M^2SP: Mining Sequential Patterns Among Several Dimensions

M. Plantevit[1], Y.W. Choong[2,3], A. Laurent[1], D. Laurent[2], and M. Teisseire[1]

[1] LIRMM, Université Montpellier 2, CNRS, 161 rue Ada, 34392 Montpellier, France
[2] LICP, Université de Cergy Pontoise, 2 av. Chauvin, 95302 Cergy-Pontoise, France
[3] HELP University College, BZ-2 Pusat Bandar Damansara, 50490 Kuala Lumpur, Malaysia

Abstract. Mining sequential patterns aims at discovering correlations between events through time. However, even if many works have dealt with sequential pattern mining, none of them considers frequent sequential patterns involving several dimensions in the general case. In this paper, we propose a novel approach, called M^2SP, to mine multidimensional sequential patterns. The main originality of our proposition is that we obtain not only intra-pattern sequences but also inter-pattern sequences. Moreover, we consider generalized multidimensional sequential patterns, called jokerized patterns, in which some of the dimension values may not be instanciated. Experiments on synthetic data are reported and show the scalability of our approach.

Keywords: Data Mining, Sequential Patterns, Multidimensional Rules.

1 Introduction

Mining sequential patterns aims at discovering correlations between events through time. For instance, rules that can be built are *A customer who bought a TV and a DVD player at the same time later bought a recorder*. Work dealing with this issue in the literature have proposed scalable methods and algorithms to mine such rules [9]. As for association rules, the efficient discovery is based on the *support* which indicates to which extend data from the database contains the patterns.

However, these methods only consider one dimension to appear in the patterns, which is usually called the *product* dimension. This dimension may also represent web pages for web usage mining, but there is normally a single dimension. Although some works from various studies claim to combine several dimensions, we argue here that they do not provide a complete framework for multidimensional sequential pattern mining [4,8,11]. The way we consider multidimensionality is indeed generalized in the sense that patterns must contain several dimensions combined over time. For instance we aim at building rules like *A customer who bought a surfboard and a bag in NY later bought a wetsuit in SF*. This rule not only combines two dimensions (*City* and *Product*) but it also combines them over time (NY appears before SF, surfboard appears before wetsuit). As far as we know, no method has been proposed to mine such rules.

In this paper, we present existing methods and their limits. Then, we define the basic concepts associated to our proposition, called M^2SP, and the algorithms to build such rules. Experiments performed on synthetic data are reported and assess our proposition.

A. Jorge et al. (Eds.): PKDD 2005, LNAI 3721, pp. 205–216, 2005.

In our approach, sequential patterns are mined from a relational table, that can be seen as a fact table in a multidimensional database. This is why, contrary to the standard terminology of the relational model, the attributes over which a relational table is defined are called *dimensions*.

In order to mine such frequent sequences, we extend our approach so as to take into account partially instanciated tuples in sequences. More precisely, our algorithms are designed in order to mine frequent jokerized multidimensional sequences containing as few $*$ as possible, i.e., replacing an occurrence of $*$ with any value from the corresponding domain cannot give a frequent sequence.

The paper is organized as follows: Section 2 introduces a motivating example illustrating the goal of our work, and Section 3 reviews previous works on sequential patterns mining. Section 4 introduces our contribution, and in Section 5, we extend multidimensional patterns to *jokerized* patterns. Section 6 presents the algorithms, and experiments performed on synthetic data are reported in Section 7. Section 8 concludes the paper.

2 Motivating Example

In this section, we first briefly recall the basic ingredients of the relational model of databases used in this paper (we refer to [10] for details on this model), and we present an example to illustrate our approach. This example will be used throughout the paper as a running example.

Let $U = \{D_1, \ldots D_n\}$ be a set of attributes, which we call *dimensions* in our approach. Each dimension D_i is associated with a (possibly infinite) domain of values, denoted by $dom(D_i)$. A relational table T over universe U is a finite set of tuples $t = (d_1, \ldots, d_n)$ such that, for every $i = 1, \ldots, n$, $d_i \in dom(D_i)$. Moreover, given a table T over U, for every $i = 1, \ldots, n$, we denote by $Dom_T(D_i)$ (or simply $Dom(D_i)$ if T is clear from the context) the *active domain* of D_i in T, i.e., the set of all values of $dom(D_i)$ that occur in T.

Since we are interested in sequential patterns, we assume that U contains at least one dimension whose domain is totally ordered, corresponding to the *time dimension*.

