HAL
open science

HYPE: Mining Hierarchical Sequential Patterns

Marc Plantevit, Anne Laurent, Maguelonne Teisseire

To cite this version:

Marc Plantevit, Anne Laurent, Maguelonne Teisseire. HYPE: Mining Hierarchical Sequential Patterns. DOLAP'06: ACM Ninth International Workshop on Data Warehousing and OLAP, Nov 2006, Arlington, VA, USA, pp.8. lirmm-00135019

HAL Id: lirmm-00135019

https://hal-lirmm.ccsd.cnrs.fr/lirmm-00135019

Submitted on 6 Mar 2007

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

HYPE: Mining Hierarchical Sequential Patterns

Marc Plantevit
LIRMM-Univ. Montpellier2
Montpellier, France
marc.plantevit@lirmm.fr

Anne Laurent
LIRMM-Polytech'Montpellier
Montpellier, France
laurent@lirmm.fr

Maguelonne Teisseire
LIRMM-Polytech'Montpellier
Montpellier, France
teisseire@lirmm.fr

ABSTRACT

Mining data warehouses is still an open problem as few approaches really take the specificities of this framework into account (e.g. multidimensionality, hierarchies, historized data). Multidimensional sequential patterns have been studied but they do not provide any way to handle hierarchies. In this paper, we propose an original sequential pattern extraction method that takes the hierarchies into account. This method extracts more accurate knowledge and extends our preceding M^2SP approach. We define the concepts related to our problems as well as the associated algorithms. The results of our experiments confirm the relevance of our proposal.

Categories and Subject Descriptors

I.5 [**Pattern Recognition**]: Miscellaneous; H. [**Information Systems**]: General

General Terms

Algorithms, Design, Theory.

Keywords

Multidimensional Sequential Patterns, Hierarchies, OLAP.

1. INTRODUCTION

Data mining techniques can be of a considerable help in the OLAP framework ([5]) where the user must make the best suitable decisions in a minimum amount of time. More precisely, data mining is a key step in the decision process when large volumes of multidimensional data are involved. Indeed, mined patterns or rules provide another outlook on the original data. However, some parameters are required to discover these rules. In particular, this mining requires minimal support that corresponds to the minimal frequency at which the patterns occur within the database. If the selected minimal support is too high, the number of rules discovered is small and the rules are too general to be useful. If the support is too low, the number of mined rules is very high, thus complicating their analysis. The decision maker is then faced with the following problem: how can the minimal support be lowered without revealing non-relevant rules? Or how can the minimal support be increased without losing the useful rules? Is it then necessary to make a trade-off between the quality of the extracted knowledge and the minimal support? It is thus difficult to mine intersting rules [15].

In this context, using hierarchies can help to solve this dilemma. It makes it possible to discover rules within several hierarchy levels. Thus, even if a high support is used, important knowledge with a too weak support in the database can be *included* in more general knowledge which is frequent. We thus wish to extend our previous proposal [13] to mine multidimensional sequential patterns by taking hierarchies into account.

Sequential patterns have been studied for more than 10 years [1], with a lot of research and industrial applications (*e.g.* user behavior, web log analysis, discovery of patterns from protein sequences, security). Algorithms have been proposed, based on the Apriori-based framework [18, 10, 2], or on other approaches [11, 7]. Some other work has been conducted on the discovery of frequent episods [9]. Sequential patterns have recently been extended to multidimensional sequential patterns by Pinto et al. [12], Plantevit et al. [13], and Yu et al. [17]. They aim at discovering patterns that take time into account and that involve several dimensions. For instance in [13], rules like *A customer who bought a surfboard with a bag in NY later bought a wetsuit in SF* are discovered.

Some approaches use hierarchies in the extraction of sequential patterns. Nevertheless, to our best knowledge, no work has combined the extraction of multidimensional sequential patterns and hierarchy management. No current method can extract knowledge like: *When the sales of soft drinks increase in Europe, exports of Perrier later increase in France and exports of soda later increase in the USA*, where *Perrier* is a kind of French carbonated soft drink. We propose a novel *HYPE* (HierarchY Pattern Extension) approach which is an extension of our previous M^2SP proposition [13]. The main unique feature of our approach is that no single hierarchy level is considered and that several levels can be mixed. Extracted sequential patterns are automatically associated with the most relevant hierarchy levels.

In this paper, we present concepts related to traditional sequential patterns and multidimensional ones, as well as approaches for managing hierarchies during knowledge extraction. We then introduce fundamental concepts related to our HYPE approach as well as algorithms allowing its implementation. Experiments carried out on synthetic data are reported and confirm the significance of our approach. We also show that using the hierarchies allows better management of joker values defined in the M^2SP approach.

