Mining Generalized Association Rules for Sequential and Path Data

Wolfgang Gaul Lars Schmidt-Thieme

Institut für Entscheidungstheorie und Unternehmensforschung,
University of Karlsruhe, D-76128 Karlsruhe, Germany
{Wolfgang.Gaul, Lars.Schmidt-Thieme}@wiwi.uni-karlsruhe.de
Phone +49 / 721-608 3726, Fax +49 / 721-608 7765

Abstract

While association rules for set data use and describe relations between parts of set valued objects completely, association rules for sequential data are restricted by specific interpretations of the subsequence relation: contiguous subsequences describe local features of a sequence valued object, noncontiguous subsequences its global features. We model both types of features with generalized subsequences that describe local deviations by wildcards, and present a new algorithm of Apriori type for mining all generalized subsequences with prescribed minimum support from a given database of sequences. Furthermore we show that the given algorithm automatically takes into account an eventually underlying graph structure, i.e., is applicable for path data also.

1 Introduction

Classical association rules describe dependencies between subsets of a large sample of set valued objects, e.g., market baskets. A typical association rule states that if a certain subset occurs in a set, then another subset is likely to occur in the same set (with estimated probabilities for the applicability and the strictness of the rule). For set valued objects subsets are natural substructures to consider.

To sequential data association rules can be applied in different ways. One can forget the sequence structure and map sequences on the sets of their elements, but looses ordering information. Or one looks for sequential association rules, i.e., pairs of subsequences that occur in order in a sequence.

Here two different notions of substructures can be found in the literature: contiguous subsequences of elements, that occur one after the other in a sequence, and non-contiguous subsequences, where between two elements arbitrary noise may be interspersed. While the former describes local features of sequences, the latter describes global features of sequences. We combine both types of descriptions in generalized subsequences where possible noise is explicitly marked by a wildcard element.

The hard part of the computation of association rules is the computation of frequent sets or subsequences. Frequent subsets of set valued objects can be mined by the standard Apriori algorithm (see Agrawal and Srikant (1994)). Frequent *contiguous or non-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. Several algorithms exist to mine frequent *generalized subsequences of a specified type* (called templates, i.e., subsequences with prescribed positions of wildcards, see, e.g., Spiliopoulou (1999)). Other authors following a broader approach have constructed algorithms to find frequent subsequences of objects 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 minimum support) as needed for association rule analysis is still missing. In this paper we describe a new algorithm that fills this gap.

If sequence data describes paths on a graph an algorithm mining frequent subpaths should take advantage of the underlying graph structure, i.e., only consider paths and no sequences of non-connected vertices. We show that our algorithm has this property automatically (by using a suitable join operator) and thus also solves the problem of mining frequent subpaths.

2 Formal background

Definition 1. Let R be an arbitrary finite set of (non-interpreted) items and $R^\star := \bigcup_{i\in\mathbb{N}} R^i$ the set of finite sequences of elements of R (with $\emptyset$ as the empty sequence). For a sequence $x \in R^\star$ the *length* $|x|$ is the number of symbols in the sequence ($|x| := n$ for $x \in R^n$, $|\emptyset| := 0$).

For $x, y \in R^\star$ we say that x *is a contiguous subsequence of* y ($x \leq_{\text{ct}} y$), if there is an index $i \in \{0, \ldots, |y| - |x|\}$ with $x_j = y_{i+j} \quad \forall j = 1, \ldots, |x|$.

We say that x *is a non-contiguous subsequence of* y ($x \leq_{\text{nct}} y$), if there is a strictly increasing map $i : \{1, \ldots, |x|\} \to \{1, \ldots, |y|\}$ with $x_j = y_{i(j)} \quad \forall j =$

$1, \ldots, |x|$.

We develop the Apriori algorithm for sequences in parallel for both subsequence types and use the neutral symbol $\leq$ to denote one of the two subsequence relations $\leq_{\text{ct}}$ or $\leq_{\text{nct}}$. x is called a *strict subsequence of* y ($x < y$), if it is a subsequence of y but not equal to y ($x \leq y \wedge x \neq y$).