In our running example, we consider a relational table T in which transactions issued by customers are stored. More precisely, we consider a universe U containing six dimensions (or attributes) denoted by D, CG, A, P and Q, where: D is the date of transactions (considering three dates, denoted by $1, 2$ and 3), CG is the category of customers (considering two categories, denoted by $Educ$ and Ret, standing for educational and retired customers, respectively), A is the age of customers (considering three discretized values, denoted by Y (young), M (middle) and O (old)), C is the city where transactions have been issued (considering three cities, denoted by NY (New York), LA (Los Angeles) and SF (San Francisco)), P is the product of the transactions (considering four products, denoted by c, m, p and r), and Q stands for the quantity of products in the transactions (considering nine such quantities).

Fig. 1 shows the table T in which, for instance, the first tuple means that, at date 1, educational young customers bought 50 products c in New York. Let us now assume that we want to extract all multidimensional sequences that deal with the age of customers, the products they bought and the corresponding quantities, and that are frequent with respect to the groups of customers and the cities where transactions have been issued. To this end, we consider three sets of dimensions as follows: (i) the dimension D, representing the date, (ii) the three dimensions A, P and Q that we call *analysis dimensions*, (iii) the two dimensions CG and C, that we call *reference dimensions*.

Tuples over analysis dimensions are those that appear in the items that constitute the sequential patterns to be mined. The table is partitioned into blocks according to tuple values over reference dimensions and the support of a given multidimensional sequence is the ratio of the number of blocks supporting the sequence over the total number of blocks. Fig. 2 displays the corresponding blocks in our example.

In this framework, $\langle\{(Y,c,50),(M,p,2)\},\{(M,r,10)\}\rangle$ is a multidimensional sequence having support $\frac{1}{3}$, since the partition according to the reference dimensions contains 3 blocks, among which one supports the sequence. This is so because $(Y,c,50)$ and $(M,p,2)$ both appear at the same date (namely date 1), and $(M,r,10)$ appears later on (namely at date 2) in the first block shown in Figure 4.

It is important to note that, in our approach, more general patterns, called *jokerized sequences,* can be mined. The reason for this generalization is that considering partially instanciated tuples in sequences implies that more frequent sequences are mined. To see this, considering a support threshold of $\frac{2}{3}$, no sequence of the form $\langle\{(Y,c,\mu)\},\{(M,r,\mu')\}\rangle$ is frequent. On the other hand, in the first two blocks of Fig. 2, Y associated with c and M associated with r appear one after the other, according to the date of transactions. Thus, we consider that the jokerized sequence, denoted by $\langle\{(Y,c,*)\},\{(M,r,*)\}\rangle$, is frequent since its support is equal to $\frac{2}{3}$.

D (Date)	CG (Customer-Group)	C (City)	A (Age)	P (Product)	Q (Quantity)
1	Educ	NY	Y	c	50
1	Educ	NY	M	p	2
1	Educ	LA	Y	c	30
1	Ret.	SF	O	c	20
1	Ret.	SF	O	m	2
2	Educ	NY	M	p	3
2	Educ	NY	M	r	10
2	Educ	LA	Y	c	20
3	Educ	LA	M	r	15

Fig. 1. Table *T*

3 Related Work

In this section, we argue that our approach generalizes previous works on sequential patterns. In particular, the work described in [8] is said to be *intra*-pattern since sequences are mined within the framework of a single description (the so-called *pattern*). In this paper, we propose to generalize this work to *inter*-pattern multidimensional sequences.

3.1 Sequential Patterns

An early example of research in the discovering of patterns from sequences of events can be found in [5]. In this work, the idea is the discovery of rules underlying the generation of a given sequence in order to predict a plausible sequence continuation. This idea is then extended to the discovery of interesting patterns (or *rules*) embedded in a database of sequences of sets of events (items). A more formal approach in solving the problem of mining sequential patterns is the AprioriAll algorithm as presented in [6]. Given a database of sequences, where each sequence is a list of transactions ordered by transaction time, and each transaction is a set of items, the goal is to discover all sequential patterns with a user-specified minimum support, where the support of a pattern is the number of data-sequences that contain the pattern.

In [1], the authors introduce the problem of mining sequential patterns over large databases of customer transactions where each transaction consists of customer-id, transaction time, and the items bought in the transaction. Formally, given a set of sequences, where each sequence consists of a list of elements and each element consists of a set of items, and given a user-specified min support threshold, sequential pattern mining is to find all of the frequent subsequences, i.e., the subsequences whose occurrence frequency in the set of sequences is no less than min support. Sequential pattern mining discovers frequent patterns ordered by time. An example of this type of pattern is *A customer who bought a new television 3 months ago, is likely to buy a DVD player now*. Subsequently, many studies have introduced various methods in mining sequential patterns (mainly in time-related data) but almost all proposed methods are Apriori-like, i.e., based on the Apriori property which states the fact that any super-pattern of a nonfrequent pattern cannot be frequent. An example using this approach is the GSP algorithm [9].