2. HIERARCHIES AND DATA MINING

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
DOLAP'06, November 10, 2006, Arlington, Virginia, USA.
Copyright 2006 ACM 1-59593-530-4/06/0011 ...$5.00.

In this section, we present t sequential patterns as well as previously published approaches dealing with the problem of the extraction of sequential patterns in a multidimensional framework (several analysis dimensions). Then we underline why it is relevant to use the hierarchies during the process of extraction of sequential patterns and we make an provide of related work.

2.1 Sequential Patterns

An early example of research to discover patterns from sequences of events can be found in [4]. In this work, the idea is to highlight 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 to solving the problem of mining sequential patterns is the AprioriAll algorithm as presented in [9]. Given a user-defined threshold and 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. A sequential pattern is a sequence with a support greater than a user-defined one. 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 minimum support threshold, sequential pattern mining is carried out to find all frequent subsequences, i.e. subsequences whose occurrence frequency in the set of sequences is not less than the minimum 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*. The main objective of sequential pattern mining methods is then the most effective extraction. Algorithms have been proposed, based on the Apriori-based framework [18, 10, 2], or on other approaches [11, 7].

In the traditional framework (only one analysis dimension) for association rule or sequential pattern extraction, several works have taken hierarchies into account in order to allow extraction of accurate knowledge. In [16], the beginnings of hierarchy management in the extraction of association rules and sequential patterns are proposed. The authors suppose that hierarchical relations between the items are represented by a set of *taxonomies*. They make it possible to extract association rules or sequential patterns according to several levels of hierarchy. They modify the transactions by adding, for each item, all of its ancestors in associated taxonomies. Then they generate the frequent sequences while trying to filter with the maximun redundant information and by optimizing the process using several properties. However, this approach cannot be scalable in a multidimensional context. Indeed, it is unthinkable to add on each dimension the list of ancestors of one item in taxonomy for each transaction. In the worst case, that would multiply the size of the database by the maximum depth of a hierarchy for each analysis dimension, so it would be too expensive to scan this base.

The approach of J. Han et al. [6] is quite different. The authors tackle the association rule extraction problem, but this approach can also be adapted to sequential pattern extraction. Beginning at the highest level of the hierarchy, they extract rules at each level while lowering the support when descending in the hierarchy. The process is reiterated until no rules can be extracted or until the lowest level of the hierarchy is reached. However, this method does not make it possible to extract rules containing items of different levels. For example, $wine$ and $drinks$ cannot cohabit in such a rule. This method thus proposes the extraction of *intra level of hierarchy* association rules. It thus does not make it possible to answer general problems concerning the extraction of *inter levels of hierarchy* sequences. Furthermore, implementation of this approach in a multidimensional context can be discussed. If several taxonomies exist (one per dimension), does the user move on the same hierarchy levels on various taxonomies or combine these levels? This kind of extraction can be expensive in time since the knowledge discovery mechanism must be reiterated several times (depth of taxonomy).

We have presented the sequential patterns as well as works that take hierarchies into account in the knowledge extraction. Nevertheless the sequential patterns are sometimes quite poor in relation to the data they describe. Indeed, correlations are extracted within only one dimension (e.g. the *product* dimension) whereas a database can contain several other dimensions. This is why several works try to combine several analysis dimensions in the extraction of sequential patterns.

2.2 Multidimensional Sequential Patterns

Combining several analysis dimensions makes it possible to extract knowledge which describes the data in a better way. [12] is the first paper dealing with several dimensions in the sequential pattern framework. For instance, purchases are not only described by considering the customer ID and the products, but also by considering the age, type of customer (Cust-Grp) and the city where he/she lives. Multidimensional sequential patterns are defined over the schema $A_1, ..., A_m, S$, where the set of A_i stands for the dimensions describing the data and S stands for 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. Note that the sequences found by this approach do not contain several dimensions since the dimension time only concerns products. The product dimension is the only dimension that can be combined over time, so it is not possible to have a rule that indicates when b is bought in $Boston$ then c is bought in NY.

Contrary to [12], [13] proposes to mine such *inter pattern* multidimensional sequences. Several analysis dimensions can be found in the sequence, which allows for the discovery of rules as *A customer who bought a surfboard with a bag in NY later bought a wetsuit in LA*.

In [17], the authors consider sequential pattern mining in the framework of Web Usage Mining. Even though they consider three dimensions (pages, sessions, days), these dimensions are very particular since they belong to a single hierarchized dimension. Moreover, the sequences found describe correlations between objects over time by considering only one dimension, which corresponds to the web pages.

Note also the work of [3], which proposes a first order temporal logic based approach for multidimensional sequential pattern mining. [8] also proposes a new method of generation of the multidimensional sequences embedded in a set of transactions.

