Fuzzy Sequential Pattern Mining In Incomplete Databases

Céline Fiot, Anne Laurent, Maguelonne Teisseire

Mathware & Soft Computing XX (2008)

Fuzzy Sequential Pattern Mining In Incomplete Databases

Céline Fiot, Anne Laurent and Maguelonne Teisseire
LIRMM, Univ. Montpellier 2 - CNRS
161 rue Ada - 34392 Montpellier, France
e-mail: {fiot,laurent,teisseire}@lirmm.fr

Abstract

Recent widening of data mining application areas have lead to new issues. For instance, frequent sequence discovery techniques that were developed for customer behaviour analysis are now applied to analyse industrial or biological databases. Thus frequent sequence mining algorithm must be adapted to handle particular characteristics of these data. Among these specificities one should consider numerical attributes and incomplete data. In this paper, we propose a method for discovering crisp or fuzzy sequential patterns from an incomplete database. This approach uses partial information contained in incomplete records, only temporary discarding the missing part of the record. Experiments run on various synthetic datasets show the validity of our proposal as well in terms of quality as in terms of the robustness to the rate of missing values.

1 Introduction

Over the last decade data mining applications have widened, drifting from their original areas. They thus confronte researchers with new issues. One of them is the adaptation of approaches first designed for a specific domain, with specific data, and now applied to other data, with special features.

One relevant example of this evolution are the applications of sequential pattern discovery techniques. This data mining technique aims at discovering knowlegde – frequent sequences – from temporal databases. First designed to analyze customer behaviors in supermarket, they are now used for mining industrial, textual, medical or even biological data. However these data are often imperfect with respect to the requirements for mining sequential patterns.

Indeed algorithms were designed to handle data that could be processed as binary ones – presence or absence of a product in a market basket. Furthermore these data were considered as complete ones, i.e. no attribute is unfilled. New algorithms were recently proposed to handle these new features. More specifically several approaches introduce fuzzy sequential pattern mining in order to handle

the quantitative information often contained in industrial databases.
Regarding data that contain missing values there only exist few approaches. In fact when such data are mined, they are most of the time deleted or replaced by a statistical value. However this leads to a loss of information or to biased knowledge.

To avoid such pre-processing in [8] we extended the principles originally developed by [2] in order to discover frequent sequences within databases containing values missing at random. This technique is based on the idea that only but all the available information should be used while the mining task. This was first applied to design an association rule method for mining incomplete databases [19] and some machine learning techniques [12]. This principle consists in only making use of available information (i.e. filled-in attributes) and ignoring missing information, without ignoring the whole involved record. Thus only partial complete databases are mined for each pattern and the whole dataset is used to discover all the patterns.

In this paper we recall the different principles we introduced in [8] to discover sequential patterns within time-stamped incomplete databases using the algorithm SPoID (Sequential Patterns for Incomplete Databases). Then we extend these definitions to handle both quantitative and incomplete data while mining for frequent sequences in real-world databases. There indeed exist approaches – based on fuzzy sets – for discovering sequential patterns in quantitative databases. However these techniques cannot handle missing values without specific additional preprocessing. Therefore we propose the algorithm F-SPoID (Fuzzy Sequential Patterns for Incomplete Data) that can be applied to discover frequent sequences within incomplete quantitative databases.

The remainder of the paper is organized as follows. Section 2 introduces the concepts linked to sequential pattern and fuzzy sequential pattern discovery and gives a brief overview of data mining in the presence of missing values. Section 3 details our approach to mine for crisp or fuzzy sequential patterns within incomplete data, and we describe our algorithm in Section 4. Section 5 is then dedicated to experiments that show the validity of our approach. Finally, we discuss further work opened by this proposal and we conclude in Section 6.

2 Sequences, Fuzziness and Missing Values

Sequential patterns are often introduced as an extension of association rules, initially proposed in [1]. They highlight correlations between database records as well as their temporal relationship. Even though these algorithms do not mine incomplete records contained in the database. These missing values must then be removed either by deletion or replacement. Quality of results then depends on this preprocessing. Moreover this step is often time-consuming. In order to reduce the preprocessing due to missing values and to improve the sequential pattern quality, we propose a method for sequential pattern mining within incomplete databases. This method is based on association rules approaches. In this section we first define the concepts linked to sequential pattern mining, then we detail our motivations

before introducing techniques that allow association rule mining handling missing values.

2.1 Sequential Patterns

Let $\mathcal{R}$ be a set of objects records where each record R consists of three information elements: an object-id, a record timestamp and a set of attributes/items in the record. Let $I = \{i_1, i_2, ..., i_m\}$ be a set of items or attributes. An *itemset* is a non-empty set of attributes i_k, denoted by $(i_1 i_2 \dots i_k)$. It is a non-ordered representation. A *sequence* s is a non-empty ordered list of itemsets s_p, denoted by $< s_1 s_2 ... s_p >$. A *n-sequence* is a sequence of n items (or of size n).

Example 1. *Let us consider an example of market basket analysis. The object is a customer, and records are the transactions made by this customer. Timestamps are the date of transactions. If a customer purchases products e, a, k, u, and f according to the sequence $s =< (e)\ (a\ k)\ (u)\ (f) >$, then all items of the sequence were bought separately, except products a and k which were purchased at the same time. In this example, s is a 5-sequence.*

A sequence $S =< s_1\ s_2 ... s_p >$ is a *subsequence* of another one $S' =<\ s'_1\ s'_2\ ...s'_m >$ if there are integers $l_1 < l_2 < ... < l_p$ such that $s_1 \subseteq s'_{l_1},\ s_2 \subseteq s'_{l_2}, ...,\ s_p \subseteq s'_{l_p}$.