3.2 Multidimensional Sequential Patterns

As far as we know, three propositions have been studied in order to deal with several dimensions when building sequential patterns. Next, we briefly recall these propositions.

Pinto et al. [8]. This work is the first one dealing with several dimensions in the framework of sequential patterns. For instance, purchases are not only described by considering the customer ID and the products, but also by considering the age, the type of the customer (Cust-Grp) and the city where (s)he lives, as shown in Fig. 1.

Multidimensional sequential patterns are defined over the schema $A_1,...,A_m,S$ where $A_1,\ldots,A_m$ are the dimensions describing the data and S is the sequence of items purchased by the customers ordered over time. A multidimensional sequential pattern is defined as $(id_1,(a_1,...,a_m),s)$ where $a_i \in A_i \cup \{*\}$. $id_1,(a_1,...,a_m)$ is said to be a multidimensional pattern. For instance, the authors consider the sequence $((*,NY,*),\langle bf \rangle)$ meaning that customers from NY have all bought a product b and then a product f. Sequential patterns are mined from such multidimensional databases either (i) by mining all frequent sequential patterns over the product dimension and then regrouping them into multidimensional patterns, (ii) or by mining all frequent multidimensional patterns and then mining frequent product sequences over these patterns. Note that the sequences found by this approach do not contain several dimensions since the dimension time only concerns products. Dimension product is the only dimension that can be combined over time, meaning that it is not possible to have a rule indicating that when b is bought in *Boston* then c is bought in *NY*. Therefore, our approach can seen as a generalization of the work in [8].

Yu et Chen. [11]. In this work, the authors consider sequential pattern mining in the framework of Web Usage Mining. Even if three dimensions (pages, sessions, days) are considered, these dimensions are very particular since they belong to a single hierarchized dimension. Thus, the sequences mined in this work describe correlations between objects over time by considering only one dimension, which corresponds to the web pages.

de Amo et al. [4]. This approach is based on first order temporal logic. This proposition is close to our approach, but more restricted since (i) groups used to compute the support are predefined whereas we consider the fact that the user should be able to define them (see reference dimensions below), and (ii) several attributes cannot appear in the sequences. The authors claim that they aim at considering several dimensions but they have only shown one dimension for the sake of simplicity. However, the paper does not provide hints for a complete solution with *real* multidimensional patterns, as we do in our approach.

4 M^2SP: *M*ining *M*ultidimensional *S*equential *P*atterns

4.1 Dimension Partition

For each table defined on the set of dimensions D, we consider a partition of D into four sets: D_t for the temporal dimension, D_{A} for the *analysis* dimensions, D_{R} for the *reference* dimensions, and D_{F} for the *ignored* dimensions.

Each tuple $c = (d_1, \ldots, d_n)$ can thus be written as $c = (f, r, a, t)$ where f, r, a and t are the restrictions of c on D_{F}, D_{R}, D_{A} and D_t, respectively.

Given a table T, the set of all tuples in T having the same restriction r over D_{R} is said to be a *block*. Each such block B is denoted by the tuple r that defines it, and we denote by $\mathrm{B}_{T,D_{\mathrm{R}}}$ the set of all blocks that can be built up from table T.

In our running example, we consider $\mathrm{F} = \emptyset$, $D_{\mathrm{R}} = \{CG, C\}$, $D_{\mathrm{A}} = \{A, P, Q\}$ and $D_t = \{D\}$. Fig. 2 shows the three blocks built up from table T.

D	CG	C	A	P	Q
1	Educ	NY	Y	c	50
1	Educ	NY	M	p	2
2	Educ	NY	M	p	3
2	Educ	NY	M	r	10

a. Block $(Educ, NY)$

D	CG	C	A	P	Q
1	Educ	LA	Y	c	30
2	Educ	LA	Y	c	20
3	Educ	LA	M	r	15

b. Block $(Educ, LA)$

D	CG	C	A	P	Q
1	Ret.	SF	O	c	20
1	Ret.	SF	O	m	2

c. Block $(Ret., SF)$

Fig. 2. Blocks defined on T over dimensions CG and C

When mining multidimensional sequential patterns, the set D_{R} identifies the blocks of the database to be considered when computing supports. The support of a sequence is the proportion of blocks embedding it. Note that, in the case of usual sequential patterns and of sequential patterns as in [8] and [4], this set is reduced to one dimension (*cid* in [8] or *IdG* in [4]).