To our best knowledge, there is no approach that fully utilizes the hierarchies during the extraction of multidimensional sequential patterns. We thus propose to integrate the management of the hierarchies into M^2SP in order to allow a more complete extraction of knowledge, suitable in the OLAP framework.

2.3 Running Example

In order to illustrate the various concepts and definitions, we propose the following running example. Table 1 describes the purchases of product carried out in various cities of the world. For the hierarchies, we choose two dimensions, i.e. cities and products, whose respective taxonomies are indicated in Figures 1 and 2.

Table 1: Running Example

D (Date)	B ($Block_{ID}$)	Pl (Place)	P (Product)
1	1	Germany	beer
1	1	Germany	pretzel
2	1	Germany	M2
3	1	Germany	chocolate
4	1	Germany	M1
1	2	France	soda
2	2	France	wine
2	2	France	pretzel
3	2	France	M2
1	3	UK	whisky
1	3	UK	pretzel
2	3	UK	M2
1	4	LA	chocolate
2	4	LA	M1
3	4	NY	whisky
4	4	NY	soda

Figure 1: Taxonomy over the $Place$ dimension

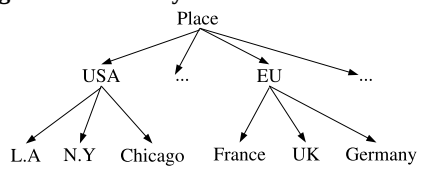
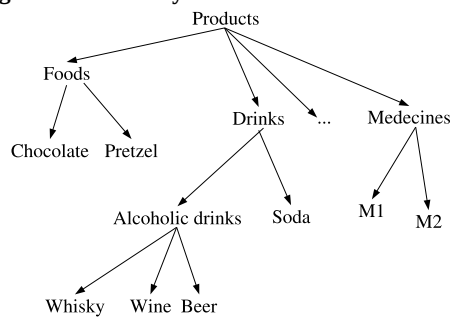


Figure 2: Taxonomy over the $Product$ dimension



3. CONTRIBUTIONS

In this section, we present our approach for the management of hierarchies in multidimensional sequential patterns. First, we define the concepts related to our approach [14]. Then, we propose then algorithms used to implement our approach.

3.1 Definitions

3.1.1 Dimension Set Partition

In order to allow users to freely customize the extraction, we consider a partition of the dimension set. Let us consider a database DB where data are described with respect to n dimensions. We consider a 3-bin partitioning of the dimensions: the set of dimensions that will be contained within the rules (*analysis dimensions*) is denoted by D_A; the set of dimensions which the counting will be based on (*reference dimensions*) is denoted by D_R; and the set of dimensions that are meant to introduce an order between events (*e.g.* $time$)[1] is denoted by D_T. Each tuple $c = (d_1, \ldots, d_n)$ can thus be denoted by $c = (r, a, t)$ with r being the restriction on D_R, a the restriction on D_A and t the restriction on D_T.

Given a table DB, the set of all tuples in DB having the same value r on D_R is said to be a *block* denoted by $\mathscr{B}_{DB,D_R}$ on the set of blocks from table DB. The block concept is necessary to define the support of a multidimensional sequence. Its application in our running example is trivial since $|D_R| = 1$ and the different blocks are described in Figure 3. We can imagine that these blocks have been built by grouping transcactions that share the same values on several dimensions (e.g. age, customer-group, etc).

Figure 3: Block Partition of DB (figure 1) according to $D_R = \{B\}$

Figure 4: block (1)

D	B	Pl	P
1	1	Germany	beer
1	1	Germany	pretzel
2	1	Germany	M2
3	1	Germany	chocolate
4	1	Germany	M1

Figure 5: block (2)

D	B	Pl	P
1	2	France	soda
2	2	France	wine
2	2	France	pretzel
3	2	France	M2

Figure 6: block (3)

D	B	Pl	P
1	3	UK	whisky
1	3	UK	pretzel
2	3	UK	M2

Figure 7: block (4)

D	B	Pl	P
1	4	LA	chocolate
2	4	LA	M1
3	4	NY	whisky
4	4	NY	soda

3.1.2 Taxonomies

In our multidimensional framework, we consider that there are hierarchical relations on each analysis dimension[2]. We consider that these hierarchical relations are materialized in the form of *taxonomies*. A taxonomy is a directed acyclic graph. The edges are *is-a* relation. The *Specialization* relation is then from root to leaves. Each analysis dimension thus has a taxonomy which makes it possible to represent hierarchical relations between the elements of its domain.

[1]All dimension sets which introduce an order relation can be considered.

[2]This relation may be reduced to the tree of depth 1 where the root is labelled by * if no hierarchy is defined.