Example 2. *The sequence $s' =<(a)\ (f)>$ is a subsequence of s because $(a) \subseteq (a\ k)$ and $(f) \subseteq (f)$. However, $<(a)\ (k)>$ is not a subsequence of s.*

All records from the same object o are grouped together and sorted in increasing order of their timestamp. They are called a data sequence. An object *supports* a sequence s if it is included within the data sequence of this object (s is a subsequence of the data sequence). The *frequency* of a sequence ($freq(s)$) is defined as the percentage of objects supporting s in the whole set of objects $\mathcal{O}$. In order to decide whether a sequence is frequent or not, a minimum frequency value ($minFreq$) is specified by the user and the sequence is said to be frequent if the condition $freq(s) \geq minFreq$ holds. A sequence that may be frequent is a *candidate sequence*. Given a database of object records, the problem of sequential pattern mining is to find all maximal sequences of which the frequency is greater than a specified threshold ($minFreq$) [2]. Each of these sequences represents a sequential pattern, also called a *maximal frequent sequence*.

Several extensions were proposed to consider incremental mining for sequential patterns [14], to handle numerical and quantitative values [5, 7, 11] or to generalize sequential patterns with respect to various temporal parameters (time-interval between events of a sequence, grouping several records into a single itemset...) [6, 15, 20]. However, no technique was proposed to deal with missing values while sequential pattern mining. For this reason, in the following sections, we propose an approach that can mine maximal frequent sequences from an incomplete sequence database.

2.2 Fuzzy Sequential Patterns

In order to allow for handling numerical or quantitative information several works proposed to partition each numerical attribute into several fuzzy sets. The quantitative database is thus converted into a membership degree database, which is then mined for fuzzy sequential patterns [5, 7, 11].

The item and itemset concepts have been redefined relative to classical sequential patterns. A **fuzzy item** is the association of one item and one corresponding fuzzy set. It is denoted by $[x, a]$ where x is the item (also called attribute) and a is the associated fuzzy set.

Example 3. $[candy, lot]$ *is a fuzzy item where lot is a fuzzy set defined by a membership function on the quantity universe of the possible purchases of the item candy.*

A **fuzzy itemset** is a set of fuzzy items. It can be denoted as a pair of sets (set of items, set of fuzzy sets associated to each item) or as a list of fuzzy items. We use the following notation: (X, A), where X is a set of items and A is a set of corresponding fuzzy sets.

Example 4. $(X, A) = ([candy, lot][soda, little])$ *is a fuzzy itemset and can also be denoted by* $((candy, soda)(lot, little))$.

One fuzzy itemset only contains one fuzzy item related to one single attribute. For example, the fuzzy itemset $([candy, lot][candy, little])$ is not a valid fuzzy itemset because it contains twice the attribute *candy*.

Last a **g-k-sequence** $S =< s_1 \cdots s_g >$ is a sequence constituted by g fuzzy itemsets $s = (X, A)$ grouping together k fuzzy items $[x, a]$.

Example 5. *The sequence* $S =< ([soda, lot][candy, lot])$ $([video\ games, little]) >$ *groups together 3 fuzzy items into 2 itemsets. It is a fuzzy* 2-3-*sequence.*

In the next sections of this article, we denote by $\mathcal{O}$ the set of objects and by $\mathcal{R}_o$ the set of records for one object o. Let $r[i]$ be the quantity value of attribute i in record r. Each attribute i is divided into fuzzy sets.

Depending on the data and the expert knowledge that is actually available, the fuzzy partitioning may be given by the expert or built automatically from the database by a clustering algorithm as presented in [9] and [10], all the more that some fuzzy clustering algorithm are robust to incomplete data [21].

The frequency of a fuzzy sequence $S = \langle s_1 \cdots s_k \rangle$ is then computed by the formula 2.2:

$$FFreq(S) = \frac{\sum_{o \in \mathcal{O}} \varphi(S, o)}{|\mathcal{O}|}$$

where $\varphi(S, o)$ gives the degree to which S is included into the object o data sequence. This degree is computed by considering the best appearance – i.e. the appearance with the highest degree – of the ordered list of itemsets of S. It is computed by the formula 2.2:

$$\varphi(S, o) = \underline{\perp}_{\varsigma \subseteq \zeta_o | S = \varsigma = <s_1 \ldots s_i \ldots s_k>} \overline{\top}_{s_1 \ldots s_k} (\overline{\top}_{j \in s_i} \mu_j(r(j)))$$

where k is the number of itemsets in S, ζ_o is the set of sequences included in the data sequence of object o and $\overline{\top}$ and $\underline{\perp}$ are the t-norm and t-conorm operators generalized to n-ary cases. Practically we use the Zadeh t-norm and t-conorm, min and max.

2.3 Missing Values and Sequential Patterns

Sequential patterns have mostly been applied and assessed to analyse customer behaviours in supermarket. Then they have been used in web usage analysis or in biological or medical areas. In the original context of the market basket analysis, missing values are rare or even unexisting, whereas in web logs or biological sequences some data are incomplete. Therefore techniques have been developped to handle these missing values while mining for frequent sequences.

Fuzzy sequential patterns have been proposed to make sequential patterns more suitable to real world databases, within the context of historically-stamped numerical data (sensor, scientific, demographic, monitoring, evolution phenomena data, etc.). However, this kind of databases often contain incomplete data, i.e. records containing unfilled attributes, or missing values, due to breakdowns or errors, for instance.

Different approaches are generally used for knowledge discovery in incomplete databases. Most of the time they consist of a preprocessing step that imput or delete missing values. Some techniques have also been developed for handling missing values within the mining step, more precisely for classification [13, 18, 23], clustering [21] or association rules mining [3, 4, 16, 17, 19].