The set D_{A} describes the analysis dimensions, meaning that values over these dimensions appear in the multidimensional sequential patterns. Note that usual sequential patterns only consider one analysis dimension corresponding to the products purchased or the web pages visited. The set F describes the ignored dimensions, i.e. those that are used neither to define the date, nor the blocks, nor the patterns to be mined.

4.2 Multidimensional Item, Itemset and Sequential Pattern

Definition 1 (Multidimensional Item). *Let $D_{\mathrm{A}} = \{D_{i_1}, \ldots, D_{i_m}\}$ be a subset of D. A multidimensional item on D_{A} is a tuple $e = (d_{i_1}, \ldots, d_{i_m})$ such that, for every k in $[1,m]$, d_{i_k} is in $Dom(D_{i_k})$.*

Definition 2 (Multidimensional Itemset). *A multidimensional itemset on D_{A} is a non empty set of items $i = \{e_1, \ldots, e_p\}$ where for every j in $[1, p]$, e_j is a multidimensional item on D_{A} and for all j,k in $[1, p]$, $e_j \neq e_k$.*

Definition 3 (Multidimensional Sequence). *A multidimensional sequence on D_{A} is an ordered non empty list of itemsets $\varsigma = \langle i_1, \ldots, i_l \rangle$ where for every j in $[1,l]$, i_j is a multidimensional itemset on D_{A}.*

In our running example, $(Y,c,50)$, $(M, p,2)$, $(M,r,10)$ are three multidimensional items on $D_{\mathrm{A}} = \{A,P,Q\}$. Thus, $\langle\{(Y,c,50),(M, p,2)\},\{(M,r,10)\}\rangle$ is a multidimensional sequence on D_{A}.

Definition 4 (Inclusion of sequence). *A multidimensional sequence $\varsigma = \langle a_1, \ldots, a_l \rangle$ is said to be a subsequence of a sequence $\varsigma' = \langle b_1, \ldots, b_{l'} \rangle$ if there exist $1 \leq j_1 \leq j_2 \leq \ldots \leq j_l \leq l'$ such that $a_1 \subseteq b_{j_1}, a_2 \subseteq b_{j_2}, \ldots, a_l \subseteq b_{j_l}$.*

With $\varsigma = \langle\{(Y,c,50)\}, \{(M,r,10)\}\rangle$ and $\varsigma' = \langle\{(Y,c,50),(M, p,2)\},\{(M,r,10)\}\rangle$, ς is a subsequence of ς'.

4.3 Support

Computing the support of a sequence amounts to count the number of blocks that *support* the sequence. Intuitively, a block supports a sequence ς if (i) for each itemset *i* in ς there exists a date in $Dom(D_t)$ such that all items in *i* appear at this date, and (ii) all itemsets in ς are successively retrieved at different and increasing dates.

Definition 5. *A table T supports a sequence $\langle i_1, \ldots, i_l \rangle$ if for every $j = 1, \ldots, l$, there exists d_j in $Dom(D_t)$ such that for every item e in i_j, there exists $t = (f,r,e,d_j)$ in T with $d_1 < d_2 < \ldots < d_l$.*

In our running example, the block $(Educ, NY)$ from Fig. 2.a supports $\varsigma = \langle\{(Y,c,50), (M,p,2)\}, \{(M,r,10)\}\rangle$ since $\{(Y,c,50), (M,p,2)\}$ appears at $date = 1$ and $\{(M,r,10)\}$ appears at $date = 2$.

The support of a sequence in a table T is the proportion of blocks of T that support it.

Definition 6 (Sequence Support). *Let D_{R} be the reference dimensions and T a table partitioned into the set of blocks $\mathrm{B}_{T,D_{\mathrm{R}}}$. The support of a sequence ς is defined by:*

$$support(\varsigma) = \frac{|\{B \in \mathrm{B}_{T,D_{\mathrm{R}}} \mid B\, supports\ \varsigma\}|}{|\mathrm{B}_{T,D_{\mathrm{R}}}|}$$

Definition 7 (Frequent Sequence). *Let $minsup \in [0,1]$ be the minimum user-defined support value. A sequence ς is said to be frequent if $support(\varsigma) \geq minsup$.*
An item e is said to be frequent if so is the sequence $\langle\{e\}\rangle$.