Let $T_{D_A} = \{T_1, \ldots, T_m\}$ be the set of taxonomies associated with analysis dimensions, where: **(i)** T_i is the taxonomy representing hierarchical relations between the elements from the domain of the analysis dimension D_i; **(ii)** T_i is a direct acyclic graph; **(iii)** $\forall$ node $n_i \in T_i,\ label(n_i) \in Dom(D_i)$.

We write $\hat{x}$ an ancestor of x according to the associated taxonomy and $\check{x}$ one of its descendants. For instance, $drinks = \widehat{soda}$ means that $drinks$ is an ancestor of $soda$ according to the *Generalization/Specialization* relation. More precisely, $drinks$ is a more general instance than $soda$.

3.1.3 Hierarchies and Data

Each analysis dimension D_i from a transaction b of DB cannot be instantiated with a value d_i of which the node associated to the label d_i in the taxonomy T_i is a leaf. Formally, $\forall d_i \in \pi_{D_i}(B)$,- $\forall$ node n_i *such that* $label(n_i) = d_i \nexists$node n' *such that* $n' = \check{n}_i$ (n_i leaf). For instance, the transaction database cannot contain the value $drinks$ since there are some more specific instances in the taxonomy ($soda, wine$).

3.1.4 h-generalized Item, Itemset and Sequence

We now define the fundemental concepts of h-generalized item, itemset and sequence.

DEFINITION 1 (MULTIDIMENSIONAL H-GENERALIZED ITEM). *A multidimensional h-generalized item $e = (d_1, \ldots, d_m)$ is a tuple defined over the set of the m D_A dimensions such that $d_i \in \{label(T_i)\}$.*

Contrary to the transactions of DB, multidimensional h-generalized items can be defined with any value d_i whose associated node in the taxonomy is not a leaf. For instance $(drinks, USA)$, $(soda, France)$ are some multidimensional h-generalized items that are defined on the analysis dimensions $product$ and $place$.

Since multidimensional h-generalized items are instantiated on various levels of hierarchy, it is possible that two items are comparable, i.e. item is more *specific or general* than another. In order to not complicate the notations, we directly use the concept of ancestor on the item and the transaction without locating them in the corresponding taxonomy.

DEFINITION 2 (HIERARCHICAL INCLUSION). *Let e and e' be two different multidimensional h-generalized items, $e = (d_1, \ldots, d_m)$ and $e' = (d'_1, \ldots, d'_m)$, we say that:*

- *e is more general than e' ($e >_h e'$) if $\forall d_i, d_i = \hat{d'_i}$ or $d_i = d'_i$;*
- *e is more specific than e' ($e <_h e'$) if $\forall d_i, d_i = \check{d'_i}$ or $d_i = d'_i$;*
- *e and e' are incomparable if there is no relation between them ($e \not>_h e'$ and $e' \not>_h e$).*

For instance, we have:

- $(USA, drinks) >_h (USA, soda)$;
- $(France, wine) <_h (EU, Alcoholic\ drinks)$;
- $(France, wine)$ and $(USA, soda)$ are incomparable.

DEFINITION 3. *A transaction b supports an item e if $\Pi_{D_A}(b) <_h e$.*

As an example, the transaction $(1, 1, France, wine)$ supports the item $(EU, alcool)$.

DEFINITION 4 (MULTIDIMENSIONAL H-GENERALIZED ITEMSET). *A multidimensional h-generalized itemset $i = \{e_1, \ldots, e_k\}$ is a non-empty set of multidimensional h-generalized items where all items are incomparable.*

Two comparable items cannot be present in the same itemset since we adopt a set-theoretic point of view. Moreover we prefer to represent the most precise possible information within an itemset. For instance, $\{(France, wine), (USA, soda)\}$ is a multidimensional h-generalized itemset whereas $\{(France, wine), (EU, Alcoholic\ drinks)\}$ is not such an itemset because $(France, wine) <_h (EU, Alcoholic\ drinks)$.

DEFINITION 5 (MULTIDIMENSIONAL H-GENERALIZED SEQUENCE). *A multidimensional h-generalized sequence $s = \langle i_1, \ldots, i_j \rangle$ is a non-empty ordered list of multidimensional h-generalized itemsets.*

For instance, $\langle \{(France, wine), (USA, soda)\}, \{(Germany, beer)\} \rangle$ is a multidimensional h-generalized sequence. Multdimensional sequences can be included into another one:

DEFINITION 6 (SEQUENCE INCLUSION). *A multdimensional 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 $a_1 \leq_h b_{j_1}, a_2 \leq_h b_{j_2}, \ldots, a_l \leq_h b_{j_l}$.*

The inclusion of the multidimensional sequences must respect the hierarchical inclusion of the multidimensional h-generalized items. As an example:

- The sequence $\langle \{(France, wine)\}, \{(Germany, beer)\} \rangle$ is a subsequence of $\langle \{(France, wine), (USA, soda)\}, \{(Germany, beer)\} \rangle$;
- The sequence $\langle \{(France, wine)\}, \{(Germany, beer)\} \rangle$ is a subsequence of $\langle \{(France, Alcoholic\ drinks), (USA, drinks)\}, \{(EU, Alcoholic\ drinks)\} \rangle$;
- The sequence $\langle \{(EU, wine)\}, \{(Germany, beer)\} \rangle$ is not a subsequence of the sequence $\langle \{(France, wine), (USA, soda)\}, \{(Germany, beer)\} \rangle$ because $(EU, wine) \not\leq_h (France, wine)$, the hierarchical inclusion between item is not respected here.

