Mining web navigation path fragments

Wolfgang Gaul Lars Schmidt-Thieme

Institut für Entscheidungstheorie und Unternehmensforschung,
Universität Karlsruhe, D-76128 Karlsruhe, Germany

Abstract

For many web usage mining applications like, e.g., user segmentation, it is crucial to compare navigation paths of different users. We model user navigation path fragments by generalized subsequences that take into consideration local deviations but still sketch the global user navigational behaviour. This paper presents a new algorithm of apriori type for mining all generalized subsequences of user navigation paths with prescribed minimal occurence from a given database.

1 Introduction

E-commerce needs web usage mining that aims at considering different phases of consumer behaviour, extending the focus from classical buying behaviour analysis to data mining of different kinds of contacts with (potential) customers. User navigation paths in the web or even fragments of visits of websites establish an important source of information. For most higher level analytical tasks and applications like user segmentation, recommender systems etc., paths of different users have to be compared. Most path distances can be viewed as ordinary distance measures on a feature space of path fragments. As this space turns out to be high-dimensional and sparsely populated, dimension reduction schemes are needed. One such scheme consists in selecting the subspace spanned by frequent path fragments.

There are different kinds of fragments: The simplest kinds of fragments are occurences of single pages or set of pages in a user path. Frequent page sets can be mined by the standard apriori algorithm (see Agrawal and Srikant (1994)). As subsets neglect the sequential structure of user paths, better choices for path fragments are subsequences. Frequent *contiguous subsequences* can be mined by a well known variant of the apriori algorithm (see Agrawal and Srikant (1995) with modifications by Srikant and Agrawal (1996)). Borges and Levene (1998 and 1999) have developed algorithms for sequence mining on aggregated data. As a third kind of path fragments *generalized subsequences* containing wildcards have been proposed in the web mining literature (see Spiliopoulou (1999)). Generalized subsequences sketch the global navigational behaviour of users. Several algorithms exist to mine frequent generalized subsequences of a specified type (called templates, i.e., subsequences with prescribed positions of wildcards, see Spiliopoulou (1999) and Buechner et al. (1999)). Other authors following a broader approach have constructed algorithms to find frequent subsequences of pages with attached attributes and relations (called generalized episodes, see Mannila and Toivonen (1996)). While those algorithms are perfectly suited for use in interactive analysis, a general algorithm mining *all* frequent generalized subsequences (of a given minimal support) still is missing. In this paper we describe a new algorithm that fills this gap.

2 Formal Background

Let R be an arbitrary set of resources extracted from a webserver's logfile, where the navigational behaviour of anonymous visitors has been recorded, and $R^\star := \bigcup_{i\in\mathbb{N}} R^i \cup \{\emptyset\}$ the set of finite sequences of elements of R (with $\emptyset$ as the empty sequence), here used to model user paths. For a sequence $x \in R^\star$ the *length* $|x|$ is the number of symbols in the sequence ($|x| := n$ for $x \in R^n$, $|\emptyset| := 0$). Let $x, y \in R^\star$ be two such sequences. We say that *x is a subsequence of y* ($x \leq y$), if there is an index $i \in \{0, \ldots, |y| - |x|\}$ with $x_j = y_{i+j} \quad \forall j = 1, \ldots, |x|$. x is a *strict subsequence of y* ($x < y$), if it is a subsequence of y but not equal to y ($x \leq y \wedge x \neq y$).

A pair of sequences $x, y \in R^\star$ *is overlapping on $k \in \mathbb{N}_0$ elements*, if the last k elements of x are equal to the first k elements of y ($x_{last-k+i} = y_i \quad \forall i = 1, \ldots k$). (Note that if x and y are overlapping on k elements, they are also overlapping on $q \in \mathbb{N}_0$ elements for $q \leq k$.) For such a pair of sequences $x, y \in R^\star$ overlapping on k elements we define the *k-telescoped concatenation of x and y* to be

$$\begin{aligned} x +_k y := \quad & (x_1, \ldots, x_{\text{last}-k}, y_1, \ldots, y_{\text{last}}) \\ = & (x_1, \ldots, x_{\text{last}}, y_{k+1}, \ldots, y_{\text{last}}). \end{aligned}$$