In our running example, let us consider $D_{\mathrm{R}} = \{CG, C\}$, $D_{\mathrm{A}} = \{A, P, Q\}$, $minsup = \frac{1}{5}$, $\varsigma = \langle\{(Y,c,50),(M,p,2)\},\{(M,r,10)\}\rangle$. The three blocks of the partition of T from Fig. 2 must be scanned to compute $support(\varsigma)$.
1. Block (Educ, NY) (Fig. 2.a). In this block, we have $(Y,c,50)$ and $(M,p,2)$ at date 1, and $(M,r,10)$ at date 2. Thus this block supports ς.
2. Block (Educ, LA) (Fig. 2.b). This block does not support ς since it does not contain $(M,p,2)$.
3. Block (Ret., SF) (Fig. 2.c). This block does not support ς since it contains only one date.
Thus, we have $support(\varsigma) = \frac{1}{3} \geq minsup$.

5 Jokerized Sequential Patterns

Considering the definitions above, an item can only be retrieved if there exists a frequent tuple of values from domains of D_{A} containing it. For instance, it can happen that neither (Y,r) nor (M,r) nor (O,r) is frequent whereas the value r is frequent. In this case, we consider $(*,r)$ which is said to be *jokerized*.

Definition 8 (Jokerized Item). *Let $e = (d_1, \ldots, d_m)$ a multidimensional item. We denote by $e_{[d_i/\delta]}$ the replacement in e of d_i by δ. e is said to be a jokerized multidimensional item if: (i) $\forall i \in [1,m], d_i \in Dom(D_i) \cup \{*\}$, and (ii) $\exists i \in [1,m]$ such that $d_i \neq *$, and (iii) $\forall d_i = *$, $\nexists \delta \in Dom(D_i)$ such that $e_{[d_i/\delta]}$ is frequent.*

A *jokerized* item contains at least one specified analysis dimension. It contains a $*$ only if no specific value from the domain can be set. A *jokerized* sequence is a sequence containing at least one *jokerized* item. A block is said to *support* a sequence if a set of tuples containing the itemsets satisfying the temporal constraints can be found.

Definition 9 (Support of a Jokerized Sequence). *A table T supports a jokerized sequence $\varsigma = \langle i_1, \ldots, i_l\rangle$ if: $\forall j \in [1,l]$, $\exists \delta_j \in Dom(D_t)$, $\forall e = (d_{i_1}, \ldots, d_{i_m}) \in i_j$, $\exists t = (f, r, (x_{i_1}, \ldots, x_{i_m}), \delta_j) \in T$ with $d_{i_k} = x_{i_k}$ or $d_{i_k} = *$ and $\delta_1 < \delta_2 < \ldots < \delta_l$.*
The support of ς is defined by: $support(\varsigma) = \frac{|\{B \in \mathrm{B}_{T,D_{\mathrm{R}}}\ s.t.\ B\, supports\ \varsigma\}|}{|\mathrm{B}_{T,D_{\mathrm{R}}}|}$

6 Algorithms

6.1 Mining Frequent Items

The computation of all frequent sequences is based on the computation of all frequent multidimensional items. When considering no joker value, a single scan of the database is enough to compute them.

On the other hand, when considering jokerized items, a levelwise algorithm is used in order to build the frequent multidimensional items having as few joker values as possible. To this end, we consider a lattice which lower bound is the multidimensional item $(*,\ldots,*)$. This lattice is partially built from $(*,\ldots,*)$ up to the frequent items containing as few $*$ as possible. At level i, i values are specified, and items at this level are combined to build a set of candidates at level $i+1$. Two frequent items are combined to build a candidate if they are $\bowtie$-compatible.

Definition 10 ($\bowtie$-compatibility). *Let $e_1 = (d_1,\ldots,d_n)$ and $e_2 = (d'_1,\ldots,d'_n)$ be two distinct multidimensional items where d_i and $d'_i \in dom(D_i) \cup \{*\}$. e_1 and e_2 are said to be $\bowtie$-compatible if there exists $\Delta = \{D_{i_1},\ldots,D_{i_{n-2}}\} \subset \{D_1,\ldots,D_n\}$ such that for every $j \in [1,n-2]$, $d_{i_j} = d'_{i_j} \neq *$ with $d_{i_{n-1}} = *$ and $d'_{i_{n-1}} \neq *$ and $d_{i_n} \neq *$ and $d'_{i_n} = *$.*

Definition 11 (Join). *Let $e_1 = (d_1,\ldots,d_n)$ and $e_2 = (d'_1,\ldots,d'_n)$ be two $\bowtie$-compatible multidimensional items. We define $e_1 \bowtie e_2 = (v_1,\ldots,v_n)$ where $v_i = d_i$ if $d_i = d'_i$, $v_i = d_i$ if $d'_i = *$ and $v_i = d'_i$ if $d_i = *$.*
Let E and E' be two sets of multidimensional items of size n, we define

$$E \bowtie E' = \{e \bowtie e' \mid (e,e') \in E \times E' \wedge \text{ e and e' are } \bowtie\text{-compatible}\}$$

In our running example, $(NY,Y,*)$ and $(*,Y,r)$ are $\bowtie$-compatible. We have $(NY,Y,*) \bowtie (*,Y,r) = (NY,Y,r)$. On the contrary, $(NY,M,*)$ and $(NY,Y,*)$ are not $\bowtie$-compatible. Note that this method is close to the one used for iceberg cubes in [2,3].