Within the context of temporal knowledge discovery and specifically sequential pattern discovery, the common way of processing missing values consist of their deletion before the mining task. As this deletion may lead to an important loss of information, a proposal was done to handle incomplete data while mining for crisp sequential pattern [8]. But this work has not been developed for handling missing values in a quantitative database.

Therefore, we propose in this paper a fuzzy extension of our previous proposal SPoID, to make possible the discovery of fuzzy sequential patterns within incomplete databases. This method uses the whole dataset without deletion before the mining task: it uses partial and temporary disabling of incomplete data. All the records will then be used for discovering all the sequential patterns, but each of them is extracted from a partial dataset.

3 Fuzzy Sequential Patterns and Incomplete Data

As deleting incomplete records leads to an important loss of information we adapted an association rule [1] mining approach robust to missing values. In this section we describe our methods, SPoID designed for crisp sequential patterns, and F-SPoID, designed for fuzzy sequential patterns. Both approaches are based on the same principles, instigated by the Robust Association Rules algorithm (RAR), proposed by [19].

3.1 Principle

The main idea of our approach, as the one of the RAR method, is incomplete elements disabling, within our context, incomplete sequences. While the RAR algorithm only regards complete records for association rules mining, we propose to only take into account the complete data sequences for each candidate sequence. In other words, when an incomplete data sequence is scanned, only filled-in time-stamped attributes will be considered for frequency calculation. Thus each candidate sequence will be considered as a frequent sequence on a partial database, but the whole dataset will be used to find the whole set of frequent sequences.

Let us consider a candidate sequence S, the set $\mathcal{O}$ of objects in the database can be divided into three disjoint subsets (Figure 1): the set of data sequences supporting S, denoted by $\mathcal{O}_S$, the set of data sequences not supporting S, denoted by $\mathcal{O}_{\overline{S}}$, and the set of data sequences for which we do not know whether they support S or not, denoted by $\mathcal{O}^*_S$.

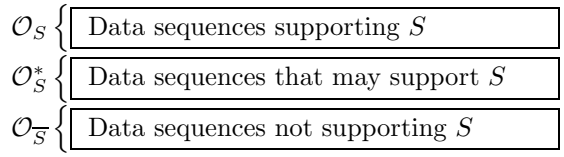


Figure 1: Partition of the database depending on S inclusion.

For each candidate sequence S, only the subsets $\mathcal{O}_{\overline{S}} \cup \mathcal{O}_S$ will be kept to determine wether the sequence S is frequent or not. This data sequence set represents the *valid database* for S as it only contains complete data sequences for the given candidate sequence S.

Constitution of a valid database leans on temporary disabling data sequences that contain missing values for items in the candidate sequence. A data sequence is *disabled* for a candidate sequence S if it is incomplete for S (i.e. we cannot decide whether it supports S or not). The set of data sequence disabled for a candidate sequence S is denoted by $Dis(S)$.

This implies to redefine the frequency calculation to take into account the database partial deactivation. The frequency definitions given in section 2 are then

modified in order to consider the valid database concept, and thus that only one part of the dataset is used for frequency calculation of a crisp or a fuzzy sequence.

3.2 SPoID: Crispy Dealing with Incomplete Data

Considering a crisp sequence S, the temporary disabling of the database is taken into account by computing the frequency over the valid database for S.

Definition 1. *The* frequency *of a sequence S is the rate of appearance of this sequence among the data sequences that can support it. It is defined as the ratio of the number of data sequences supporting S over the number of data sequences that surely include S or not (complete data sequences for S). It is given by:*

$$Freq(S) = \frac{|\mathcal{O}_S|}{|\mathcal{O}| - |Dis(S)|}$$

In [8], we proved that, considering minor restrictions, this frequency definition holds the antimonotonicity property of the support definition enonciated by [2]. So we can use the various properties described in [2] in order to implement the sequential mining algorithm within incomplete databases. However, the *frequency* concept must also be regarded taking into account the size of the valid database used to compute it. Therefore we define a *representativity* criterion and a minimum representativity threshold *minRep*, that must be satisfied: a valid database must be a significative sample of the whole dataset for a sequence S. Then a sequence is said to be *representative* if its representativity is greater than a minimum representativity value *minRep*.

Definition 2. *The* representativity *$Rep(S)$ of a sequence S is defined as the ratio of the number of data sequences that either include or cannot include S over the total number of data sequences in the whole dataset. It is given by:*

$$Rep(S) = \frac{|\mathcal{O}| - |Dis(S)|}{|\mathcal{O}|}$$

Eventually, to be kept as frequent, a candidate sequence must have a representativity greater than the minimum representativity threshold *minRep* and its frequency must be no less than the user-defined minimum threshold *minFreq*.

Example 6. *Let us consider the database given by Table 1, with a minimum frequency equal to 50%.*

O.	Seq.
O1	(*a b*) (*b c d*) (*b c e*)
O2	(*a*) (*b c*) (*b d*)
O3	(*a b*) (*b c*) (*b c d*)

Table 1: Complete database.

O.	Seq.
O1	(*a b*) (? ? *c*) (? *b c*)
O2	(*a*) (? *c*) (*b d*)
O3	(*a b*) (? *c*) (? ? *c*)

Table 2: Incomplete database.

The sequential patterns obtained are: <(a b)(b c d)>, <(a b)(b c)(b c)> and <(a)(b c)(b d)> from the complete database, Table 1. Now, let us consider the incomplete database given by Table 2.