Definition 2. A pair of sequences $x, y \in R^\star$ *overlaps on* $k \in \mathbb{N}$ *elements*, if the last k elements of x are equal to the first k elements of y ($x_{|x|-k+i} = y_i \quad \forall i = 1, \ldots k$). 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

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

Note that any two sequences 0-overlap and the 0-telescoped concatenation of two sequences is just their arrangement one behind the other.

Definition 3. 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 as the *set of k-telescoped sequences of X and Y*:

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

Now let S be a finite (multi)set of such sequences $x \in R^\star$ representing a given database of sequences (allowing multiplicities if the same sequence is contained several times in the database).

Definition 4. 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 (multi)set of sequences S means to find all sequences $x \in R^\star$ with at least a given minimum 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 minimum 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 (Agrawal and Srikant 1995)

Require: set of items R, (multi)set S of (finite) sequences of elements of R, minimum support value minsup $\in \mathbb{R}^+$.

Ensure: set of frequent subsequences $F := \bigcup_{n \in \mathbb{N}, n \geq 1} F_n$ 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 occurrences 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

This adaption of the classical Apriori algorithm for sets (see Agrawal and Srikant (1994)) to sequences was first published by Agrawal and Srikant (1995) (with modifications by Srikant and Agrawal (1996)). It is given here merely for the purpose of comparison with our Algorithm 2 adapted for generalized sequences below. — Depending on the subsequence relation used (contiguous or non-contiguous subsequences), the corresponding method has to be used to count support values. Thus the Apriori algorithm adapted for sequences yields all frequent contiguous or all non-contiguous subsequences of sequences of S.

We state some additional results for the special case of sequences describing paths on a graph.

Definition 5. Let $G = (R, E)$ be a directed graph with vertices R and edges $E \subseteq R \times R$. A sequence $x \in R^\star$ of vertices is called a *path*, if every two consecutive elements are linked by an edge, i.e., $(x_i, x_{i+1}) \in E$ for all $i = 1, \ldots, |x| - 1$. We denote as *set of paths of* G the set $R^\star|_G := \{x \in R^\star \mid x \text{ is a path}\}$.

Lemma 1 (Path Construction Lemma). *If $x, y \in R^\star|_G$ are paths on a graph G of length $|x|, |y| \geq 2$ k-overlapping with $k \geq 1$, then their k-telescoped concatenation $x +_k y$ again is a path.*

Proof. Let $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_m)$ and $z = x +_k y = (z_1, \ldots, z_{n+m-k})$. Then for $i = 1, \ldots, n$ it is $z_i = x_i$ and thus for $i \in I_1 := \{1, \ldots, n-1\}$: $(z_i, z_{i+1}) = (x_i, x_{i+1}) \in E$ as x is a path. For $i = n-k+1, \ldots, n+m-k$ it is $z_i = y_{i-n+k+1}$ and thus for $i \in I_2 := \{n-k+1, \ldots, n+m-k-1\}$: $(z_i, z_{i+1}) = (y_{i-n+k-1}, y_{i-n+k}) \in E$ as y is a path. But as $k \geq 1$ it is $n - k \leq n - 1$ and thus $I_1 \cup I_2 = \{1, \ldots, n+m-k-1\}$, i.e., z is a path. □

Due to the path construction lemma candidates of length greater or equal 2 automatically are paths again, so there are no further checks afforded. Only the first join step for 0-overlapping pairs of sequences of length 1, i.e., pairs of nodes of the graph, could be simplified by considering only nodes that are linked by an edge. But as in reasonable implementations the support values of sequences of length 2 are counted in a two-dimensional array, this circumstance allows no algorithmic improvement.

3 Mining frequent generalized subsequences

Definition 6. 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*, so 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. 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 $x \in (R \cup \{\star\})^n$.

Definition 7. 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 increasing.