Note that any two sequences are 0-overlapping and the 0-telescoped concatenation of two sequences is just their arrangement one behind the other. For a pair of sets of sequences $X, Y \subseteq R^\star$ we denominate the set of k-overlapping pairs $x \in X, y \in Y$ by $X \oplus_k Y$ and the set of k-telescoped sequences of all k-overlapping pairs shortly as the *set of k-telescoped sequences of X and Y*:

$$\begin{aligned} X +_k Y := \quad & +_k(X \oplus_k Y) \\ = & \{x +_k y \mid x \in X, y \in Y \text{ are overlapping on } k \text{ elements}\}. \end{aligned}$$

Now let S be a finite list of such sequences $x \in R^\star$ (allowing multiplicities if different users take the same path). For an arbitrary sequence $x \in R^\star$ we denominate the relative frequency of sequences of S containing x as subsequence as *support of x with respect to S*:

$$\sup_S(x) := \frac{|\{s \in S \mid x \leq s\}|}{|S|}$$

The task of *searching all frequent subsequences* in the given list of sequences S means to find all sequences $x \in R^\star$ with at least a given minimal support, i.e. with $\sup_S(x) \geq \text{minsup}$ and $\text{minsup} \in \mathbb{R}^+$ a given constant. As the support of subsequences of a sequence is greater than or equal to the support of the sequence itself, one can build frequent subsequences recursively starting from the sequences of length $n = 1$. With all sequences of length 1 as initial *set of candidates* the algorithm performs two steps: first, it computes the support values of all candidates and selects those candidates as frequent subsequences that satisfy the minimal support constraint; second, it builds a new set of candidates of length $n + 1$ for the next step by trying to join frequent subsequences of length n in the following manner: two sequences c and d of length n are joined to a sequence of length $n+1$ if they overlap on $n-1$ elements, i.e. $(c_2, \ldots, c_n) = (d_1, \ldots, d_{n-1})$; the joined sequence is $c +_{n-1} d$. Algorithm 1 gives the formal description of this procedure.

Algorithm 1 Apriori algorithm adapted for sequences

Require: set of items R (resources), list S of (finite) sequences (user paths) of elements of R, minimal support value minsup $\in \mathbb{R}^+$.
Ensure: set of frequent subsequences $F := \bigcup_{n \in \mathbb{N}} F_n$ of the sequences of S with support of at least minsup.
  $C := \{\{r\} \mid r \in R\}$ set of initial candidates,
  $n := 1$.
  while $C \neq \emptyset$ **do**
    compute $\sup_S(c) \quad \forall c \in C$ by counting the number of occurences of each c in S (one loop through S).
    $F_n := \{c \in C \mid \sup_S(c) \geq \text{minsup}\}$
    $C := F_n +_{n-1} F_n$ {compute new candidate sequences with length n+1}
    $n := n + 1$
  end while

Please note, that for the special case of sequences describing paths on a graph, in the first join step only 0-overlapping pairs of sequences of length 1, i.e., pairs of nodes of the graph, have to be considered that are linked by an edge. — This adaption of the classical apriori algorithm for sets (see Agrawal and Srikant (1994)) to sequences has first been published by Agrawal and Srikant (1995) (with modifications by Srikant and Agrawal (1996)). It has been used for finding subsequences in web mining paths by Chen et al. (1996) and other authors afterwards (Viveros et al. (1997), Chen et al. (1998), Cooley et al. (1999)).

3 Mining frequent generalized subsequences

By a *generalized sequence in R* we mean a (finite ordinary) sequence in the symbols $R \cup \{\star\}$ with an additional symbol $\star \notin R$ called *wildcard*, such that no two wildcards are adjacent:

$$R^{\text{gen}} := \{x \in (R \cup \{\star\})^\star \mid \nexists i \in \mathbb{N} : x_i = x_{i+1} = \star\}$$

The wildcard symbol $\star$ is used to model partially indeterminate sequences, matching arbitrary subsequences. This notion of generalized sequence first has been introduced in web mining literature by Spiliopoulou (1998). For a generalized sequence $x \in R^{\text{gen}}$ we define its *length* $|x|$ as the length of the sequence in the symbols $R \cup \{\star\}$, i.e. $|x| := n$, if $x \in (R \cup \{\star\})^n$. Now let $x, y \in R^{\text{gen}}$ be two generalized sequences. We say that x *matches* y or y *generalizes* x ($y \vdash x$), if there exists a mapping

$$m : \{1, \ldots, |x|\} \to \{1, \ldots, |y|\}$$

(called *matching*) with the following properties:

1. m maps indices of elements of x to indices of elements of y that coincide or to a wildcard ($y_{m(i)} = x_i$ or $y_{m(i)} = \star$).
2. m covers all indices of y of non-wildcard elements ($y_i \in R \Rightarrow m^{-1}(i) \neq \emptyset$).
3. m is weakly monotonic increasing.
4. m is even strictly monotonic at places where its image does not belong to a wildcard
($m(i) = m(i+1) \Rightarrow y_{m(i)} = \star$).