Item a is certainly supported by the three objects, then its frequency is $freq(a) = 3/3 = 100\%$ *and its representativity is equal to 1. It is the same for items b and c. For item d,* $\mathcal{O}_{<d>} = \{O2\}$ *and* $Dis(< d >) = \{O1, O3\}$*, then* $freq(d)$ *= 1/(3-2) = 1 and* $rep(d)$ *= (3-2)/3 = 0.33. If minRep is 0.3, then* $rep(d) > minRep$ *and d is a frequent item. On the other hand, if minRep = 0.4, then* $rep(d) < minRep$ *and d is not a frequent item because the valid database regarded to compute its frequency is not significant enough.*

Let us consider minRep=0.3 and consider sequence $S =< (a\ b)(a\ b\ c\ d) >$*. It cannot be supported by one of the data sequence, because none of them contains an itemset composed of 4 items, either complete or not, then* $freq(S)$*=0.*

Now consider $S' =< (a\ b)(b\ c) >$*. It is supported by O1, it cannot be supported by O2 but maybe by O3. Then,* $\mathcal{O}_{S'} = \{O1\}$*,* $Dis(S') = \{O3\}$ *and* $\mathcal{O}_{\overline{S'}} = \{O2\}$*. This leads to* $freq(S')$*=1/(3-1)=50% and* $rep(S')$*=(3-1)/3 = 0.67.* S' *is then both representative and frequent.*

Applying this method, discovered patterns for minFreq=50% and minRep = 0.3 are: <(a b)(c)(b c)> and <(a)(c)(b d)>.

Even if the patterns extracted using this approximation of the frequency are not exactly the one obtained on the complete database, they are closer to the one we should get than the one discovered using the preprocessed dataset. Experiments detailed in section 5 show that there exists a value of the minimum representativity for which the algorithm SPoID extracts the whole set of sequential patterns of the complete database from an incomplete one.

3.3 F-SPoID: Fuzzy Sequential Patterns and Missing Values

This definition of the frequency handling missing values may now be extended to allow for incomplete data processing within the context of fuzzy sequential pattern discovery.

Considering the framework of fuzzy sequential patterns, $|\mathcal{O}_S|$ is computed on the same principle as in formula 2.2. Then to take into account that only one part of the dataset is complete for a given sequence S, we compute the frequency of S over this partial database, instead of considering the whole one.

So the frequency of a fuzzy sequence S is given by the formula:

$$FFreq(S) = \frac{\sum_{o \in \mathcal{O}_S} \varphi(S,o)}{|\mathcal{O}| - |Dis(S)|}$$

$$= \frac{\sum_{o \in \mathcal{O}} \overline{\bigsqcup}_{\varsigma \subseteq \zeta_o | S = \varsigma = < s_1 \ldots s_k >} \overline{\top}_{s_1 \ldots s_i \ldots s_k} (\overline{\top}_{j \in s_i} \mu_j(r(j)))}{|\mathcal{O}| - |Dis(S)|}$$

This definition is consistent with the crisp definition given in the previous subsection and also with the definition of the fuzzy frequency given for complete

database mining.

This frequency definition also holds the antimonotonicity property required to ensure completeness of results while mining for sequential patterns.

Example 7. *Let us consider the membership database, table 3, obtained from a quantitative datasets and some previously designed membership functions. This dataset contain three data sequences in which we search for the sequence* $S =< ([X_3, a])([X_2, b]) >$*, i.e. the sequence made of the fuzzy item* $[X_3, a]$ *followed by the fuzzy item* $[X_2, b]$*.*

		Fuzzy items									
		X_1		X_2			X_3			X_4	
		a	b	a	b	c	a	b	c	a	b
O1	r1	1			1		?	?		1	
	r2	1					**0.8**	0.2			1
	r3	0.5	0.5	0.25	**0.75**						
	r4				**1**					1	
O2	r1	1					**?**	?			
	r2			0.25	**0.75**					0.5	0.5
O3	r1		1					0.2	0.8		
	r2				0.5	0.5				0.4	0.6

Table 3: Example of an incomplete membership database.

$S =< ([X_3, a])([X_2, b]) >$ *is supported by* $O1$*, it cannot be supported by* $O3$ *but maybe by* $O2$*. Then,* $\mathcal{O}_S = \{O1\}$*,* $Dis(S) = \{O2\}$ *and* $\mathcal{O}_{\overline{S'}} = \{O3\}$*. This leads to*

$$freq(S) = \frac{\perp(\top(0.8, 0.75), \top(0.8, 1))}{3-1} = \frac{0.8}{2} = 0.4 \sim 40\%$$

and $rep(S')$*=(3-1)/3 = 0.67.* S' *is then both representative and frequent.*

4 Implementation

4.1 Algorithms

The algorithm we implemented for mining crisp or fuzzy sequences within incomplete data runs similarly to the generate-and-prune sequential pattern mining algorithms. It is described by Algorithm 1. It consists in generating all the candidate k-sequences from the frequent (k-1)-sequences. Then the database is scanned to count the number of data sequences that support each candidate sequence. The main difference stands in the counting of incomplete data sequences.

This counting step is described by the algorithm SPoID (Alg. 2) in the crisp case and by the algorithm F-SPoID (Alg. 3) in the fuzzy case. Both algorithms follow the same principle: for each candidate sequence S, for each object o,

- if the candidate sequence is found, the absolute value of the frequency is incremented by 1 – crisp version – or by $\varphi(S, o)$ – fuzzy mining,

- if the candidate sequence is not found nor a sequence with missing values that could be replaced to complete the candidate sequence, then the object does not support the candidate sequence. The absolute frequency is not incremented.

- an incomplete data sequence in which missing values could be replaced by items of the candidate sequence is found. In that case, the object is added to the disabled object set.