3.1.5 Support

Computing the support of a multidimensional h-generalized sequence is equivalent to counting the number of blocks defined over the reference dimensions D_R which support the sequence. A block supports a multidimensional h-generalized sequence if it is possible to find a set of tuples which satisfies it. All itemsets from the multidimensional h-generalized sequence must be found on various dates within the domain of D_t such that the order of the itemsets respects the sequentiality.

DEFINITION 7. *A block B supports a sequence $\langle i_1, \ldots, i_l \rangle$ if $\forall j = 1 \ldots l,\ \exists d_j \in Dom(D_t)$, for each item e from i_j, $\exists t = (r, e, d_j)$ where $t = (r, \check{e}, d_j) \in B$ and $d_1 < d_2 < \ldots < d_l$.*

Thus, the *support* of a multidimensional h-generalized sequence is the number of blocks defined over D_R which contain this sequence.

According to a user-defined minimal treshold, a *multidimensional h-generalized sequential pattern* is a sequence whose support is greater than the minimal threshold.

EXAMPLE 1. *According to our running example database DB, let us consider $D_R = \{B_{id}\}$, $D_A = \{Place, Product\}$, $D_T = \{Date\}$, $minimal\ support = 2$, and $\varsigma = \langle\{(EU, Alcoholic\text{-}drinks), (EU, pretzel)\}\{(EU, M2)\}\rangle$. The sequence is frequent if at least two blocks of the partition of DB support the sequence.*
1. block $(\mathbf{1})$ *(Fig. 4). According to the taxonomies, Germany is more specific than EU and beer is an instance of Alcoholic drinks. Thus, at the date 1, there is the first itemset $\{(EU, Alcool), (EU\text{-}, pretzel)\}$ of ς. The last itemset $\{(EU, M2)\}$ is contained at a date later (2). The sequence ς is supported by this block.*
2. block $(\mathbf{2})$ *(Fig. 5). France is an instance of EU and wine is more specific than Alcoholic drinks. The sequence ς is supported by this block.*
3. block $(\mathbf{3})$ *(Fig. 6). UK is an instance of EU and whisky is an instance of Alcoholic drinks. This block then supports the sequence ς.*
4. block (4) *(Fig. 7). This block does not support the sequence ς since the $place$ dimension does not contain any instance of EU.*
The support of ς is 3. The sequence is thus frequent.

3.2 The HYPE Algorithm Proposal

3.2.1 Overview

We briefly describe the general behavior of our approach before presenting the algorithms that allow the extraction of multidimensional h-generalized sequential patterns.

HYPE is divided into two phases. Firstly, the maximally specific items are extracted (we say an item is maximally specific if there are no more specific items). We think the maximally specific items are an alternative to the huge amount of extracted knowledge. Indeed, they make it possible to *factorize* knowledge. The user can infer more general knowledge in a post-processing step. Secondly, the multidimensional h-generalized sequences are mined in a further step. These sequences are generated and validated from the frequent maximally specific items.

However, the fact of using maximally specific items to generate the frequent sequences does not enable us to extract all knowledge embedded in the database. Indeed, some sequences whose first items are not maximally specific cannot be mined. Some longer sequences cannot then be mined (blocks quickly support more general knowledge). However, this deficiency is relative because these non-mined sequences often describe too general knowledge and provide little information to the user.

It is not necessary to prune the taxonomies in a preprocessing step. Indeed, this operation can easily be carried out during the multidimensional h-generalized sequential pattern mining process.

