Mining Unexpected Multidimensional Rules

Marc Plantevit, Sabine Goutier, Françoise Guisnel, Anne Laurent, Maguelonne Teisseire

Mining Unexpected Multidimensional Rules

Marc Plantevit
LIRMM - Univ. Montpellier II
Montpellier, France
plantevi@lirmm.fr

Sabine Goutier
EDF R&D
Clamart, France
sabine.goutier@edf.fr

Françoise Guisnel
EDF R&D
Clamart, France
francoise.guisnel@edf.fr

Anne Laurent
LIRMM - Univ. Montpellier II
Montpellier, France
laurent@lirmm.fr

Maguelonne Teisseire
LIRMM - Univ. Montpellier II
Montpellier, France
teisseire@lirmm.fr

ABSTRACT

Discovering unexpected rules is essential, particularly for industrial applications with marketing stakes. In this context, many works have been done for association rules. However, non of them address sequences. In this paper, we thus propose to discover unexpected multidimensional sequential rules in data cubes. We define the concept of multidimensional sequential rule, and then unexpectedness. We formalize these concepts and define an algorithm for mining this kind of rules. Experiments on a real data cube are reported and highlight the interest of our approach.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database applications, data mining

General Terms

Algorithms, Design, Theory

Keywords

Unexpected Patterns, Sequential Patterns, Multidimensional Framework

1. INTRODUCTION

The extraction of patterns and rules is an active domain of data mining. These patterns have been extensively applied in various areas such as customer market basket analysis, web-log analysis, discovery of patterns from protein sequences network security and music analysis. These practical applications have been made possible by the development of robust algorithms [1, 24, 9]. However, the extraction of rules presents some non negligible limits, thus putting restrictions to its effective use. The main disadvantage stems from the huge set of extracted rules which may imply a second data-mining problem. The existence of a large number of rules makes them unmanageable for any human user in a decision making framework. This disadvantage directly comes from the type of knowledge the rules try to extract: frequent and confident rules. Although it may be useful when users want to discover frequent unobserved relations, it may be not when they want to discover unexpected relations.

It has been noticed that, in fact, 10% of the rules cause 90% of labor. The occurrence of a frequent event carries less information than the occurrence a rare or hidden event [3, 22, 20]. Therefore, it is more interesting to mine unexpected unfrequent events than frequent events which are all normally already known. Some works allow to discover unexpected knowledge from association rules like exceptions, surprising rules [16, 2]. For instance, *"seat belt* and child $\rightarrow$ *danger"* is an exception rule according to the strong rule *"seat belt* $\rightarrow$ *safe"*. But these rules cannot take time into account whereas the data stored in data warehouses are historical data. For instance, purchases are reported every day or every hour. Furthermore, data are often aggregated according to several dimensions in a data cube. For instance, sales are aggregated according to the shop, city, customer group, customer age, etc.

Even if various approaches have been proposed for discovering unexpected rules thanks to association rules, there is no approach that combines unexpected rules and time in a multidimensional framework.

This paper thus aims at introducing unexpected multidimensional sequential rules. This new kind of rules highlights, in a multidimensional framework, correlations between events through time. Unexpected multidimensional sequential rules are unfrequent and confident rules. They represent deviations from common and well-known behaviors. These common behaviors can be modeled by multidimensional sequential rules (frequent and confident). Indeed, a common behavior corresponds to rules which often occur in the data set, and so are frequent. These rules should be highly confident to be considered. Unexpected multidimensional sequential rules are *hidden* by a a dominant rule. They represent potential future common rules and surprising behaviors that have to be handled.

In this paper, we present the approaches to discover unexpected knowledge and we present the concepts related to

multidimensional sequential patterns. We introduce the fundamental concepts related to our approach as well as algorithms allowing its implementation. Experiments carried out on real data are reported and highlight the interest of our approach.

2. MOTIVATING EXAMPLE

In order to illustrate our approach, we consider the following running example that will be used throughout the paper.

Let us consider a data cube DC in which transactions issued from customers are aggregated. We assume that DC contains the number of sales reported over the following dimensions: D is the date of sale (ranging from $1 \dots 12$), C is the city where transactions have been issued (considering several cities: N.Y., L.A., etc.), A is the age of customers (considering three discretized values, denoted by Y (young), M (middle) and O (old)), CH is the hobby of the customers (walking, surf, golf, etc.), P is the product sold (car, bike, etc.), and M is the aggregated number of sales (measure).

For instance, the first cell of DC (see the first tuple Fig. 1) means that, at date 1, 12 *little cars* were bought in $N.Y$ by *young* customers who like *golf*.

D Date	C City	A Age	CH Customer –Hobby	P Product	M
1	NY	Young	Golf	Little_car	12
				Bike	14
				Van	2
...	...	...	...	...	...
...	...	...	...	...	
12	L.A	Old	Walking	Bike	19
				Shoes	25
				...	...