SPoID - **Input:** $|\mathcal{O}|$, sequence database, *minFreq*, minimum support
minRep, minimum representativity (user-defined or computed)
Ouput: $SPList$, frequent sequence list

$C \leftarrow \{i \in \mathcal{I}\}$; $k = 1$;
$F \leftarrow$ getFrepnRep(C,$minFreq$,$minRep$); $SPList$.add(F) ;
While ($C \neq \varnothing$) **do**
  k++ ; $C \leftarrow$ generate(F,k) ;
  For each candidate sequence $s \in C$ **do**
    For each object $o \in \mathcal{O}$ **do**
      [Search for s within S_o: in the crisp case computeFreq(s) is run as SpoID, in the fuzzy case computeFreq(s) is run as F-SpoID]
      $frequency(s)$+= computeFreq(s,o) ;
    End For
    $Freq(s) \leftarrow frequency(s)/(|\mathcal{O}| - |Dis(s)|)$; $Rep(s) \leftarrow |\mathcal{O}| - |Dis(s)|/|\mathcal{O}|$;
    If $((Freq(s) < minFreq)||(Rep(s) < minRep))$ **Then** prune(s) ; **End If**
  End For
  $SPList$.add(C) ;
End While
return SPList ;

Algorithm 1 – *Main algorithm.*

Once the whole dataset scanned, the absolute value of the frequency is divided by the substraction of the number of disabled objects to the number of objects in the database. The representativity is also computed. Then the pruning step is run to delete candidate sequences that are neither frequent nor representative.

SPoID - **Input:** S_o, a data sequence,
s a candidate sequence
Ouput: f, frequency of sequence and updated $Dis(S)$

If ($s \in S_o$) **Then**
  f=1 ; $Dis(s) \leftarrow Dis(s) \backslash o$;
Else
  If ($\tilde{s} \in S_o/\tilde{s}$ may be s) **Then**
    $Dis(s) \leftarrow Dis(s) \cup o$;
  End If
End If
return f ;

Algorithm 2 – *SPoID.*

F-SPoID - **Input:** S_o, a data sequence,
s a candidate sequence
Ouput: f, frequency of sequence and updated $Dis(S)$

If ($s \in S_o$) **Then**
  $f = \underline{\perp}_{\varsigma \subseteq \zeta_o | s=\varsigma=<s_1 \cdots s_k>} \overline{\top}_{s_i} (\overline{\top}_{j \in s_i} \mu_r(j))$;
  $Dis(s) \leftarrow Dis(s) \backslash o$;
Else
  If ($\tilde{s} \in S_o/\tilde{s}$ may be s) **Then**
    $Dis(s) \leftarrow Dis(s) \cup o$;
  End If
End If
return f ;

Algorithm 3 – *F-SPoID.*

The temporal complexity of this algorithm is, in the worse case, the same as the one of the algorithm TotallyFuzzy presented by [7] for mining fuzzy sequential patterns.

When considering classical sequential patterns an object supports a sequence or not. So the scan can stop as soon as the sequence is found within the record set of an object. On the contrary, (F-)SPoID searches for the best complete occurrence of a sequence for each object and each sequence. This leads to an exhaustive scanning of the record set, as performed for association rule mining.

A naive approach could be that for each candidate k-sequence (likely frequent sequence) all the database is scan to find its frequency, which would involve n^k scans of the database at most if n is the number frequent items. The only structure kept in memory would thus be the list of candidate sequences. However, this would be very inefficient, as the computational time would explode.

Therefore we use the same data structure as the one described by [7] enabling us to find all representations of all k-sequences in only k scans of the database. The computational time is also lower but the used memory space is increased. However, some optimizations have been implemented to bound this spatial complexity. We also added some property-based optimisation to avoid unnecessary scans to improve the efficiency of our approach.

4.2 Minimal Representativity and Margin of Error

In order to determine the representativity threshold one may consider statistical results. Statistics indeed use sampling techniques that allow to only consider a subpopulation subset to assess a proportion, satisfying an error interval with a sufficient confidence. These tools help to determine the optimal sampling size depending on the data distribution. Thus considering a random data distribution, [22] uses the Chernoff bound to set the minimal size of a random sample for association rule mining. This result was also proved theoretically and experimentally by [24].

We thus propose to use two kinds of representativity depending on the user needs: the minimum representativity threshold can be defined either by the user as a percentage of the dataset size, or it can be an absolute number of data sequences computed from statistics formula related to the data distribution and user-defined parameters for error and confidence level. However our experiments show that the optimal representativity threshold is not an absolute value but rather depends on the missing value rate of datasets.

5 Experiments

These experiments were carried out on a PC - Linux 2.6.7 OS, CPU 2.8 GHz with 512 MB of memory. The algorithm was implemented in Java on the PSP principle; PSP is a generate-and-prune algorithm developed for sequential pattern mining. In

particular, we used the Prefix-Tree structure to store the candidate and frequent sequences.
Synthetic datasets were randomly generated. Then some items were randomly replaced by missing values. Sequential patterns were extracted from the complete database and from the preprocessed incomplete ones (i.e. incomplete databases in which incomplete records have been deleted). Then those patterns are compared to the one discovered by our algorithms (F-)SPoID.

Results here detailed were obtained from several synthetic datasets containing around 2000 sequences of 20 transactions in average, all with the same kind of item and missing value distributions. Each transaction contains around 10 items chosen among 100 for the binary data and 20 fuzzy items chosen among 300.

5.1 Result presentation

Our analysis is based on the several counting:

- the total number of sequential patterns discovered by (F-)SPoID,
- the number of sequential patterns discovered by (F-)SPoID, that are discovered in the complete database *(true positive patterns)*,
- the number of wrong sequential patterns discovered by (F-)SPoID (that groups together the patterns that are not discovered in the complete database *(false positive patterns)* and the one not found by (F-)SPoID but should be *(false negative patterns)*).

Table 4 sums up these notations.

β	# sequential patterns discovered by (F-)SPoID, also contained in the complete dataset *(true positive patterns)*
δ	# different sequential patterns *(false negative + false positive patterns)*
θ	# sequential patterns discovered by (F-)SPoID in the incomplete database *(true positive + false positive patterns)*
τ	# sequential patterns discovered in the complete dataset

Table 4: Notations for the different kinds of patterns.