4. m is even strictly increasing at places where its image does not belong to a wildcard ($m(i) = m(i+1) \Rightarrow y_{m(i)} = \star$).

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 $\star\text{A}\star \vdash \text{AA}$ with the two matchings $m_1 : 1 \mapsto 1, 2 \mapsto 2$ and $m_2 : 1 \mapsto 2, 2 \mapsto 3$. 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 generalizing a subsequence of it (i.e. A $\star$ C $\vdash$ ABC and ABC $\leq$ ABCD). (For simplicity of notation we omit parentheses and commas in example sequences and just write ABC instead of (A,B,C).)

Definition 8. Again, let S be a finite (multi)set of ordinary sequences. 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*:

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

Mining frequent generalized subsequences is the label for the task of finding all generalized sequences with at least a given minimum support. As subsequences actually are, what we are looking for, we can narrow our view to *closed generalized subsequences*, i.e. generalized subsequences without leading or trailing wildcard ($x \in R^{\text{gen}}$ with $x_1, x_{|x|} \in R$).

At present no general algorithm for finding all frequent generalized subsequences in a (multi)set of sequences is known. We present a modification of the Apriori algorithm for sequences to generalized sequences. The idea is rather straightforward. As we restrict ourselves to 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. Adjacent wildcards are not allowed, therefore we obtain every closed generalized sequence of length $n+1$ (for $n \geq 3$) as the junction of two overlapping closed generalized sequences of the kind described in table 1.

More formally we state:

	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.

Lemma 2 (Generalized Sequence Construction). *For any closed generalized sequence $z \in R^{\text{gen}}$ there are closed generalized sequences $x, y \in R^{\text{gen}}$ with*

- *x and y are shorter than z ($|x|, |y| < z|$) and*
- *z can be constructed from x and y (i.e., x and y are k-overlapping (for a suitable k) with $z = x +_k y$).*

Proof. Let $z = (z_1, \ldots, z_n)$. Depending on z_2 and z_{n-1} being in R or being a wildcard we distinguish four cases: (1) If $z_2, z_{n-1} \in R$ let $x = (z_1, \ldots, z_{n-1})$, $y = (z_2, \ldots, z_n)$ and $k = n - 1$. x and y are closed by assumption and obviously overlap on k elements and yield z. (2) If $z_2 \in R, z_{n-1} = \star$ let $x = (z_1, \ldots, z_{n-2})$, $y = (z_2, \ldots, z_n)$ and $k = n - 2$. As $z_{n-1} = \star$ and no adjacent wildcards are allowed, $z_{n-2} \neq \star$, i.e. x is closed. y is closed by assumption and again x and y obviously overlap on k elements and yield z. (3) If $z_2 = \star, z_{n-1} \in R$ let $x = (z_1, \ldots, z_{n-1})$, $y = (z_3, \ldots, z_n)$ and $k = n - 2$. As $z_2 = \star$ and no adjacent wildcards are allowed, $z_3 \neq \star$, i.e. y is closed. x is closed by assumption and again x and y obviously overlap on k elements and yield z. (4) Finally if $z_2, z_{n-1} = \star$ let $x = (z_1, \ldots, z_{n-2})$, $y = (z_3, \ldots, z_n)$ and $k = n-3$. As $z_{n-1} = \star$ and no adjacent wildcards are allowed, $z_{n-2} \neq \star$, i.e. x is closed. As $z_2 = \star$ and no adjacent wildcards are allowed, $z_3 \neq \star$, i.e. y is closed. x and y obviously overlap on k elements and yield z. □

We simply 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 previous step, 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 comparisons. Obviously, the computation of the support values of the candidate generalized sequences also has to be modified. The performance characteristics 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.

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 of the set F of all frequent 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.

Our additional results for the special case of sequences describing paths on a graph easily carry over to generalized sequences.