Figure 1: Cube DC

Let us now assume that we want to extract all unexpected multidimensional sequential rules that deal with the age of customers, the hobby of the customers, the products they bought and that are enough frequent and confident with respect to the cities where transactions have been issued. We considered unexpected rules compared to common rules who are highly frequent and confident.

As an illustration, we suppose the following common rule: *If young customer has recently received his car license then he will buy a little car.* An unexpected rule according to this common rule may be: *If a young customer who loves surfing has received his car license then he will buy a van.* This rule has a low support but it is highly confident. In this case, extra information specified in the if-part (*love surfing*) changes the conclusion (little car to van).

3. RELATED WORK

In this section, we report works on mining unexpected rules and we introduce the multidimensional sequential pattern mining problem formulation.

3.1 Unexpected Knowledge

In the literature, it has been noted that a frequent event carries less information than a rare or hidden event [3, 22, 20]. Therefore, it is often more interesting to discover unexpected and non-frequent events than frequent ones. The main problem is to determine what is an interesting event (low support is not enough) and how to extract it.

Among works on discovering unexpected rules , there are two types of approaches. On the one hand, *user-driven methods* require the intervention of a human expert. On the other hand, *data-driven* methods try to autonomously discover more restrictive rules. In the literature, the terms *subjective* and *objective* can also be found to characterize the user-driven and the data-driven approaches [13].

In user-driven methods, a human expert has to intervene at least on one of the following points, in order to:

- Determinate some restrictions over the attributes which can potentially occur in the relations [15].
- Describe data with a hierarchy [5].
- Indicate the potentially useful rules according to prior beliefs [7].
- Eliminate all the uninteresting rules in a first step so that other rules can automatically appear in subsequent steps [12].

Thus, user-driven approaches are quite constraining since they also require an expert intervention as soon as data are updated. That is the reason why we focus on data-driven approaches.

Data-driven approaches are divided into two sub-fields. Some works use interestingness measures that are different from usual confidence and support measures [21, 4]. Other approaches try to discover *unexpected* knowledge which are not extracted by classical algorithms.

Meaningful rules are not necessary frequent. [3] and [8] try to discover unfrequent itemsets which are highly correlated. In [25], the authors try to obtain *peculiarities* which are defined as highly confident unfrequent rules according to a nearness measure. These peculiarities are significantly different from the rest of the individuals. [22] extracts unusual sequences where items with low appearance probability appear together. In this case, the sequences are quite surprising. This approach does not use the support to determine the frequency of a sequence but the authors use entropy measures to detect surprising sequences.

Suzuki's approach is very interesting and consists in looking for *exceptions* that occur in a database [16, 6, 17, 18, 20]. The presence of an attribute interacting with another may change the consequent of a strong rule. For instance, *"seat belt and child → danger"* is an exception rule according to the strong rule *"seat belt → safe"*. The general form of an exception rule is as follows:

$$X \Rightarrow Y, XZ \Rightarrow \neg Y, X \not\Rightarrow Z$$

$X \Rightarrow Y$ is a *common sense rule*, $XZ \Rightarrow \neg Y$ is an *exception* rule where $\neg Y$ can be a concrete value E. $X \not\Rightarrow Z$ is the

reference rule. In [6] and [20], the authors use 5 user-defined parameters to describe this rules. In general terms, they try to discover interaction between attributes: in [17], X represents *antibiotics*, Y *recovery*, Z *staphylococci* and E *death*. The following rule can be discovered: *with the help of antibiotic, the patient usually recovered, unless staphylococci appears; in this case, antibiotic combined with staphylococci may result to death"*. This type of rules is very interesting and cannot be detected by association rule algorithms. In [18], the authors define 5 thresholds in order to extract the pairs. However, a strict specification of the different thresholds for a data set can lead to no discovery. In [16], the authors propose a solution to this problem with an avl tree for each threshold to easily and efficiently update the values.

In [2], the authors define the *anomalous rules*. Anomalous rules are association rules which are verified when the common rules fail. More formally, let X, Y and A be three itemsets, $X \rightsquigarrow A$ is an " anomalous rule " according to the rule $X \Rightarrow Y$ where A is the anomaly, if the following conditions hold:

- $X \Rightarrow Y$ is a strong rule (support and confidence)
- $X\neg Y \Rightarrow A$ is a confident rule
- $XY \Rightarrow \neg A$ is a confident rule.

This approach is based on support ($MinSupp$) and confidence ($MinConf$) thresholds since the rules are a case of association rules.

According to the definition of Suzuki, these rules are semantically different. Moreover, this approach do not require the existence of the "conflictual" itemset (Z). The authors define the confidence of an anomalous rule as follows: $conf_R(X \rightsquigarrow A) = \frac{supp_R(X \cup A)}{supp_R(X)}$ where R corresponds to the subset of the database which contains X and which does not verify the rule $X \Rightarrow Y$. In other words, R is the data set which does not verify the rule and which may contain an anomaly. Since $supp_R(X)$ is equal to $supp(X \cup \neg Y)$ on the whole database, the support can easily be computed as $supp(X) - supp(X \cup Y)$.