5.2 SPoID Experiments

First, Figure 2(a) shows the evolution of the ratio β/θ, with respect to the minimum representativity threshold. It can be noted that this rate increases according to *minRep*. It means that among sequential patterns discovered by SPoID, the proportion of sequential patterns obtained on the complete dataset increases with the *minRep* threshold.

This observation can be completed by analysing Figure 2(b), which represents evolution of the ratio β/τ (number of discovered sequential patterns with respect

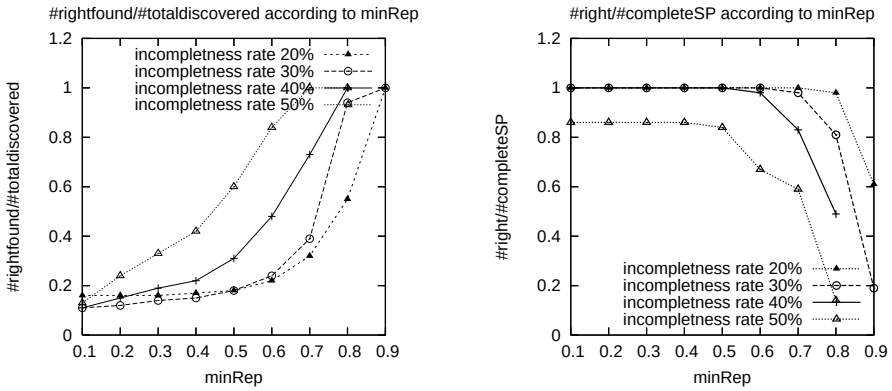


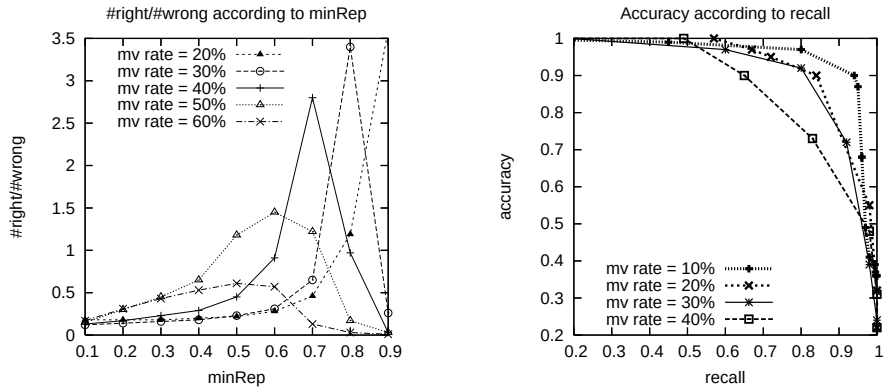
Figure 2: (a): β/θ rate (accuracy) according to *minRep*; (b): β/τ (rate according to *minRep*.

to sequential patterns that should be discovered) according to the minimum representativity threshold. This ratio decreases while *minRep* increases. It means that the minimum representativity threshold should be low enough to allow the discovery of all the sequential patterns obtained in the complete database.

Then we show that there exists an optimal value of the representativity threshold for which the ratio β/θ and β/τ are the closest to 1. This value is the threshold at which the right patterns discovered in the incomplete database are the most numerous compared to the number of wrong patterns. Figure 3(a) focuses on this optimal value *minRep*. This graph describes the evolution of the ratio β/δ according to the minimum representativity. It can be noted that there is not an absolute value for the minimum representativity, that would be common to every database independently from the incompleteness rate and only depending on an error margin. From these results, the minimum representativity threshold only depends on the incompleteness rate of the database.

Whatever the proportion of missing values in the incomplete database, the overall behavior of the ratio β/δ is quite the same: it increases until it reaches a maximum before decreasing. This maximal point corresponds to the average optimal representativity, for which the number of right patterns discoverd by SPoID is the highest and the number of wrong patterns is the lowest.
All these observations can be synthetized and confirmed by the analysis of figure 3(b), giving accuracy of SPoID according to recall for several incompleteness rate. Table 5 gives the value of optimal representativity empirically found, for each incompleteness rate in datasets.

We ran several experiments on different datasets with different size and item distributions. However, the empirical optimal representativity values do not cor-



(a) (b)

Figure 3: (a): β/δ rate according to *minRep*; (b): accuracy according to recall.

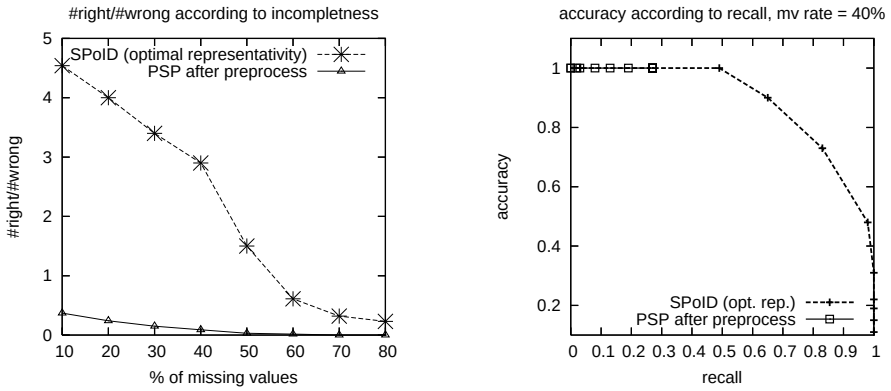
respond to the ones obtained by computing the statistical size of representative samples (see Section 4.2). This may come from the temporal aspect of the data. Therefore we currently work on using other statistical sampling methods, more adapted to time-related data.