Let F_1^i denote the set of 1-frequent items having i dimensions which are specified (different from $*$). F_1^1 is obtained by counting each value over each analysis dimension, i.e., $F_1^1 = \{f \in Cand_1^1,\ support(f) \geq minsup\}$. Candidate items of size i are obtained by joining the set of frequent items of size $i-1$ with itself: $Cand_1^i = F_1^{i-1} \bowtie F_1^{i-1}$.

```
Function supportcount
Data   : ς, T, D_R, counting //counting indicates if joker values are considered or not
Result : support of ς
Integer support ⟵ 0 ; Boolean seqSupported;
B_{T,D_R} ⟵ {blocks of T identified over D_R};
foreach B ∈ B_{T,D_R} do
    seqSupported ⟵ supportTable(ς, B, counting) ;
    if seqSupported then support ⟵ support + 1;
return (support / |B_{T,D_R}|)
```

Algorithm 1: Support of a sequence (supportcount)

Function supportTable
Data : $\varsigma, T, counting$
Result : Boolean
$ItemSetFound \longleftarrow false$; $seq \longleftarrow \varsigma$; $itset \longleftarrow seq.first()$; $it \longleftarrow itset.first()$
if $\varsigma = \emptyset$ **then** return (true) // End of Recursivity
while $t \longleftarrow T.next \neq \emptyset$ **do**
  if $supports(t, it, counting)$ **then**
    if $(NextItem \longleftarrow itset.second()) = \emptyset$ **then** $ItemSetFound \longleftarrow true$
    // Look for all the items from the itemset
    else
      // Anchoring on the item (date)
      $T' \longleftarrow \sigma_{date=t.date}(T)$
      while $t' \longleftarrow T'.next() \neq \emptyset \wedge ItemSetFound = false$ **do**
        if $supports(t', NextItem, counting)$ **then** $NextItem \longleftarrow itset.next()$
        if $NextItem = \emptyset$ **then** $ItemSetFound \longleftarrow true$
    if $ItemSetFound = true$ **then**
      // Anchoring on the current itemset succeeded; test the other itemsets in *seq*
      return $(supportTable(seq.tail(), \sigma_{date>t.date}(T), counting))$
    else
      // Anchoring failure: try anchoring with the next dates
      $itset \longleftarrow seq.first()$
      $T \longleftarrow \sigma_{date>t.date}(T)$ // Skip to next dates
return(false) // Not found

Algorithm 2: supportTable *(Checks if a sequence ς is supported by a table T)*

6.2 Mining Jokerized Multidimensional Sequences

The frequent items give all frequent sequences containing one itemset consisting of a single item. Then, the candidate sequences of size k ($k \geq 2$) are generated and validated against the table T. This computation is based on usual algorithms such as PSP [7] that are adapted for the treatment of joker values.

The computation of the support of a sequence ς according to the reference dimensions D_{R} is given by Algorithm 1. This algorithm checks whether each block of the partition supports the sequence by calling the function supportTable (Algorithm 2). *supportTable* attempts to find a tuple from the block that matches the first item of the first itemset of the sequence in order to *anchor* the sequence. This operation is repeated recursively until all itemsets from the sequence are found (return true) or until there is no way to go on further (return false). Several possible anchors may have to be tested.

7 Experiments

In this section, we report experiments performed on synthetic data. These experiments aim at showing the interest and scalability of our approach, especially in the jokerized approach. As many databases from the real world include quantitative information, we have distinguished a quantitative dimension. In order to highlight the particular role of this quantitative dimension, we consider four ways of computing frequent sequential patterns: (i) no joker (M^2SP), (ii) jokers on all dimensions but the quantitative one (M^2SP-$alpha$), (iii) jokers only on the quantitative dimension (M^2SP-mu), (iv) jokers on all dimensions (M^2SP-$alpha$-mu). Note that case (iv) corresponds to the jokerized approach presented in Section 5. Our experiments can thus be seen as being conducted in the context of a fact table of a multidimensional database, where the quantitative dimension is the *measure*. In Figures 5-12, minsup is the minimum support taken into account, nb_dim is the number of analysis dimensions being considered, DB_size is the number of tuples, and avg_card is the average number of values in the domains of the analysis dimensions.