Please note that as the set of ordinary sequences $R^\star$ is a subset of the set of generalized sequences R^{gen}, this also defines the notion of an ordinary sequence matching a generalized sequence. Obviously matchings are not uniquely determined by two generalized sequences x and y. A trivial example is $\star A \star \vdash AA$ with the two matchings $m_1 = \{(1,1),(2,2)\}$ and $m_2 = \{(1,2),(2,3)\}$. Finally we carry over the notions of subsequence and of k-telescoped concatenation from ordinary sequences to generalized sequences without any change. Note the difference between $A \star C$ not being a subsequence of $ABCD$ but matching a subsequence of it (i.e. $A \star C \vdash ABC$ and $ABC \leq ABCD$).

Again, let S be a finite list of ordinary sequences (user paths) $x \in R^\star$. For an arbitrary generalized sequence $x \in R^{\text{gen}}$ we denominate the relative frequency of sequences containing a subsequence which matches x as *support of x with respect to S*:

$$\text{sup}_S(x) := \frac{|\{s \in S \mid \exists y \leq s : x \vdash y\}|}{|S|}$$

Now, *mining frequent generalized subsequences* is the label for the task to find all generalized sequences with at least a given minimal support. As we are looking for subsequences anyway, we can narrow our view to *closed generalized subsequences*, i.e. generalized subsequences without leading or trailing wildcard ($x \in R^{\text{gen}}$ with $x_1, x_{\text{last}} \in R$), that we call *path fragments*.

Up to now no general algorithm for finding all frequent generalized subsequences in a list of sequences is known. Spiliopoulou (1999) has invented an algorithm for finding frequent generalized subsequences in a limited subspace of the search space: her generalized sequence miner (GSM) looks for generalized sequences of a given length and wildcards at given positions (such subspaces are described by so called *templates*; see Buechner et al. (1999) for another approach using templates to limit the search space; templates are useful in the framework of interactive tools like WUM, see Spiliopoulou and Faulstich (1998) and Spiliopoulou et al. (1999)).

	sequence	length		sequence	length
	ab...cd	n+1		a⋆b...cd	n+1
=	ab...c	n	=	a⋆b...c	n
$+_{n-1}$	b...cd	n	$+_{n-2}$	b...cd	n-1
	ab...c⋆d	n+1		a⋆b...c⋆d	n+1
=	ab...c	n-1	=	a⋆b...c	n-1
$+_{n-2}$	b...c⋆d	n	$+_{n-3}$	b...c⋆d	n-1

Table 1: Construction of closed generalized subsequences of length ≥ 4.

We present a modification of the apriori algorithm for sequences to path fragments, resulting in a general algorithm for finding frequent generalized subsequences. The idea is pretty simple. As we are looking only at closed generalized sequences, the support of any subsequence of such a closed generalized sequence again is greater than or equal to the support of the sequence itself. Now, as adjacent wildcards are not allowed, we can get every path fragment of length $n+1$ (for $n \geq 3$) as junction of two overlapping path fragments of the kind described in table 1.

Thus, we only have to modify the join step of the apriori algorithm for building new candidates of length $n+1$ in such a way that we not only use the frequent (closed generalized) subsequences of length n but also those of length $n-1$ from the step before, and try all possible combinations. Closed generalized subsequences of length 3 containing a wildcard have the form $(x, \star, y)$ with $x, y \in R$, shorter closed generalized subsequences cannot contain wildcards.

Algorithm 2 gives the exact formulation of the necessary comparisions. Of course, the computation of the support values of the candidate generalized sequences also has to be modified. The performance characteristcs of the algorithm is the same as for the apriori algorithm for ordinary sequences: to find sequences of length n, n loops through the database have to be accomplished.