The confidence of an anomalous rule $X \rightsquigarrow A$ can then be defined as follows: $conf_R(X \rightsquigarrow A) = \frac{supp(X \cup A) - supp(X \cup Y \cup A)}{supp(X) - supp(X \cup Y)}$. The authors estimate that in practical case, A is an item and no an itemset. Their approach is based on the fact that even if $X \cup A$ and $X \cup Y \cup A$ may be unfrequent, they are the extensions of frequent itemsets X and $X \cup Y$. Therefore, their problem is reduced to the support computation of $L \cup i$ for each frequent itemset L and item i which is potentially an anomaly. The extraction of such anomalous rules is Apriori-based.

There are several objective approaches which try to mine particular types of rules. Each type of rules has a particular semantic. All these approaches are managed in a database framework without taking time into account. Moreover, they only consider one dimension in the association rule mining.

3.2 Multidimensional Sequential Patterns

In this section, we describe the problem of mining multidimensional sequential rules. Three works try to combine several analysis dimensions [10], [23] and [11]. We present the formal definition from [11] because it is the most general framework.

Rules combine several dimensions but they also combine these dimensions over time. In the rule *A customer who bought a surfboard together with a bag in NY later bought a wetsuit in SF*. NY appears before SF, and $surfboard$ appears before $wetsuit$.

Let us consider a data cube DC defined on the set of dimensions D_n, and a partition of D_n into four sets:

- D_T for the temporal dimensions, the set of dimensions that are meant to introduce an order between events (*e.g. time*);
- D_A for the *analysis* dimensions, the set of dimensions on which the rules will be built;
- D_R for the *reference* dimensions, the set of dimensions on which the counting will be based (*customer ID*);
- D_I for the *ignored* dimensions, the set of dimensions which are not taken into account in the mining process.

Each tuple $c = (d_1, \ldots, d_n)$ can thus be written as $c = (i, r, a, t)$ with i being the restriction on D_I of c, r its restriction on D_R, a the restriction on D_A, and t the restriction on D_T.

Given a cube DC, the set of all tuples in DC having the same value r on D_R is called a *block* and we denote the set of blocks from cube DC by B_{DC,D_R} . Thus, each block $\mathcal{B}_r$ in B_{DC,D_R} is identified by the tuple r that defines it.

During the mining of multidimensional sequential patterns, the set D_R identifies the blocks of the data cube to be considered when computing the support. For this reason, this set is called *reference*. The support of a sequence is the proportion of blocks embedding it. Note that with usual sequential patterns, and sequential patterns from [10], this set is reduced to one dimension (*cid* in [10]). Besides, the set D_A describes the *analysis* dimensions, so patterns defined by these dimensions will be found in the multidimensional sequential pattern mining. Note that with usual sequential patterns mining, we only consider a unique analysis dimension corresponding for instance to the products purchased or the web pages visited. Finally, set D_I describes the *ignored* dimensions, which are used neither to define the date, nor the blocks, and which are not present within the patterns mined.

In our running example, we consider $D_I = \emptyset, D_R = \{C\}, D_T = \{D\}$ and $D_A = \{A, CH, P\}$.

According to the dimension set partition, a *multidimensional item* e is a m-tuple defined over the set of the m D_A dimensions. We consider $e = (d_1, d_2, \ldots, d_m)$ where $d_i \in Dom(D_i) \cup \{*\}, \forall D_i \in D_A$ and where $*$ stands for

the *wild-card* value. For instance, $(Young, Golf, Bike)$ and $(*, Walking, *)$ are two multidimensional items defined with respect to three analysis dimensions.

A *multidimensional itemset* $i = \{e_1, \ldots, e_k\}$ is a non-empty set of multidimensional items. According to the notion of itemset, two *comparable items* cannot appear in the same itemset. For instance, $\{(Young, Tennis, car)(Old, *, bike)\}$ is a multidimensional itemset whereas $\{(M, Tennis, *), (M, *, *)\}$ cannot be an itemset since $(M, Tennis, *) \subset (M, *, *)$.

A *multidimensional sequence* $\varsigma = \langle i_1, \ldots, i_l \rangle$ is a non-empty ordered list of multidimensional itemsets. For instance, $\varsigma_1 = \langle \{(Young, *, car), (*, golf, *)\}\{(Young, Music, guitar)\}\rangle$ is a multidimensional sequence. A multidimensional sequence can be included into another one.

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

For instance, the sequence $\varsigma_1 = \langle \{(Young, *, car), (*, golf, *)\}$ $\{(Young, Music, guitar)\}\rangle$ is a subsequence of the sequence $\varsigma_2 = \langle \{(Young, *, car), (*, golf, *)\}\{(Old, Surfing, *)\}$ $\{(Young,$ $Music, *)\}\rangle$, denoted by $\varsigma_1 \subseteq \varsigma_2$.

We consider that each block defined over D_R contains one multidimensional data sequence, which is identified by that block. A block *supports* a sequence ς if ς is a subsequence of the data sequence identified by that block.