3.2.2 Generation of frequent items

Multidimensional h-generalized items are the basis of multidimensional h-generalized sequential pattern mining. They are sequences whose length is 1. For scalability, items cannot be mined in only one scan. Indeed, the cartesian product of analysis dimension domains cannot be considered in applications where the number of dimensions and the cardinality of their domains can be very large. If the number of analysis dimensions is m, then the number of generated items χ is exponential according to m:

$$2^m \leq \chi \leq \sum_{i=1}^{m} \binom{m}{i} i^k \ where\ k = \max |Dom(D_i)|$$

We thus consider that such an approach can jeopardize the scalability of the extraction. It is thus necessary to define a method which limits the number of candidate items and the number of database scans. In order to generate the multidimensional h-generalized items, we adopt a levelwise algorithm that only considers items that have at least some probabilty of being frequent. To this end, we consider a lattice whose lower bound is the $(*, \ldots, *)$ multidimensional item[3]. This lattice is partially built from $(*, \ldots, *)$ up to the frequent items[4] containing as few $*$ as possible. At level i, i values are specified. Then items at level i are combined to build a set of candidates at level $i + 1$. The process is iterated m times until the complete set of multidimensional h-generalized items is obtained. Two frequent items are combined to build a candidate if they are $\bowtie$-compatible, i.e. they share a sufficient number of values over analysis dimensions (Definition 8). For instance, $(a, *, c)$ and $(*, b, c)$ are $\bowtie$-compatible. Items $(a, b, *)$ and $(a, b, *)$ are not compatible.

DEFINITION 8 ($\bowtie$-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 $\exists \Delta = \{D_{i_1}, \ldots, D_{i_{n-2}}\} \subset \{D_1, \ldots, D_n\}$ 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 $d'_{i_n} = *$.*

The join operation is defined as follows:

DEFINITION 9 (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 $E \bowtie E' = \{e \bowtie e'\ s.t.\ (e, e') \in E \times E' \wedge$ e and e' are $\bowtie$-compatible$\}$*

For instance, suppose there are 3 items with only one value instantiated: $(a, *, *)$, $(*, b, *)$ and $(*, *, c)$. Items $(a, *, *)$ and $(*, b, *)$ are $\bowtie$-compatible, then $(a, *, *)$ is joined with $(*, b, *)$ into item $(a, b, *)$. The join operation is applied on all $\bowtie$-compatible items. So the result of the join operations are: $(a, b, *)$, $(a, *, c)$ and $(*, b, c)$. The process is reiterated to give item (a, b, c).

3.2.3 Generation of Frequent Sequences

The frequent items give all frequent sequences containing one itemset consisting of a single item. To mine the frequent multidimensional h-generalized sequences, we follow the Apriori-like paradigm. Indeed, the multidimensional framework keeps the anti-monotony of the support (all subsets of a frequent set are frequent). Once 1-frequent items are mined, the candidate sequences of size k ($k \geq 2$) are generated and validated to keep the frequent items. We use a prefixed-tree-like structure as [10] to efficiently maintain the set of frequent sequences.

3.2.4 Counting the Support of a Sequence

Support counting is one of the main operations of the data mining process.

The reference dimensions enable to identify all blocks which may support a sequence ς. The enumeration of all blocks is essential to compute the support of a sequence and thus determine if the sequence is frequent. Algorithm 1 checks whether each block of DB supports the sequence by calling the function *supportBlock* (Algorithm 2). If the sequence is supported, then its support is incremented. The algorithm then returns the relative support of the sequence.

[3] $*$ on dimension D_i can be seen as the root of the taxonomy T_i.

[4] By definition, an h-generalized item is instantiated over all its dimensions. By misnomer, we use the term of item to describe the frequent tuples which are instantiated in a levelwise method in order to mine multidimensional h-generalized items.

Algorithm 2 checks if a sequence ς is supported by a block B. To achieve it, the algorithm combines *recursivity and anchoring operations*. The anchoring operation is used to reduce the space search. This algorithm 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 further (return false). Several possible anchors may be tested if the first ones do not work.

Algorithm 1: SupportCount: Compute the support of a sequence

Data: Sequence ς, database DB, reference dimension set D_R
Result: The support of the sequence ς
begin
  $Integer\ support \longleftarrow 0$;
  $Boolean\ supportedSeq$;
  $\mathscr{B}_{DB,D_R} \longleftarrow \{blocks\ identified\ over\ D_R\}$;
  foreach $B \in \mathscr{B}_{DB,D_R}$ **do**
    $supportedSeq \longleftarrow supportBlock(\varsigma, B)$;
    if *supportedSeq is true* **then**
      $support \longleftarrow support + 1$;
  return $\left(\frac{support}{|B_{DB,D_R}|}\right)$;
end

In order to study the complexity of these algorithms, we adopt the following notations: n_B is the number of tuples in B, $m = |D_A|$ is the number of analysis dimensions, P_{max} is the maximal depth of a taxonomy.

SupportBlock (algorithm 2) The block B is ordered w.r.t. D_T, the anchoring operation is implemented in $O(\log n_B)$. It is sufficient to carry out a dichotomic search to find all the tuples w.r.t. the date condition. Checking if a tuple supports an item takes, in the worst case, $O(P_{max} \times m)$. We should compare the m dimensions of the item to those of the tuple. In the worst case, the complexity is $O(n_B \times P_{max} \times m \times \log n_B)$.