Definition 9. Let $G = (R, E)$ be a directed graph with vertices R and edges $E \subseteq R \times R$. A generalized sequence $x \in R^{\text{gen}}$ of vertices is called a *path fragment*, if there are replacements for the wildcards that yield a path on G, i.e., for all $i = 1, \ldots, |x| - 1$ with $x_i \neq \star$:

- if $x_{i+1} \neq \star$ then $(x_i, x_{i+1}) \in E$.
- if $x_{i+1} = \star$ then there is a path $p^i \in R^\star$ connecting x_i and x_{i+2} (i.e., $p^i = (p^i_1, \ldots, p^i_n)$ with $p^i_1 = x_i$, $p^i_n = x_{i+2}$ and $(p^i_j, p^i_{j+1}) \in E$ for all $j = 1, \ldots, n-1$).

We denote as *set of path fragments of* G the set $R^{\text{gen}}|_G := \{x \in R^{\text{gen}} \mid x \text{ is a path fragment}\}$.

Lemma 3 (Path Fragment Construction Lemma). *If $x, y \in R^{\text{gen}}|_G$ are path fragements on a graph G of length $|x|, |y| \geq 2$ k-overlapping with $k \geq 1$, then their k-telescoped concatenation $x +_k y$ again is a path fragment on g.*

The proof is exactly the same as for the Path Construction Lemma for ordinary sequences given before. As for ordinary sequences the lemma guarantees that only path fragments are constructed during the candidate generation process.

Algorithm 2 Apriori algorithm adapted for generalized sequences

Require: set of items R, (multi)set S of (finite) sequences of elements of R, minimum 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 $\sup_S(c)$ $\forall c \in C$ by counting the number of occurrences 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}
    $C := C$
      $\cup\{x +_{n-2} y \mid (x, y) \in F_n \oplus_{n-2} F_{n-1}, x_2 = \star\}$
      $\cup\{x +_{n-2} y \mid (x, y) \in F_{n-1} \oplus_{n-2} F_n, y_{|y|-1} = \star\}$
      $\cup\{x +_{n-3} y \mid (x, y) \in F_{n-1} \oplus_{n-3} F_{n-1}, x_2 = y_{|y|-1} = \star\}$
  end if
  $n := n + 1$
end while

4 Generalized association rules

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 association rules for generalized sequences with prescribed minimum support and confidence.

Definition 10. In analogy to ordinary association rules between sets a *contiguous generalized association rule* is (described by) a pair of 1-overlapping (generalized) sequences $x, y \in R^{\text{gen}}$ (written $x \to y$). One defines the *support of an association rule* $x \to y$ as the support of the concatenated sequence $x +_1 y := (x_1, \ldots, x_{|x|} = y_1, \ldots, y_{|y|})$ and its *confidence* as the fraction of the sequences containing $x +_1 y$ of the sequences containing x:

$$\begin{aligned} \sup_S(x \to y) &:= \sup_S(x +_1 y) \\ \text{conf}_S(x \to y) &:= \frac{\sup_S(x \to y)}{\sup_S x} \end{aligned}$$

Speaking of association rules one has their interpretation as fuzzy rules in mind, i.e. that if the body x of the rule has occurred in a sequence, then the

occurrence of x is continued by the head y of the rule, where occurrence is related to sequences from the underlying set S. The support gives a measure for the applicability of the rule, i.e. the overall percentage of sequences where it holds, while the confidence gives a measure for the strictness with which the rule holds, i.e., in what percentage of sequences that it is applicable to it holds.

Finding all association rules with a given minimum support and confidence means nothing else but finding all frequent sequences with at least the given minimum support and then trying the different splits of the found frequent sequences and checking the confidence of the resulting rules. — Please note that it is crucial for this application that the algorithm which finds the frequent subsequences also finds all subsequences of every subsequence returned and accordingly already has computed all support values needed.