Let us define the *support* of a multidimensional sequence as the number of blocks defined over D_R containing this sequence. Given a user-defined minimal support threshold, denoted σ $(0 \leq \sigma \leq 1)$, the goal of *multidimensional sequential pattern mining* is to extract all the sequences S in DC such that $support(S) \geq \sigma$.

When considering the *classical* case of sequential patterns, the sets of analysis, reference, and order dimensions consist of only one dimension (usually the *product*, *customer_id* and *time* dimensions). We note that even in this classical case, the number of frequent sequential patterns discovered from a database can be huge. The problem is worse in the case of multidimensional patterns since the multidimensional framework produces more patterns than the classical framework.

4. UNEXPECTED MULTIDIMENSIONAL SEQUENTIAL RULES

4.1 Overview

Our goal is to extract unexpected rules which are often hidden by common rules with high support. The main idea, in this paper, is to use wild-card values in the if-part of a common rule. We want to instantiate at least one wild card value in the antecedent of a common rule in order to detect a different and unexpected conclusion.

Let CR be a common rule (frequent and confident):

$$CR : P \rightarrow Q$$

where P and Q are two multidimensional sequences.

According to common rule CR, an unexpected rule UR is an unfrequent and confident rule as follow:

$$UR : P_{specialized} \rightarrow Q'$$

$P_{specialized}$ is an *instantiation* of P. At least one wild-card value from P has been instantiated with an analysis dimension domain value. Q' is different from Q. UR is unfrequent but confident.

Before introducing the formal definition of unexpected multidimensional sequential rules, it is necessary to formalize multidimensional sequential rule concept, the instantiation of wild-card values in the if-part and the difference of conclusions.

4.2 Multidimensional Sequential Rules

Let $\alpha = \langle i_1, i_2, \ldots i_k, i_{k+1}, \ldots, i_n \rangle$ be a multidimensional sequence where each i_j represents a multidimensional itemset. A *multidimensional sequential rule* R is an implication:

$$R : \langle i_1, i_2, \ldots i_k \rangle \rightarrow \langle i_{k+1}, \ldots, i_n \rangle$$

As for association rules, the relevance of a multidimensional rule is indicated by its *support* and its *confidence*. The support of R is equal to:

$$support(R) = support(\langle i_1, i_2, \ldots i_k, i_{k+1}, \ldots, i_n \rangle)$$

The confidence of R is equal to:

$$Conf(R) = \frac{support(\langle i_1, i_2, \ldots i_k, i_{k+1}, \ldots, i_n \rangle)}{support(\langle i_1, i_2, \ldots i_k \rangle)}$$

4.3 Specification of wild-carded premises

In order to discover unexpected multidimensional sequential rules, we have to instantiate wild-carded premises of common rules.

We define the following functions :

- $*_\lambda(x)$ such that $*_\lambda(x) = x$
- a_{i_λ} such that

$$a_{i_\lambda}(x) = \begin{cases} a_i & \text{if} \quad x = a_i \\ \emptyset & \text{otherwise} \end{cases}$$

These functions are associated with the dimension values of a multidimensional item. Thus, for a multidimensional item $C = (t_1, t_2, \ldots, t_m)$, we can construct the function C_λ:

$$X = (x_1, \ldots, x_m) \mapsto C_\lambda(X) = (t_{1_\lambda}(x_1), t_{2_\lambda}(x_2), \ldots, t_{m_\lambda}(x_m))$$

As an example, we can construct from an item $C = (a, *, c)$ the function $C_\lambda = (a_\lambda, *_\lambda, c_\lambda)$.

Definition 2 (Instance). *Let* C *and* X *be two multidimensional items,* X *is said to be an instance of* C *if* $C_\lambda(X) = X$.

For instance, $X = (a, b, c)$ is an instance of $C = (a, *, c)$ since $C_\lambda(X) = X$. We can denote that X is an instance of itself ($X_\lambda(X) = X$).

DEFINITION 3 (PSEUDO-INSTANCE OF AN ITEMSET). *Let $i = \{e_1, e_2, \ldots, e_m\}$ and $i' = \{e'_1, e'_2, \ldots, e'_{m'}\}$ be two itemsets such that $m \leq m'$, i is said to be a pseudo-instance of i' if there exist some integers $1 \leq k_1 \leq k_2 \leq \ldots \leq k_m \leq m'$ such that $\forall e_j \in i$, $e'_{k_j\lambda}(e_j) = e_j$.*

We use the term *pseudo* because itemset i can be smaller than i'. Besides, if they have the same size, we can use the term *instance*. For instance, $\{(a, b, c)\}$ is a pseudo-instance of $\{(a, *, c), (*, b, b)\}$ and $\{(a, b, c), (*, b, b)\}$ is an instance of $\{(a, *, c), (*, b, b)\}$ since they have the same size.