Algorithm 2 Apriori algorithm adapted for generalized sequences

Require: set of items R (resources), list S of (finite) sequences (user paths) of elements of R, minimal support value minsup $\in \mathbb{R}^+$.
Ensure: set of frequent (closed) generalized subsequences $F := \bigcup_{n \in \mathbb{N}} F_n$ of the sequences of S with support of at least minsup.
  $C := \{\{r\} \mid r \in R\}$ set of initial candidates,
  $n := 1$, $F_0 := \emptyset$.
  while $C \neq \emptyset$ or $F_{n-1} \neq \emptyset$ **do**
    compute $\sup_S(c) \quad \forall c \in C$ by counting the number of occurences of each c in S (one loop through S).
    $F_n := \{c \in C \mid \sup_S(c) \geq \text{minsup}\}$
    $C := F_n +_{n-1} F_n$ {compute new candidate sequences with length n+1}
    if $n = 2$ **then** {introduce wildcards}
      $C := C \cup \{(x, \star, y) \mid x, y \in F_{n-1}\}$
    else if $n > 2$ **then** {additional joins considering wildcards}

$$
\begin{aligned}
C \quad :=\ & C \\
& \cup\{x +_{n-2} y \mid (x, y) \in F_n \oplus_{n-2} F_{n-1},\ x_2 = \star\} \\
& \cup\{x +_{n-2} y \mid (x, y) \in F_{n-1} \oplus_{n-2} F_n,\ y_{\text{last}-1} = \star\} \\
& \cup\{x +_{n-3} y \mid (x, y) \in F_{n-1} \oplus_{n-3} F_{n-1},\ x_2 = y_{\text{last}-1} = \star\}
\end{aligned}
$$

    end if
    $n := n + 1$
  end while

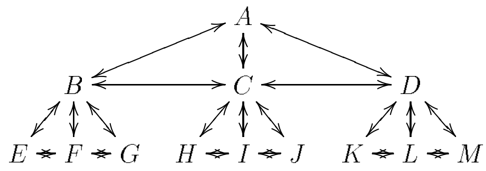
As algorithms of the apriori type return all subsequences of the frequent sequences found, one often prunes the result set by removing all subsequences of a frequent sequence contained in the result set, thus retaining only the "maximal" subsequences:

$$F := \{c \in F \mid \nexists d \in F : c < d\}$$

For generalized subsequences the algorithm also returns all generalizations of all subsequences found. Reasonably one prunes the result set further, by removing all generalizations of a sequence contained in the result set, thus retaining only the "most concrete" subsequences:

$$F := \{c \in F \mid \nexists d \in F : c \vdash d\}$$

We call these two pruning steps *subsequence pruning* and *generalization pruning*, respectively.



(a) site graph

nr	path
1	ABEF(EB)CHIJ
2	ACBE(BC)HI(HC)D
3	BCJ(C)HI
4	ABG(B)E(B)CH(C)I(C)D
5	ABEFG(FEB)CH(C)JI
6	ACJ(C)D(C)B(C)HI
7	BEFG(FEB)CHIJ(IHC)DKLM
8	ABF(B)CIH(I)J
9	ADK(D)L(D)AB(A)CHI
10	ABEFG(FEBA)CJ(C)HI(HC)D
11	ABCD(C)HIJ(IHCD)M
12	CBF(BC)H(C)DK(DC)I(CDKDCHCB)E

(b) anaylzed paths

Figure 1: Example web site and example set of paths.

4 Example and application

Figure 1 shows a simple example web site and some paths travelled on the site. Looking for ordinary frequent subsequences by applying the apriori algorithm for sequences (algorithm 1) does not give very useful results here: one finds the sequences CHI with a support of 8/12 and BCH with a support of 7/12. The first sequence containing more than three resources appears at support 5/12: EBCH.

Searching for frequent generalized sequences with algorithm 2 results in the set of three sequences with high support: B$\star$C$\star$H$\star$I with support 12/12 and two lightly more specialized sequences B$\star$CH$\star$I and BC$\star$H$\star$I with support 11/12 and 10/12 respectively. Of course, the algorithm finds all literal subsequences of these sequences as well as all more general sequences (like B$\star$H$\star$I etc.), but these less useful subsequences are pruned by the two pruning steps (subsequence pruning and generalization pruning) presented at the end of section 3. — Already this simple example gives some insights into the good properties of generalized sequences: first, they are more robust than ordinary sequences against artefacts coming from navigation path construction steps; second, they can cope with local deviations of the navigation paths, thus resulting in longer paths with higher support values, i.e. they better sketch user navigational behaviour in the large, contrary to local descriptions by ordinary subsequences.