The definition of an association rule can be extended to generalized sequences without any modification. But using generalized sequences opens an additional possibility:

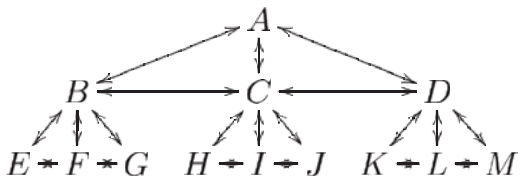
Definition 11. A *non-contiguous generalized association rule* is (described by) a pair of generalized sequences x, y (written $x \rightsquigarrow y$). Its support and confidence are defined as follows:

$$\begin{aligned} \sup_S(x \rightsquigarrow y) &:= \sup_S(x_1, \ldots, x_n, \star, y_1, \ldots, y_m) \\ \mathrm{conf}_S(x \rightsquigarrow y) &:= \frac{\sup_S(x \rightsquigarrow y)}{\sup_S x} \end{aligned}$$

5 Example and experiments

We give a simple example and an experiment for web usage mining data. Figure 1 shows an example web site and some paths traveled on the site. Looking for ordinary frequent subsequences by applying the Apriori algorithm for sequences (algorithm 1) does not render results all too useful in this case: one will find 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⋆C⋆H⋆I with support 12/12 and two slightly more specialized sequences B⋆CH⋆I and BC⋆H⋆I with support 11/12 and 10/12 respectively. Naturally, the algorithm finds all the literal subsequences of these sequences as well as all the more general sequences (like B⋆H⋆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.



(a) site graph

nr	path
1	ABEFEBCHIJ
2	ACBEBCHIHCD
3	BCJCHI
4	ABGBEBCHCICD
5	ABEFGFEBCHCJI
6	ACJCDCBCHI
7	BEFGFEBCHIJIHCDKLM
8	ABFBCIHIJ
9	ADKDLDABACHI
10	ABEFGFEBACJCHIHCD
11	ABCDCHIJIHCDM
12	CBFBCHCDKDCICDKDCHCBE

(b) user paths

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

Thus generalized sequences are able to cope with local deviations of the navigation paths, resulting in longer path fragments with higher support values, i.e. they render improved sketches of user navigational behavior in the large, contrary to local descriptions by ordinary contiguous subsequences.

Let us use the generalized association rules that can be derived from the frequent generalized sequences to recommend users further resources to browse. Let us look at user 9 and imagine he has already visited ADKDLDABAC. 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 browsing history. But the subsequence B⋆C of the frequent generalized subsequence B⋆C⋆H⋆I matches a subsequence of the tail BAC of his browsing history. Thus, using the association rules B⋆C⇝H and B⋆C⇝I (both with support and confidence 1), we can recommend H and I for subsequent browsing, exactly the resources he

does in fact visit afterwards.

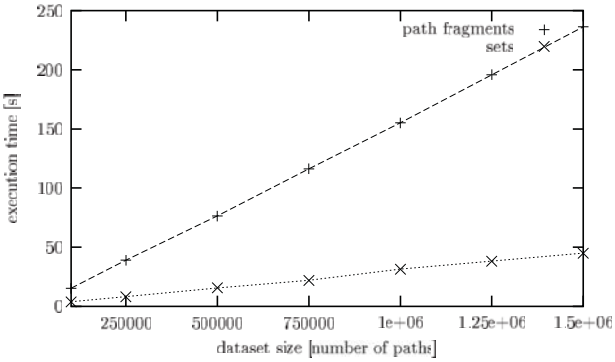
We tested our algorithm for generalized sequences with synthetic data created by randomly instantiating a set of sequence templates. Each template describes the navigational behavior of a user segment on a given website by a generalized sequence and a distribution of the lengths of the replacements of the wildcards as well as its relative size by a segment weight. A navigation template is instantiated by randomly replacing the wildcards by concrete sequences of resources.

Figure 2 shows the experimental results. For a given site of 100 resources we created datasets of different sizes from a set of 5 templates with relative segment sizes of 0.3, 0.3, 0.2, 0.1, and 0.1, respectively, and $\mathbb{N}(4,2)$-distributed replacement lengths (independent of the segments). — We implemented the Apriori algorithm for sets and our adaptation for generalized sequences using prefix trees (see, e.g., Mueller (1995)) in Java. All experiments were run on the IBM JVM 1.3 on an Athlon-600 Linux-PC with 256 MB RAM.