DEFINITION 4 (PSEUDO-INSTANCE OF A SEQUENCE). *Let $s = \langle i_1, i_2, \ldots i_m\rangle$ and $s' = \langle i'_1, i'_2, \ldots i'_{m'}\rangle$ be two sequences such that $m \leq m'$, s is said to be a pseudo-instance of s' if there exist some integers $1 \leq k_1 \leq k_2 \leq \ldots \leq k_m \leq m'$ such that $\forall i_j \in s$, i_j is a pseudo instance of i'_{k_j}.*

As an example, $\langle\{(a, b, c)(a, *, d)\}\{(a, b, *)(b, *, *)\}\rangle$ is a pseudo-instance of $\langle\{(a, b, c)(a, *, d), (d, e, f)\}\{(a, *, *)(b, *, *)\}\rangle$.

Discovering an unexpected rule amounts to the discovery of a sequential rule where adding some information in the premise modifies the consequent. So we have to define an instanciation operation in order to substitute some wild-card values (*) by some dimension domain values. This operation is a specification of at least one wild-carded item in a sequence s' according to its instance in a sequence s. This operation is defined as follows:

DEFINITION 5 (INSTANTIATION). *Let s' and s be two sequences such that s is a pseudo-instance of s', the function $\iota(s', s)$ is the substitution of at least one item e'_i in s' with an instance of e'_i in s.*

$$\iota : sequence \times sequence \rightarrow set\ of\ sequences$$
$$\iota(s', s) \mapsto \{s''\ such\ that\ the\ following\ conditions\ hold\}$$

- *s'' is an instance of s'.*
- *$\exists$ items $e''_i \in s'', e'_i \in s'$ and $e_i \in s$ such that e_i is an instance of e'_i and $e''_i = [e_i/e'_i]$ where $[e_i/e'_i]$ is the substitution of e'_i by e_i.*

For instance, $\iota(\langle\{(a, *, c), (e, f, *)\}\{(*, c, d)\}\rangle, \langle\{(a, b, c)\}\rangle)$ is equal to the sequence $\langle\{(a, b, c), (e, f, *)\}\{(*, c, d)\}\rangle$. The wild-card value of the item $(a, *, c)$ has been instantiated by the value b according to its instance (a, b, c).

Instantiation operation does not return only one sequence but a set of sequences. This set can make its management non-trivial because of non-deterministic solution. However, we can produce a deterministic solution by greedily instantiating item by item.

4.4 Difference in the conclusion

In order to discover unexpected rules, we have to find a different conclusion from the instantiated premise. The conclusion of a multidimensional sequential rule is also a sequence, so we have to define the difference between two sequences.

DEFINITION 6 (DIFFERENCE). *Let $s = \langle i_1, i_2, \ldots, i_l\rangle$ and $s' = \langle i'_1, i'_2, \ldots, i'_{l'}\rangle$ be two sequences, s and s' are said to be different ($s \neq s'$) if $s \not\subseteq s'$ and $s' \not\subseteq s$.*

Two sequences s and s' are not comparable if s is more specific or general than s'. If we consider that two comparable sequences are different, we may discover many unexpected rules where the conclusion of an unexpected rule is just a more specific sequence than the conclusion of the common rule. So we consider two sequences as being different if they have at least one different item and if they are not comparable. As an example, the sequences $s_1 = \langle\{(a_1, b_1)\}\{(a_2, *)\}\rangle$ and $s_2 = \langle\{(a_1, b_1)\}\rangle$ are not different since $s_2 \subset s_1$. Sequences s_1 and $s_3 = \langle\{(a_2, b_2)\}\{(a_1, b_2)(a_2, b_1)\}\rangle$ are different since there is no relation between s_1 and s_3.

4.5 Unexpected Multidimensional Sequential Rules

We consider the following user-defined thresholds :

- $minCR$: the minimal support threshold for common rules,
- $maxUR$: the maximal support threshold for unexpected rules,
- $minUR$: the minimal support threshold for unexpected rules,
- $minConf$: the minimal confidence threshold (the same for all rules).

The threshold $minCR$ represents the minimal support value above which a rule can be considered as frequent. The threshold $minConf$ represents the minimal confidence value above which a rule can be considered as confident. We consider that a rule which is not frequent automatically cannot be a potential unexpected rules. In this way, we consider two thresholds, if the support of a rule is greater than $maxUR$, this rule is too *"frequent"* to be an unexpected rule. Indeed, we argue that unexpectedness is related to non frequent event. So we propose this threshold to highlight this difference. We can denote that $maxUR$ can be equal to $minCR$. In order to differentiate unexpectedness from noise, we introduce the threshold $minUR$. All patterns with support smaller than $minUR$ are considered as noise.

Figure 2 illustrates the use of different support thresholds. We consider that unexpected rules may occur enough in the data cube so that not to be considered as noise. Furthermore, the support of unexpected rules is weak enough not to be considered as a common rule.