As the retrieval of frequent (generalized) subsequences is the hard part of the generation of association rules, we can easily apply our algorithm to find associtation rules with prescribed minimal support and confidence. An *association rule* is (described by) a pair of (generalized) sequences $x, y \in R^{\text{gen}}$ with the meaning that if x (the body of the rule) has occured then — under conditions to be explained in the following — y (the head of the rule) will occur, too, where occurence is related to the underlying list S of sequences. We suggest different interpretations of the rule notation that all have their origin in the web site traversal behaviour of users as reconstructed via path completion and depicted in S. First, $x \to y \cong x +_1 y = (x_1, \ldots, x_{\text{last}} = y_1, \ldots, y_{\text{last}})$ which best corresponds to the usage of ordinary navigation paths. Second, $x \rightsquigarrow y \cong x \star y = (x_1, \ldots, x_{\text{last}}, \star, y_1, \ldots, y_{\text{last}})$, i.e., a wildcard is used to combine x and y. Both cases can be handled with the tools described so far. In addition to

$$\begin{aligned} \sup_S(x \to y) &:= \sup_S(x +_1 y) \quad \text{or} \\ \sup_S(x \rightsquigarrow y) &:= \sup_S(x \star y) \end{aligned}$$

we need the confidence

$$\begin{aligned} \operatorname{conf}_S(x \to y) &:= \tfrac{\sup_S(x+_1 y)}{\sup_S(x)} \quad \text{or} \\ \operatorname{conf}_S(x \rightsquigarrow y) &:= \tfrac{\sup_S(x \star y)}{\sup_S(x)} \end{aligned}$$

a number that counts the occurence of $x +_1 y$ or $x \star y$ given x.

From early papers on web usage mining, the idea of feeding back the usage information extracted from the logfiles to the hyperlink structure of the underlying website has been suggested as an application of the results found by various data analysis tasks (see Yan et al. (1996)). Recently this idea has been revived by the name of *recommender system* making use of frequent item sets and association rules (see Mobasher (2000)). The paths of active users are compared to the left sides (the bodies) of a rule set previously extracted from the logfiles and (parts of) the right sides (the heads) of the matching rules with highest confidence are recommended via dynamically included direct hyperlinks.

Using only sets of resources (and not sequences) as the base for recommendations has the drawback of neglecting the order of the navigation patterns, and, thus, may result in directing users back to resources, they might no longer have an interest in. On the other hand, using ordinary subsequences as base for recommendations retains the order information, but only catches local navigational behaviour. Generalized subsequences, i.e. path fragments, combine the strengths of the two methods, retaining order and not being bound to local behaviour (by allowing deviations).

Let us go back to our simple example from above and look at user 9. Imagine he has already done ADK(D)L(D)AB(A)C. Using frequent ordinary subsequences we cannot recommend a next resource, because no subsequence of the frequent literal subsequences (CHI, BCH, BCHI and EBCH) can be found in his partial path. But the subsequence B$\star$C of the frequent generalized subsequence B$\star$C$\star$H$\star$I matches a subsequence of the tail B(A)C of his partial path. Thus, using the assocation rules B$\star$C$\rightsquigarrow$H and B$\star$C$\rightsquigarrow$I (both with support and confidence 1) we can recommend H and I for subsequent browsing, exactly the resources, he visits afterwards.

5 Outlook

Path fragments described by frequent closed generalized subsequences are ideal candidates to describe the user navigation path space, and thus the basis of distance computations of user paths, which in turn are necessary for clustering user paths. We will report about user path clustering resting upon path fragments in an upcoming paper.

References

AGRAWAL, R. and SRIKANT, R. (1994): Fast Algorithms for Mining Association Rules. In: Bocca, J.B., Jarke, M., and Zaniolo, C. (eds.): *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94),* September 12-15, 1994, Santiago de Chile, Morgan Kaufmann, Chile, 487–499.