% of missing values	10%	20%	30%	40%	50%	60%	70%
optimal *minRep*	0.97	0.9	0.81	0.74	0.6	0.48	0.39

Table 5: Average optimal representativity according to the missing value proportion in the database.

Figure 3(a) also shows the SPoID algorithm performances depending on the missing value proportion in the database. A difference can be noted between the overall evolution of the ratio β/δ for databases containing less than 40% of missing values and the one containing 50% of incomplete records or more.
Thus Figure 4(a) gives the comparison of SPoID success rate with results obtained on a preprocessed incomplete database. This figure shows that some right patterns, discovered by SPoID are not found using a deletion preprocessing step. This is also shown by computing accuracy and recall for each method. On Figure 4(b) we can thus note that each pattern discovered by the after-deletion method is a pattern found in the complete database (accuracy equals to 1). However the recall is very low, which means that only a small number of patterns that should be found are actually discovered. On the contrary, using the SpoID approach, even if the accuracy is not equal to 1, there exist a value of *minRep* that optimizes both recall and accuracy.

Figure 4(a) also shows that the ratio of right patterns strongly decreases between 40 and 50% of missing values: the number of wrong patterns becomes pro-



(a) (b)

Figure 4: (a): β/δ rate according to the missing value proportion; (b): accuracy according to recall (mv rate=40%).

portionnaly slightly higher compared to the number of right patterns. This ratio even becomes less than 1 when the missing value percentage exceeds 50%. SPoID can then discover sequential patterns within incomplete database if at least half of the records in the dataset are complete, while the former methods requiring a preprocessing step do not find all the frequent patterns since 10% of missing values.

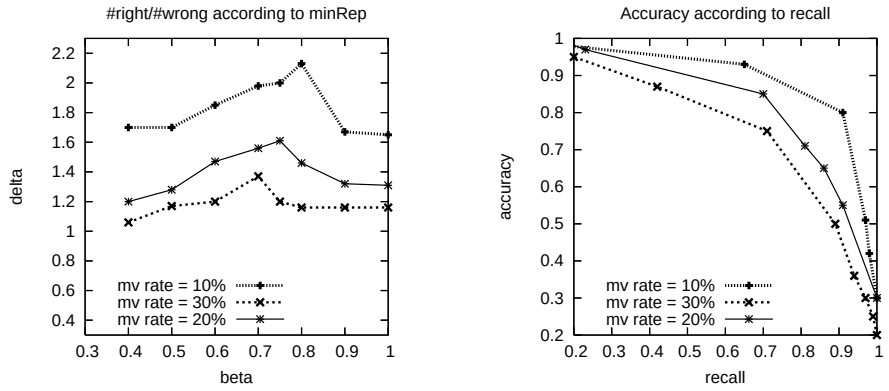
Last, the analysis of runtime performances shows that at constant incompleteness rate, runtime of SPoID is slightly constant while the *minRep* threshold decreases.The qualitative analysis of corresponding candidate and frequent sequences has shown that the number of candidate sequences increases but, as the minimum representativity is lower, the time spent for scanning the database decreases. We also noted that runtime increases with the incompleteness rate.

5.3 F-SPoID Experiments

The same kind of observations were done on fuzzy databases, using F-SPoID. We ran experiments on several datasets, varying the parameters $minFreq$, $minRep$ and the incompleteness rate of the databases.

Figure 5(a) shows the evolution of the ratio β/δ according to $minRep$, depending on the rate of incompleteness. There exists a value of $minRep$ for which the number of true positive sequential patterns is the highest compared to the number of different patterns (false positive and false negative) . This value of $minRep$ is the optimal value for which the accuracy and recall of F-SPoID are maximal. Table 6 gives the average optimal values for our synthetic datasets depending on the incompleteness rate.

The existence of this optimal value is confirmed by the analysis of the parametric curve of accuracy according to recall considering the parameter $minRep$, depending



(a) (b)

Figure 5: (a): β/δ rate according to *minRep*; (b): accuracy according to recall.

on the proportion of missing values within the data. On figure 5(b) it can be noted that for all datasets the evolution of accuracy with respect to recall is similar. It is also close to the one observed on crisp data, even if the results on fuzzy data are overall not as good as on the crisp datasets.
We also observed this kind of behaviors on quantitative datasets generated with different quantity or item distributions and also modifying the way quantity fuzzy sets are computed.

% of missing values	10%	20%	30%	40%	50%
optimal *minRep*	0.81	0.76	0.7	0.64	0.44

Table 6: Average optimal representativity according to the missing value proportion in the fuzzy database.

5.4 Complementary Work

From these experiments on synthetic datasets one can note that quality of results is highly linked to the different parameters: data distribution, dataset size, missing value distribution and minimum representativity threshold. With these experiments we found out that the empirical optimal representativity value is not the same as the one computed from statistical formulae. This observation can be done on both binary or quantitative datasets. So our hypothesis is that the sampling methods used for mining association rules on database samples [24] are not the most adapted to time-related datasets.

Therefore these works and experiments should be extended to determine the most adapted sampling method for mining sequential data.

Another additional work is currently in progress on applying the techniques SPoID and F-SPoID on real-world data. This validation does not necessarily require an expert knowledge as it is done using a complete database in which missing values are randomly spread by deleting some information. Then the comparison between approaches is done on the same principle as the one used on synthetic data. However experts could help in subjective analysis of the true negative and false positive patterns.

6 Conclusion

Temporal databases available from application areas such as biological data or industrial process data most of the time contain numerical attributes and a lot of incomplete data. The most adapted data mining technique to analyze such time-stamped datasets are sequential pattern discovery algorithms. However, although some works were done to handle these numerical features by specifying fuzzy sequential pattern algorithms, only few proposals focus on extracting frequent sequences – either crisp or fuzzy – from incomplete data.