A common rule CR is a frequent and confident rule as following :

$$CR : s_\alpha \rightarrow s_\beta\ s.t.\ supp(CR) > minCR\ and\ conf(RC) > minConf \quad (1)$$

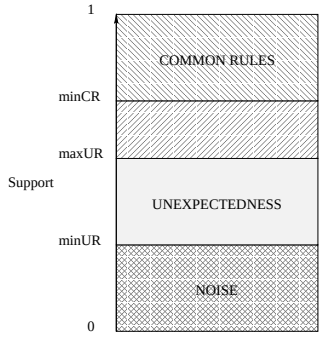

Figure 2: Different Support Thresholds

We consider only the instance CR's premise having a support greater than $minCR$:

$$s' \ s.t. \ supp(s') > minCR \ and \ s' \ is \ an \ instance \ of \ s_\alpha \quad (2)$$

An unexpected rule is a confident rule which premise is an instantiation of s_α with s' and verifies support conditions:

$$UR : \iota(s_\alpha, s') \rightarrow s_c \ s.t. \ minUR \leq supp(UR) \leq maxUR \ and \ s_c \neq s_\beta \quad (3)$$

We have to verify if there is no common rule with premise s_α. If there is such rule, UR is not considered as an unexpected rule.

$$VR : s_\alpha \rightarrow s_c \ s.t. \ conf(VR) < minConf \ and/or \ supp(RV) < maxUR \quad (4)$$

We have defined unexpected multidimensional sequential rules. In order to discover all of them, for each common rule RC (1), we have to discover the set of sequences S verifying (2), (3) and (4).

5. ALGORITHM

In this section, we define the algorithm for mining unexpected multidimensional sequential rules.

Algorithm 1 describes our approach. This algorithm is divided into three steps. First, multidimensional sequences are mined and stored in a prefix tree. Then, common rules are discovered in the prefix tree. Finally, the last step is about searching for unexpected rules.

Multidimensional sequences are mined according to support threshold $minUR$. These sequences are stored in a prefix tree. Subroutine $getFreqSet()$ performs this extraction. We use a *pattern growth* based method to mine multidimensional sequences. Subroutine $getFreqSet()$ is the most expensive operation of this algorithm. Indeed, it need several scans over the data cube DC to mine multidimensional sequences having a support greater than $minUR$. However, this subroutine can be called once with a lower support threshold. Then, from the returned prefix tree of multidimensional sequences, we can discover common rules and unexpected rules according to several parameter settings ($minUR, maxUR, minConf$ and $minCR$). As an example, the sequence extraction can be performed during the night; the decision maker performs then several unexpected rule mining during the day.

Common rules are obtained by one scan on the tree. Subroutine $getCR()$ only has to consider each node of tree $freqTree$ once (one tree traversal). This subroutine returns the set of common rules with respect to confidence threshold $minConf$ and support threshold $minCR$.

In the same way, $freqTree$ is traversed to discover the set of unexpected rules. Nevertheless, mining unexpected multidimensional sequential rule set has also to consider the set of common rules (denoted CRS). Indeed, $freqTree$ is traversed to find rules $r : p \rightarrow q$ where support and confidence conditions hold ($minUR \leq support(r) \leq maxUR$ and $conf(r) \geq minConf$). To be considered as a potential unexpected rule, it is necessary to find a premise p' in common rule set CRS such that p is an instance of p'. In this case, rule r is potentially an unexpected multidimensional sequential rule. To check it, it is necessary to verify the non-existence of a common rule $p' \rightarrow q$. If this condition holds, r is added to the set of unexpected multidimensional sequential rules URS. Otherwise, r is not an unexpected rule at all since r has the same consequent as a common rule (frequent and confident). The different operations on set CRS can be enhanced by using a more elaborated structure like hash tables with single or double hashing.

Algorithm 1: Mining Unexpected Multidimensional Sequential Rules

Data: $DC, D_A, D_T, D_R, minCR, maxUR, minUR, minConf$
Result: The set URS of unexpected multidimensional sequential rules

```
begin
    FreqTree ← getFreqSeq(DC, D_A, D_R, D_T, minUR)
    CRS ← getCR(FreqTree, minCR, minConf)
    URS ← ∅
    foreach rule r : p → q ∈ freqTree s.t.
    minUR ≤ supp(r) ≤ maxUR and conf(r) ≤ minConf do
        if ∃ premise p' ∈ CRS s.t. p is an instance of p' then
            if ∃seq x s.t. ι(p', x) → p and supp(x) ≥ minCR
            then
                if ∄p' → q ∈ CRS then
                    URS ← URS ∪ {r}
end
```

6. EXPERIMENTS

In this section, we report experiments performed on real data. They aim at showing the relevance of our approach. They have been performed on data issued from EDF (Electricité de France) marketing context and are in the scope of a research collaboration between EDF Research and Development and LIRMM laboratory. Indeed, EDF, the main French energy supplier and electricity producer, has resort to On-Line Analytical Processing (OLAP) applications to analyze internal and external data. In this context, EDF and LIRMM are developing efficient tools to discover unexpected temporal evolution within OLAP data.