SupportCount (algorithm 1) The previous function is called for each of the l blocks B_i from $\{B_{DB,D_R}\}$. Let $n_{max} = \max n_{B_i}$, the complexity in the worst case is then: $O(l) \times O(n_{max} \times P_{max} \times m \times \log n_{max}) = O(l \times n_{max} \times P_{max} \times m \times \log n_{max})$

3.3 HYPE Against M²SP

Managing hierarchies can be seen as a better way to manage the joker values previously defined in [13]. Indeed, M²SP does not consider hierarchical relations between elements from the analysis dimensions. Then, if there is no possible value instantiation, M²SP uses a joker value (*). This joker value can be seen as the root of a one-depth taxonomy. So, M²SP directly goes from the leaf to the root of the taxonomy (Figure 8.B) by using the joker value.

Thanks to HYPE, more accurate knowledge is mined. Indeed, taxonomies are an alternative when M²SP is not able to instantiate a dimension. We do not directly go from leaf to root. We try to instantiate the dimension with the most specific ancestor of the leaf (Figure 8.A).

Comparison with M²SP

Given a user-defined threshold, taking hierarchies into account (HYPE) makes it possible to mine knowledge which is not mined by M²SP. M²SP:

- $(*, chocolate)$, $(*, pretzel)$, $(*, M1)$, $(*, soda)$, $(*, M2)$,

Algorithm 2: SupportBlock: Checking if a sequence is supported by a block

Data: Sequence ς, block B
Result: Boolean
begin
  `/* initialization */`
  $boolean\ foundItemSet \longleftarrow false$
  $sequence \longleftarrow \varsigma$
  $itemset \longleftarrow sequence.first()$
  $item \longleftarrow itemset.first()$
  `/* Condition to stop the recursivity */`
  if $\varsigma = \emptyset$ **then**
    return (true)
  `/* Scanning the block */`
  while $tuple \longleftarrow B.next \neq \emptyset$ **do**
    if $tuple$ *supports* $item$ **then**
      $followingItem \longleftarrow itemset.second()$
      if $followingItem = \emptyset$ **then**
        $foundItemSet \longleftarrow true$
      `/* Searching for all the items of the itemset */`
      else
        `/* Anchoring w.r.t. item (date) */`
        $B' \longleftarrow \sigma_{date=cell.date}(B)$
        while $tuple' \longleftarrow B'.next() \neq \emptyset \wedge foundItemSet = false$ **do**
          if $tuple'$ *supports* $followingItem$ **then**
            $followingItem \longleftarrow itemset.next()$
            if $followingItem = \emptyset$ **then**
              $foundItemSet \longleftarrow true$
      if $foundItemSet$ *is true* **then**
        `/* Searching for the other itemsets */`
        return $(SupportBlock(sequence.tail(), \sigma_{date>tuple.date}(B)))$
      else
        $itemset \longleftarrow sequence.first()$
        `/* Reducing the search space */`
        $C \longleftarrow \sigma_{date>cell.date}(B)$
  `/* ς is not supported */`
  return (false)
end

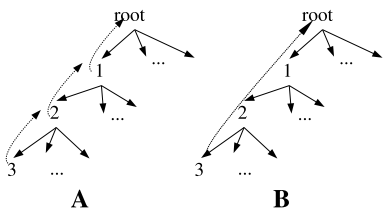


Figure 8: Hierarchy management with HYPE and joker value (*) management with M²SP

$(*, whisky)$

- $\langle\{(*, chocolate)\}\{(*, M1)\}\rangle$, $\langle\{(*, pretzel)\}\{(*, M2)\}\rangle$

HYPE:

- $(Place, chocolate)$,$(EU, pretzel)$, $(Place, M1)$, $(Place,$-$soda)$, $(EU, M2)$, $(Place, Whisky)$, $(EU, Alcoholic\,drinks)$,
- $\langle\{(Place, chocolate)\}\{(Place, M1)\}\rangle$
  $\langle\{(EU, pretzel)\}\{(EU, M2)\}\rangle$
  $\langle\{(EU, Alcoholic\,drinks)\}\{(EU, M2)\}\rangle$
- $\langle\{(EU, Alcoholic\,drinks), (EU, Pretzel)\}\{(EU, M2)\}\rangle$

Taking hierarchies into account makes it possible to mine more finer sequences than in the M^2SP approach.

4. EXPERIMENTS

In this section, we report experiments performed on synthetic data. These experiments aim at showing the relevance of our approach, especially for hierarchy management. The synthetic database contains $5,000$ tuples defined over 5 analysis dimensions. These first experiments compare the number of frequent mined sequences over the depth of the taxonomies (specialisation level) and the user defined threshold. We compared our results to the M^2SP results in order to study the quality of the mined knowledge.