Fig. 3 and 4 compare the behavior of the four approaches described above when the support changes. M^2SP-$alpha$ and M^2SP-$alpha$-mu have a similar behavior, the difference being due to the verification of quantities in the case of M^2SP-$alpha$. Note that these experiments are not led with the same minimum support values, since no frequent items are found for M^2SP and M^2SP-mu if the support is too high. Fig. 5 shows the scalability of our approach since runtime grows almost linearly when the database size increases (from $1,000$ tuples up to $26,000$ tuples).

Fig. 6 shows how runtime behaves when the average cardinality of the domains of analysis dimensions changes. When this average is very low, numerous frequent items are mined among few candidates. On the contrary, when this average is high, numerous candidates have to be considered from which few frequent items are mined. Between these two extrema, the runtime decreases. Fig. 7 and 8 show the behavior of our approach when the number of analysis dimensions changes. The number of frequent items increases as the number of analysis dimensions grows, leading to an increase of the number of frequent sequences. Fig. 9 and 10 show the differential between the number of frequent sequences mined by our approach compared to the number of frequent sequences mined by the approach described in [8], highlighting the interest of our proposition.

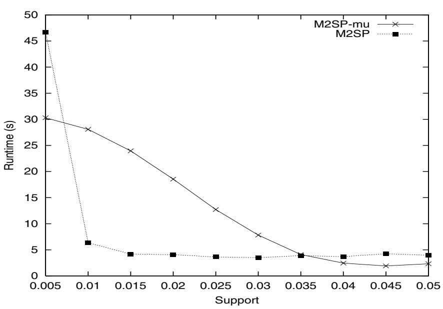

Fig. 3. Runtime over Support (DB_size=12000, nb_dim=5, avg_card=20)

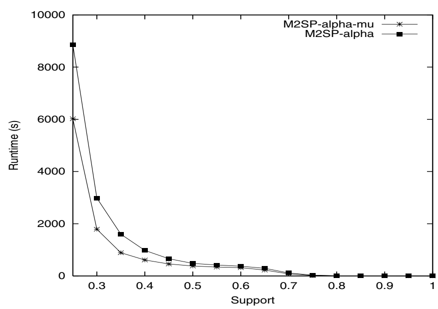

Fig. 4. Runtime over Support (DB_size=12000, nb_dim=5, avg_card=20)

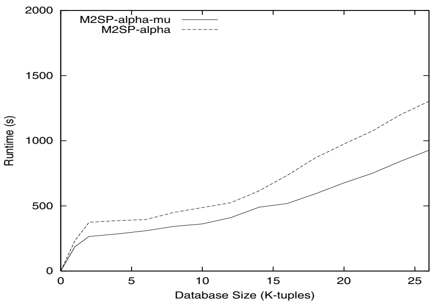

Fig. 5. Runtime over database size (minsup=0.5, nb_dim=15, avg_card = 20)

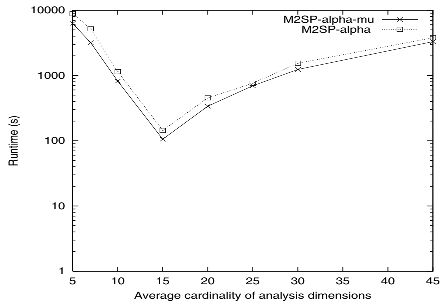

Fig. 6. Runtime over Average Cardinality of Analysis Dimensions (minsup=0.8, DB_size=12000, nb_dim=15)

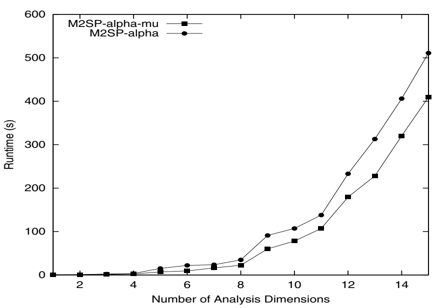

Fig. 7. Runtime over Number of Analysis Dimensions (minsup=0.5, DB_size=12000, nb_dim=15, avg_card=20)