We consider the simplified detailed table (Fig. 3), describing marketing activity on a very large EDF customer database (about 30 millions of residential customers).
The simplified database scheme is:
Marketing_Offer (#Offer, Offer, State_Of_Offer, Heating, Geography, Support_Of_Offer, Time)

Figure 3: Example of customers detailed table

#Offer	Offer	State_Of_Offer	Heating	Geography	Support_Of_Offer	Time
1	BIEN	Flux	Electrical	Bordeaux	Phoning	Jan 2003
2	BIEN	Flux	Gas	Montpellier	Mailing	Mar 2005
3	RENO	Stock	Electrical	Lyon	Phoning	Nov 2004
4	DCS	Flux	Gas	Lyon	Phoning	Feb 2003
5	RENO	Stock	Fuel	Paris	Mailing	Jan 2004
...	...	...	...	...	...	...

One tuple of Tab. 3 is related to one EDF marketing offer (Offer), at different states (State_Of_Offer) such as *current* ($FLUX$) or *closed* ($STOCK$), proposed:

- to customers characterized by their heating energy (Heating),
- by customers' relationship entities (Geography),
- on different supports (Support_Of_Offer) such as *mailing* or *phoning*,
- and at different dates (Time).

On this detailed table, we build a datacube structured by these 6 dimensions and the aggregated number of offers as a measure.

According to the dimension set partition, we consider one reference dimension ($D_R = \{\text{Geography}\}$), one temporal dimension ($D_T = \{\text{Time}\}$) and five analysis dimensions. The measure is transformed and put in the analysis dimensions.

As an example, according to the common rule $CR = \langle\{(*, FLUX, *, *, *)\}\{(*, FLUX, *, Electrical, *)\} \rightarrow \{(*, *, *, Electrical, *)\}\rangle$, the rule $UR = \langle\{(\mathbf{BIEN}, FLUX, *, *, *)\}\{(*, FLUX, *, Electrical, *)\} \rightarrow \{(\mathbf{BIEN,FLUX}, *, Electrical, *)\}\rangle$ is an unexpected rule.

In order to highlight the relevance of our approach, we report the behavior (runtime, number of mined unexpected rules, ratio of unexpected rules over rules) of our approach when a threshold ($minUR$, $minConf$) changes.

Figure 5 shows the runtime of our approach for mining the data cube when the confidence changes for four different $minUR$ values. This behavior is quite similar as observed on Fig. 8 where runtime is related to minUR changes according to four different confidence values. Indeed, when a threshold become lower, the number of unexpected rules (Figure 4 and 7) or potential unexpected rules increases. Indeed, the less thresholds are, the more space search is large.

Figure 6 shows the percentage of unexpected rules when the confidence changes. An an example, according to $minUR = 0.375$, only about 1% of rules such that their support is between $minUR$ and $maxUR$ are unexpected. When the confidence threshold becomes too low, this confidence does not represent a sufficient constraint and the percentage of unexpected rules become important. If Data Mining is about finding gems in a database, our approach is quite closed to this metaphor when the confidence threshold is enough constraining.

7. CONCLUSION AND FUTURE WORKS

In this paper, we define unexpected multidimensional sequential rules, by proposing to combine unexpectedness and time in order to mine such rules. The different concepts and the algorithm used to implement our approach are presented. Moreover, experiments on real data cube are reported and highlight the interest of our approach.

This work offers several perspectives. The efficiency of the extraction can be enhanced by discovering unexpected rules during pattern extraction and by directly using measure in the support counting. We should also take hierarchies into account in order to discover unexpected multidimensional sequential rules defined over several hierarchies level. We can also return the top k unexpected multidimensional sequential rules in order to focus on the most relevant ones. In order to generalize Suzuki's approach, we should duplicate the dimensions on D_T in D_A. Indeed, thanks to date, we can construct union between two sequences and then check if the conclusion becomes different.

8. REFERENCES

[1] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu. Sequential pattern mining using a bitmap representation. In *KDD*, pages 429–435, 2002.

[2] F. Berzal, J.-C. Cubero, and N. Marín. Anomalous association rules. In *IEEE ICDM Workshop Alternative Techniques for Data Mining and Knowledge Discovery.*, 2004.

[3] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. D. Ullman, and C. Yang. Finding interesting associations without support pruning. *IEEE Trans. Knowl. Data Eng.*, 13(1):64–78, 2001.

[4] L. Geng and H. J. Hamilton. Interestingness measures for data mining: A survey. *ACM Comput. Surv.*, 38(3), 2006.

[5] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *VLDB*, pages 420–431, 1995.

[6] F. Hussain, H. Liu, E. Suzuki, and H. Lu. Exception rule mining with a relative interestingness measure. In *PAKDD*, pages 86–97, 2000.

[7] B. Liu, W. Hsu, S. Chen, and Y. Ma. Analyzing the subjective interestingness of association rules. *IEEE Intelligent Systems*, 15(5):47–55, 2000.

[8] S. Ma and J. L. Hellerstein. Mining mutually dependent patterns. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 409–416, Washington, DC, USA, 2001. IEEE Computer Society.

[9] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. Mining sequential patterns by pattern-growth: The prefixspan approach. *IEEE Transactions on Knowledge and Data Engineering*, 16(10), 2004.

