find the most relevant ones. The advantage of this method is to be more selective and therefore to find the most relevant sequential patterns. The user will thus be provided with a selection of patterns, and not only have to assess a selection of patterns and not a really large quantity of them. So within the context of web mining, SpeedyFuzzy could be used to identify user profiles, whereas TotallyFuzzy could be used for mining detailed downloading rate.

3. Experiments

In this section, we present a comparison of performances between the three algorithms SpeedyFuzzy, MiniFuzzy and TotallyFuzzy and the algorithm PSP [5] and results of web access logs mining with TotallyFuzzy.

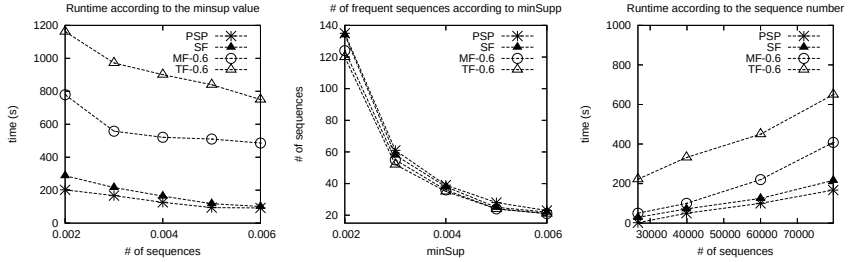
The aim of these experiments is to show that soft sequential pattern mining bring more relevant information compared to crisp sequential pattern mining. In our case, access logs from a laboratory website were prepared and mined to find frequently visited pages – such as in crisp sequential pattern mining – but also repeatedly visited pages. The access logs were pre-processed and the dataset recorded the number of accesses to one page, the same half-day by one user. For example, record *"1500 5067 10 6"* means that *"visitor 21500"* on half-day *5067* visited *6* times the URL coded by *10*. This dataset contained 27209 web pages visited by 79756 different IPs over 16 days (32 half-days). As explained previously, we fuzzify the number of access per visitors on each page 3 fuzzy sets using a tool based on the *DiscretizeFilter* module of Weka [13]. One should note that all data modeling choices have impact on what can be extracted from the data. Next experiments should thus be carried out using a fuzzification more adapted to the dataset, based on the approach of [14].

First, we compared performances of the fuzzy algorithms to those of PSP [5], a sequential pattern algorithm working on binary data with the same efficient structure as our fuzzy algorithms. Figure 2(a) shows the runtime according to *minSupp*. Note that the extraction time increases as *minSupp*

decreases. The number of frequent patterns and generated candidates is indeed higher for a low *minSupp*. So the algorithms scan the dataset more times. It can also be noted that SpeedyFuzzy is almost as fast as PSP despite the fact that it scans three times more items. TotallyFuzzy and MiniFuzzy run slightly slower.

If we compare the number of frequent sequences for a same minimum support, Figure 2(b) shows that MiniFuzzy and TotallyFuzzy extract less frequent sequences than SpeedyFuzzy and PSP, due to the support definition. In fact those two fuzzy algorithms, MiniFuzzy and TotallyFuzzy, only keep the items which have a degree greater than ω and so which are considered as relevant by the user. The number of frequent sequences is then necessarily reduced compared to SpeedyFuzzy or PSP.

Figure 2(c) shows the extraction time according to the number of data sequences in database for $minSupp = 0.2$. The fuzzy algorithms have the same linear behavior as PSP, even if they have not equal performances.



(a) (b) (c)

Figure 2. (a)Runtime according to *minSupp* for 79756 sequences; (b)Number of frequent sequences according to *minSupp* for 79756 sequences; (c)Runtime according to the number of sequences in the datasets, for *minSupp*=0.2%

These behaviors confirm the good results obtained in experiments on synthetic datasets described in [9].

Thes preliminary experiments showed that results on fuzzy logs are consistent with the one on crisp values. The same frequent URLs can indeed be found using crisp or soft sequential pattern mining. The advantages of our method concern the additional knowledge supplied by quantities. Indeed, while the crisp algorithm PSP found URL 139 was frequently accessed, TotallyFuzzy found that it was frequently accessed between 2 and 5 times during the same period. Our patterns also revealed that the root page was only a step in the browsing sequences and was often accessed only once in

the same session. A global trend in this browsing dataset is that people tend to access only few pages frequently in the same session.