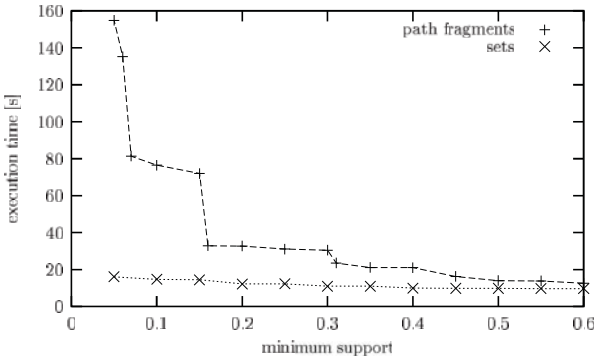
Figure 2a shows the execution time for datasets of different sizes for both algorithms. As expected execution times increase linear within the size of the dataset. In figure 2b the dependency of execution times on minimum support values is depicted for a dataset size of 500.000 paths (similar results can be obtained for other dataset sizes). In the case of the algorithm for generalized sequences steps of the performance curve at support values that correspond to the weights of the user segments (0.1, 0.2, and 0.3) are clearly visible. Figure 2c shows the execution times the algorithm for generalized sequences spends on the individual passes (i.e., on mining the subsequences with 1, 2, etc. non-wildcard elements). As support values for single items are computed in parallel with reading the data, the execution time for the first pass includes the time for file I/O. For a minimum support of 0.2 most items are not frequent, so the algorithm does not have much to check. For minimum support of 0.1 all items are frequent and the main part of the computation time is spent on sequences with two non-wildcard elements (20000 candidates have to be checked). For an even lower minimum support of 0.05 frequent subsequences with 2 non-wildcard elements are common enough to yield a large pool of candidates with 3 non-wildcard elements (ca. 15000 candidates).

6 Outlook

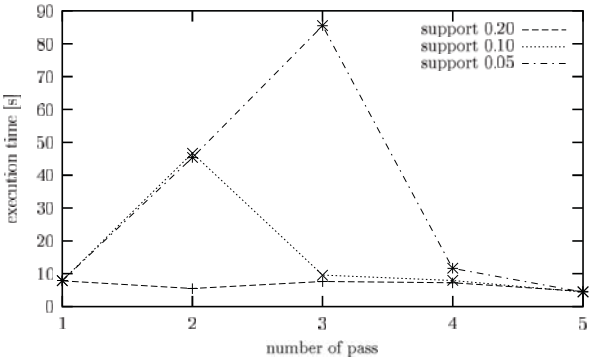
Existing implementations of the Apriori algorithm adapted for contiguous and non-contiguous sequences can easily be extended to cover generalized



(a) Execution time depending on dataset size (minimum support: 0.1).



(b) Execution time depending on minimum support (size of dataset: 500000).



(c) Execution time depending on number of pass (size of dataset: 500000).

Figure 2: Experimental results.

sequences describing contiguous and non-contiguous parts of a sequence in a single pattern as explained in this paper.

Generalized sequences can be interpreted as non-contiguous subsequences of contiguous subsequences, i.e., as nested structures of second order. This interpretation opens the application of Apriori type algorithms to a huge range of new data and pattern structures. A unified framework for mining adequate substructures of such data will be presented in a forthcoming 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.

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.

MANNILA, H., and TOIVONEN, H. (1996): Discovering generalized episodes using minimal occurrences. In: *The Second International Con-*

ference on Knowledge Discovery and Data Mining (KDD '96), Portland, Oregon, August 2-4 1996, 146–151.

MUELLER, A. (1995): Fast Sequential and Parallel Algorithms for Association Rule Mining: A Comparison. Department of Computer Science, University of Maryland-College Park, CS-TR-3515.

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 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.