[10] H. Pinto, J. Han, J. Pei, K. Wang, Q. Chen, and U. Dayal. Multi-dimensional sequential pattern mining. In *CIKM2001*, pages 81–88. ACM, 2001.

[11] M. Plantevit, Y. W. Choong, A. Laurent, D. Laurent, and M. Teisseire. M^2SP: Mining sequential patterns among several dimensions. In *PKDD*. 2005.

[12] S. Sahar. Interestingness via what is not interesting.

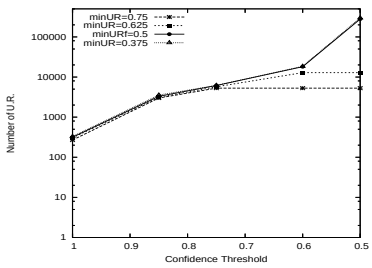

Figure 4: #Unexpected Rules over $minConf$ ($maxUR = 0.8$)

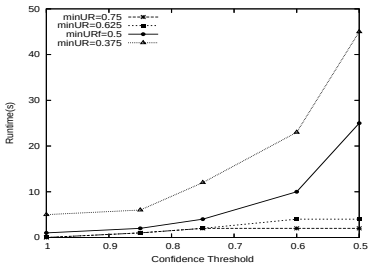

Figure 5: Runtime over Confidence ($maxUR = 0.8$)

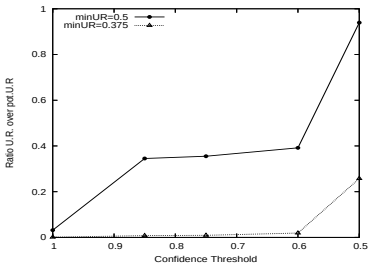

Figure 6: Percentage of Unexpected Rules over $minConf$ ($maxUR = 0.8$)

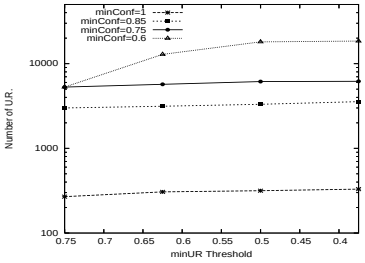

Figure 7: #Unexpected Rules over $minUR$ Threshold ($maxUR = 0.8$)

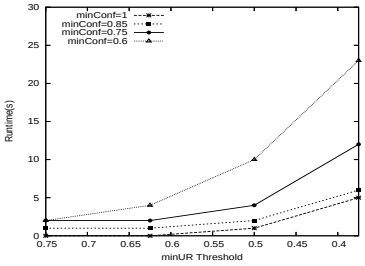

Figure 8: Runtime over $minUR$ Threshold ($maxUR = 0.8$)

In *Knowledge Discovery and Data Mining*, pages 332–336, 1999.

[13] A. Silberschatz and A. Tuzhilin. What makes patterns interesting in knowledge discovery systems. *IEEE Trans. Knowl. Data Eng.*, 8(6):1996, 1996.

[14] P. Smyth and R. M. Goodman. An information theoretic approach to rule induction from databases. *IEEE Trans. Knowl. Data Eng.*, 4(4):301–316, 1992.

[15] R. Srikant, Q. Vu, and R. Agrawal. Mining association rules with item constraints. In *KDD*, pages 67–73, 1997.

[16] E. Suzuki. Scheduled discovery of exception rules. In *DS '99: Proceedings of the Second International Conference on Discovery Science*, pages 184–195, London, UK, 1999. Springer-Verlag.

[17] E. Suzuki. In pursuit of interesting patterns with undirected discovery of exception rules. In *Progress in Discovery Science*, pages 504–517, 2002.

[18] E. Suzuki. Undirected discovery of interesting exception rules. *IJPRAI*, 16(8):1065–1086, 2002.

[19] E. Suzuki and M. Shimura. Exceptional knowledge discovery in databases based on information theory. In *KDD*, pages 275–278, 1996.

[20] E. Suzuki and J. M. Zytkow. Unified algorithm for undirected discovery of exception rules. *International Journal of Intelligent Systems*, 20(7):673–691, 2005.

[21] P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the right objective measure for association analysis. *Inf. Syst.*, 29(4):293–313, 2004.

[22] J. Yang, W. Wang, and P. S. Yu. Mining surprising periodic patterns. *Data Min. Knowl. Discov.*, 9(2):189–216, 2004.

[23] C.-C. Yu and Y.-L. Chen. Mining sequential patterns from multidimensional sequence data. *IEEE Transactions on Knowledge and Data Engineering*, 17(1):pp. 136–140, 2005.

[24] M. J. Zaki. Spade: An efficient algorithm for mining frequent sequences. *Machine Learning*, 42(1/2):31–60, 2001.

[25] N. Zhong, Y. Yao, and M. Ohshima. Peculiarity oriented multidatabase mining. *IEEE Trans. Knowl. Data Eng.*, 15(4):952–960, 2003.