Therefore, in this paper, we first described the definitions for sequential pattern mining in order to handle random incompleteness in data sequences, we detailed in [8]. Then, we extended these principles to fuzzy sequential patterns for mining incomplete quantitative sequence databases. These new definitions enable the user to manage missing values directly during the mining task, then avoiding a heavy preprocessing step. Our methods and algorithms SPoID and F-SPoID have been implemented and tested on synthetic datasets. We have thus shown the robustness of our approaches until an incompleteness rate around 40%, while the existing approaches give erroneous results since 10% of missing values.

After having proposed two methods for handling random missing values, one for binary databases, the second for quantitative databases, we now work on extending these approaches to handle other types of missing values such as related to one specific attribute, for instance. Next step will then consist in detecting the different kinds of incomplete information including the attributes that should not be considered as incomplete even if unfilled. Last, to help the use of our methods on real-life databases, we work on the specification of analysis tools to express the interest and quality of discovered sequences with respect to incompleteness of the data sequences that include these patterns.

Acknowledgment

The authors would like to thank Prof. Dr. Eyke Hüllermeier, from Philipps-Universität in Marburg, Germany, for the fruitfull discussions we had.

References

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216. Peter Buneman and Sushil Jajodia, 1993.

[2] R. Agrawal and R. Srikant. Mining sequential patterns. In *Eleventh International Conference on Data Engineering*, pages 3–14, Taipei, Taiwan, 1995. IEEE Computer Society Press.

[3] D. Arotaritei. Mining fuzzy association rules in incomplete databases. In *IEEE International Conference on Fuzzy Systems (FuzzIEEE'02)*, 2002.

[4] T. Calders, B. Goethals, and M. Mampaey. Mining itemsets in the presence of missing values. In *ACM Symposium on Applied Computing (SAC'07)*, 2007.

[5] R.-S. Chen, G.-H. Tzeng, C.-C. Chen, and Y.-C. Hu. Discovery of Fuzzy Sequential Patterns for Fuzzy Partitions in Quantitative Attributes. In *ACS/IEEE Int. Conf. on Computer Systems and Applications*, pages 144–150, 2001.

[6] C. Fiot, A. Laurent, and M. Teisseire. Extended time constraints for sequence mining. In *14th IEEE International Symposium on Temporal Representation and Reasoning (TIME'07)*, 2007.

[7] C. Fiot, A. Laurent, and M. Teisseire. From crispness to fuzziness: Three algorithms for soft sequential pattern mining. *IEEE Transactions on Fuzzy Systems*, 2007.

[8] C. Fiot, A. Laurent, and M. Teisseire. Spoid: Do not throw meaningful incomplete séquences away. In *European Society For Fuzzy Logic and Technologies - New Dimensions in Fuzzy Logic and Related Technologies*, 2007.

[9] 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*, pages 263–268, 1998.

[10] A. Gyenesei and J. Teuhola. Multidimensional fuzzy partitioning of attribute ranges for mining quantitative data: Research articles. *Int. J. Intell. Syst.*, 19(11):1111–1126, 2004.

[11] T.P. Hong, K.Y. Lin, and S.L. Wang. Mining Fuzzy Sequential Patterns from Multiple-Items Transactions. In *Joint 9th IFSA World Congress and 20th NAFIPS Int. Conf.*, pages 1317–1321, 2001.

[12] W. Z. Liu, A. P. White, S. G. Thompson, and M. A. Bramer. Techniques for dealing with missing values in classification. *Lecture Notes in Computer Science*, 1280, 1997.

[13] W. Z. Liu, A. P. White, S. G. Thompson, and M. A. Bramer. Techniques for Dealing with Missing Values in Classification. In *the 2nd International Symposium on Advances in Intelligent Data Analysis, Reasoning about Data*, volume 1280 of *Lecture Notes in Computer Science*, 1997.

[14] F. Masseglia, P. Poncelet, and M. Teisseire. Incremental Mining of Sequential Patterns in Large Databases. *Data and Knowledge Engineering*, 46(1):97–121, 2003.

[15] F. Masseglia, P. Poncelet, and M. Teisseire. Pre-Processing Time Constraints for Efficiently Mining Generalized Sequential Patterns. In *11th Int. Symp. on Temporal Representation and Reasoning (TIME '04)*, pages 87–95, 2004.

[16] J. Nayak and D. Cook. Approximate association rule mining. In *Florida Artificial Intelligence Research Symposium*, 2001.

[17] V. Ng and J. Lee. Quantitative association rules over incomplete data. In *IEEE International Conference*, pages 2821–2826, 1998.

[18] J. R. Quinlan. Unknown Attribute Values in Induction. In *Proc. 6th Int. Machine Learning Workshop Cornell*. Morgan Kaufmann, 1989.

[19] A. Ragel and B. Cremilleux. Treatment of missing values for association rules. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 258–270, 1998.

[20] R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *5th International Conference on Extending Database Technology (EDBT '96)*, pages 3–17, London, UK, 1996. Springer-Verlag.

[21] H. Timm, C. Doring, and R. Kruse. Different Approaches to Fuzzy Clustering of Incomplete Datasets. *International Journal of Approximate Reasoning*, 35:239–249, 2004.

[22] H. Toivonen. Sampling large databases for association rules. In *VLDB '96: Proceedings of the 22th International Conference on Very Large Data Bases*, pages 134–145, 1996.

[23] S. M. Weiss and N. Indurkhya. Decision-Rule Solutions for Data Mining with Missing Values. In *the 7th International Joint Ibero-American Conference on AI: Advances in Artificial Intelligence*, volume 1952 of *Lecture Notes In Computer Science*, pages 1–10, 2000.

[24] M. J. Zaki, S. Parthasarathy, W. Li, and M. Ogihara. Evaluation of sampling for data mining of association rules. Technical report, Rochester, NY, USA, 1996.