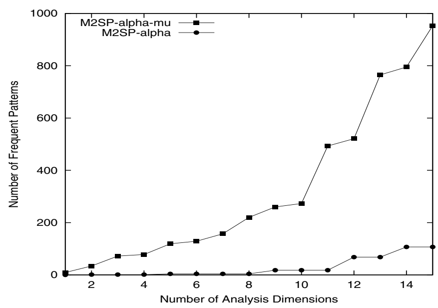

Fig. 8. Number of Frequent patterns over number of analysis dimensions (minsup=0.5, DB_size=12000, nb_dim=15, avg_card=20)

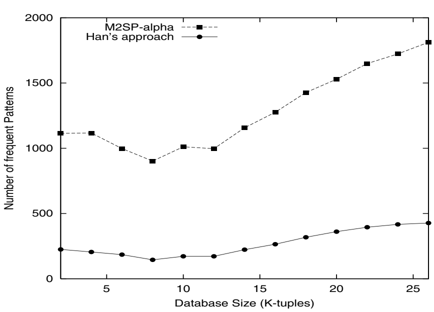

Fig. 9. Number of Frequent Sequences over Database Size (minsup=0.5, nb_dim=15, avg_card=20)

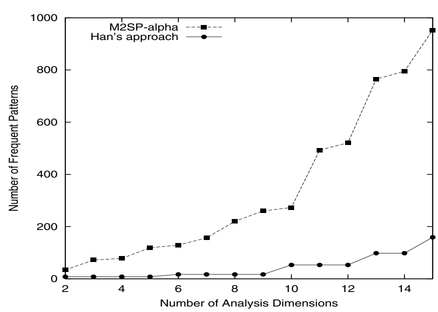

Fig. 10. Number of Frequent Sequences over Number of Analysis Dimensions (minsup=0.5, DB_size=12000, avg_card=20)

8 Conclusion

In this paper, we have proposed a novel definition for multidimensional sequential patterns. Contrary to the propositions [4,8,11], several analysis dimensions can be found in the sequence, which allows for the discovery of rules as *A customer who bought a surfboard together with a bag in NY later bought a wetsuit in LA*. We have also defined *jokerized sequential patterns* by introducing the joker value $*$ on analysis dimensions. Algorithms have been evaluated against synthetic data, showing the scalability of our approach.

This work can be extended following several directions. For example, we can take into account approximate values on quantitative dimensions. In this case, we allow the consideration of values that are not fully jokerized while remaining frequent. This proposition is important when considering data from the real world where the high number of quantitative values prevents each of them to be frequent. Rules to be built will then be like *The customer who bought a DVD player on the web is likely to buy* almost 3 *DVDs in a supermarket later*. Hierarchies can also be considered in order to mine multidimensional sequential patterns at different levels of granularity in the framework of multidimensional databases.

References

1. R. Agrawal and R. Srikant. Mining sequential patterns. In *Proc. 1995 Int. Conf. Data Engineering (ICDE'95)*, pages 3–14, 1995.
2. K. Beyer and R. Ramakrishnan. Bottom-up computation of sparse and iceberg cube. In *Proc. of ACM SIGMOD Int. Conf. on Management of Data*, pages 359–370, 1999.
3. A. Casali, R. Cicchetti, and L. Lakhal. Cube lattices: A framework for multidimensional data mining. In *Proc. of 3rd SIAM Int. Conf. on Data Mining*, 2003.
4. S. de Amo, D. A. Furtado, A. Giacometti, and D. Laurent. An apriori-based approach for first-order temporal pattern mining. In *Simpósio Brasileiro de Bancos de Dados*, 2004.
5. T.G. Dietterich and R.S. Michalski. Discovering patterns in sequences of events. *Artificial Intelligence*, 25(2):187–232, 1985.
6. H. Mannila, H. Toivonen, and A.I. Verkamo. Discovering frequent episodes in sequences. In *Proc. of Int. Conf. on Knowledge Discovery and Data Mining*, pages 210–215, 1995.
7. F. Masseglia, F. Cathala, and P. Poncelet. The PSP Approach for Mining Sequential Patterns. In *Proc. of PKDD*, volume 1510 of *LNCS*, pages 176–184, 1998.
8. H. Pinto, J. Han, J. Pei, K. Wang, Q. Chen, and U. Dayal. Multi-dimensional sequential pattern mining. In *ACM CIKM*, pages 81–88, 2001.
9. R. Srikant and R. Agrawal. Mining Sequential Patterns: Generalizations and Performance Improvements. In *Proc. of EDBT*, pages 3–17, 1996.
10. J.D. Ullman. *Principles of Database and Knowledge-Base Systems*, volume I. Computer Science Press, 1988.
11. C.-C. Yu and Y.-L. Chen. Mining sequential patterns from multidimensional sequence data. *IEEE Transactions on Knowledge and Data Engineering*, 17(1):136–140, 2005.