AGRAWAL, R. and SRIKANT, R. (1995): Mining Sequential Patterns. In: Yu, P.S., and Chen, A.L.P. (eds.): *Proceedings of the Eleventh International Conference on Data Engineering,* March

6-10, 1995, Taipei, Taiwan, IEEE Computer Society, 3–14.

BORGES, J. and LEVENE, M. (1998): Mining Association Rules in Hypertext Databases. In: Agrawal, R. (ed.): *Proceedings / The Fourth International Conference on Knowledge Discovery and Data Mining,* August 27 - 31, 1998, New York, New York, Menlo Park, Calif., 149–153.

BORGES, J. and LEVENE, M. (1999): Data Mining of User Navigation Patterns. In: *Proceedings of the Workshop on Web Usage Analysis and User Profiling (WEBKDD'99),* August 15, 1999, San Diego, CA, Springer, 31–36.

BUECHNER, A.G., BAUMGARTEN, M., ANAND, S.S., MULVENNA, M.D., and HUGHES, J.G. (1999): Navigation Pattern Discovery from Internet Data. In: *Proceedings of the Workshop on Web Usage Analysis and User Profiling (WEBKDD'99),* August 15, 1999, San Diego, CA, Springer, 25-30.

CHEN, M.-S., PARK, J.S., and YU, P.S. (1996): Data Mining for Path Traversal Patterns in a Web Environment. In: *Proceedings of the 16th International Conference on Distributed Computing Systems (ICDCS),* May 27-30, 1996, Hong Kong, IEEE Computer Society, 385–392.

CHEN, M.-S., PARK, J.S., and YU, P.S. (1998): Efficient Data Mining for Path Traversal Patterns. *IEEE Transactions on Knowledge & Data Engineering 10/2 (1998),* 209–221.

COOLEY, R., MOBASHER, B., and SRIVASTAVA, J. (1999): Web Mining: Information and Pattern Discovery on the World Wide Web. In: *9th International Conference on Tools with Artificial Intelligence (ICTAI '97),* November 3-8, 1997, Newport Beach, CA.

MANNILA, H., and TOIVONEN, H. (1996): Discovering generalized episodes using minimal occurences. In: *The Second International Conference on Knowledge Discovery and Data Mining (KDD '96),* Portland, Oregon, August 2-4 1996, 146–151.

MOBASHER, B. (2000): Mining Web Usage Data for Automatic Site Personalization. To appear in *Studies in Classification, Data Analysis, and Knowledge Organization 2000.*

SPILIOPOULOU, M. and FAULSTICH, L.C. (1998): WUM: A Tool for Web Utiliziation Analysis. In: Atzeni, P., Mendelzon, A., and Mecca, G. (eds.): *The World Wide Web and Databases, International Workshop WebDB'98,* Valencia, Spain, March 27-28, 1998, LNCS 1590, Springer, 184–203.

SPILIOPOULOU, M., FAULSTICH, L.C., and WINKLER, K. (1999): A Data Miner Analyzing the Navigational Behaviour of Web Users. In: *Proc. of the Workshop on Machine Learning in User Modelling of the ACAI'99 Int. Conf.,* Creta, Greece, July 1999.

SPILIOPOULOU, M. (1999): The Laborious Way from Data Mining to Web Mining. *Int. Journal of Comp. Sys., Sci. & Eng. 14 (1999),* Special Issue on "Semantics of the Web", 113-126.

SRIKANT, R. and and AGRAWAL, R. (1996): Mining Sequential Patterns: Generalizations and Performance Improvements. In: Apers, P.M.G., Bouzeghoub, M., and Gardarin, G. (eds.): *Advances in Database Technology - EDBT'96, 5th International Conference on Extending Database Technology,* Avignon, France, March 25-29, 1996, Proceedings. LNCS 1057, Springer.

VIVEROS, M.S., ELO-DEAN, S., WRIGHT, M.A., and DURI, S.S. (1997): Visitor's Behaviour: Mining Web Servers. In: *Proceedings of the 1st International Conference on the Practical Application of Knowledge Discovery and Data Mining,* Blackpool 1997, 257–269.

YAN, T.W., JACOBSEN, M., GARCIA-MOLINA, H., and DAYAL, U. (1996): From User Access Patterns to Dynamic Hypertext Linking. In: *Fifth International World Wide Web Conference* May 6-10, 1996, Paris, France.