4. Conclusion and perspectives

Historical analysis of web logs and more especially sequential pattern extraction from web databases is highly interesting for customer targeting, server dimensioning or transfer optimization. In this paper we propose to mine web logs using fuzzy sequential patterns handling three fuzzification levels thanks to three algorithms. This method is indeed well-adapted to process time-stamped numerical data contained in access logs. It also enable one to mine information with a minimum relevancy, for instance the time spent on a web page or the number of visits on the same page. This choice allows the extraction of frequent sequences by making a trade-off between relevancy and performance. The algorithms we use, SpeedyFuzzy, MiniFuzzy and TotallyFuzzy, have been implemented and tested, showing the interest of our novel approach and the feasibility for the fuzzy methods described. Thus the user can handle three fuzzification levels thanks to our three algorithms. Experiments on web access logs have highlighted the relevance of our proposal. This work builds many perspectives, more particularly we plan to extend our experiments to mine web-purchases for e-marketing.

References

1. M. Spiliopoulou and L. C. Faulstich. WUM: A tool for Web utilization analysis. *Lecture Notes in Computer Science*, 1590, 1999.
2. T.-W. Yan, M. Jacobsen, H. Garcia-Molina and U. Dayal. From user access patterns to dynamic hypertext linking. In *Proc. of the 5th Int. World Wide Web Conf. on Computer networks and ISDN systems*, pages 1007–1014, 1996.
3. O.-R. Zaiane, M. Xin and J. Han. Discovering web access patterns and trends by applying olap and data mining technology on web logs. In *Proc. of the Advances in Digital Libraries Conf.*, pages 19–30, 1998.
4. E. Damiani, B. Oliboni, E. Quintarelli and L. Tanca. Modeling users' navigation history. In *Workshop on Intelligent Techniques for Web Personalisation. (IJCAI01)*, 2001.
5. F. Masseglia, P. Poncelet and R. Cicchetti. An efficient algorithm for web usage mining. *Networking and Information Systems Journal*, 2(5-6), 1999.
6. A. Fu, M. Wong, S. Sze, W. Wong and W. Yu. Finding Fuzzy Sets for the Mining of Fuzzy Association Rules for Numerical Attributes. In *Proc. of the 1st Int. Symp. on Intelligent Data Engineering and Learning*, 1998.
7. C. M. Kuok, A. W.-C. Fu and M. H. Wong. Mining Fuzzy Association Rules in Databases. *SIGMOD Record*, 27(1), pages 41–46, 1998.

8. R. Srikant and R. Agrawal. Mining Quantitative Association Rules in Large Relational Tables. In *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, pages 1–12, 1996.
9. C. Fiot, A. Laurent and M. Teisseire. Speedy, Mini and Totally Fuzzy: Three Ways for Fuzzy Sequential Patterns Mining. Technical Report 5035, LIRMM: http://www.lirmm.fr/~fiot/Publi/rtech5035.pdf, 2005.
10. R. Agrawal and R. Srikant. Mining Sequential Patterns. In *Proc. of the 11th Int. Conf. on Data Engineering*, pages 3–14, 1995.
11. T.P. Hong, K.Y. Lin and S.L. Wang. Mining Fuzzy Sequential Patterns from Multiple-Items Transactions. In *Proc. of the Joint 9th IFSA World Congress and 20th NAFIPS Int. Conf.*, pages 1317–1321, 2001.
12. Y.-C. Hu, R.-S. Chen, G.-H. Tzeng and J.-H. Shieh. A Fuzzy Data Mining Algorithm for Finding Sequential Patterns. *Int. J. of Uncertainty Fuzziness Knowledge-Based Systems*, 11(2), pages 173–193, 2003.
13. I. H. Witten and E. Frank. Data Mining: Practical Machine Learning Tools with Java Implementations, 2000.
14. A. Gyenesei and J. Teuhola. Multidimensional fuzzy partitioning of attribute ranges for mining quantitative data: Research articles. *Int. J. Intell. Syst.*, 19(11), pages 1111–1126, 2004.