Figure 9: Number of frequent sequences over the depth of the taxonomies (minsup=0.3, $D_A = 5$, average number of sons = 3)

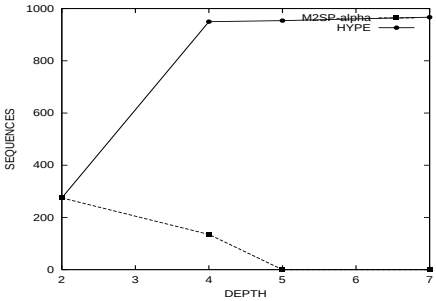
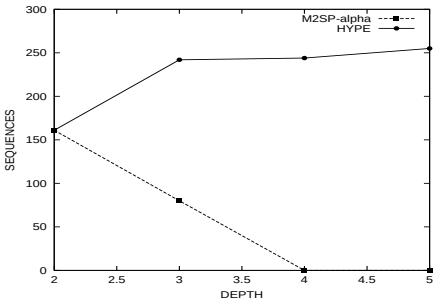


Figure 10: Number of frequent sequences over the depth of the taxonomies (minsup=0.4, $D_A = 5$, average number of sons = 4)



Figures 9 and 10 show the number of frequent mined sequences over the depth of the taxonomies. Increasing the size of the taxonomies generates an additional specialization level (drinks become alcoholic drinks or sodas). When the data becomes more specific, M^2SP mines less frequent sequences until it cannot mine any more knowledge. Taking hierarchies into account provides robustness to deal with the specialization phenomena. Indeed, the sequences are mined among several hierarchy levels.

Figure 11: Number of frequent sequences over the minimal support ($D_A = 5$, average number of sons = 3, highly correlated data)

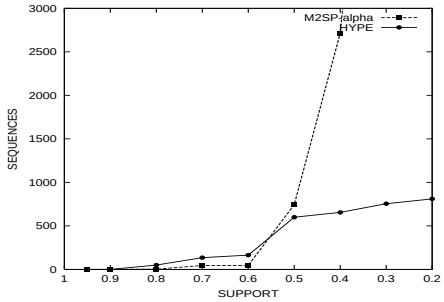


Figure 12: Number of frequent sequences over the minimal support (D_15, average number of sons = 4, depth= 4)

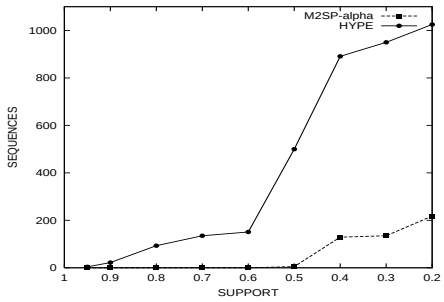


Figure 11 shows the number of frequent mined sequences over the user-defined threshold in a highly correlated database (lower cardinality of analysis dimensions). As soon as the minimal support becomes too low, M^2SP extracts too many frequent sequences. Taking hierarchies into account introduces a powerful subsumption ability which prevents HYPE from mining too many sequences.

Furthermore, in poorly correlated databases, the number of frequent mined sequences is simular to that in highly correlated databases whereas M^2SP mined a very low number of sequences. This highlights the relevance of our approach accordig to the data quality (highly or poorly correlated databases).

5. CONCLUSIONS

In this paper, we have defined multidimensional h-generalized sequential patterns. We take hierarchies into account through taxonomies on analysis dimensions. This makes possible to build multidimensional sequences defined over several hierarchy levels.

We have defined the different concepts (multidimensional h-generalized item, itemset and sequence) and algorithms used to implement our approach. Experiments on synthetic data are reported and highlight the significance of HYPE. These experiments particulary show its ability to subsume knowledge and its strength in dealing with data diversity (density, specialization, etc).

This work offers several perspectives. The efficiency of the extraction could be enhanced by using condensed representations of mined knowledge (closed, free, non-derivable). The use of condensed representations can allow additional pruning and thus enhance the extraction process. Other proprosals can be put forward concerning the hierarchy management. We can imagine modular hierarchy management where some dimensions would not have the same behiavour as other ones in order to meet to the user's needs (prohibition to exceed the hierarchy level λ over the dimension ξ,

. . .). Hierarchy management can allow us to define a novel automatic method to help users to navigate in data cubes.

6. ACKNOWLEDGEMENT

We thank Pr. Dominique Laurent for preliminary discussions on the topic of this paper.

7. REFERENCES

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In *Proc. 1995 Int. Conf. Data Engineering (ICDE'95)*, pages 3–14, 1995.

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

[3] S. de Amo, D. A. Furtado, A. Giacometti, and D. Laurent. An apriori-based approach for first-order temporal pattern mining. In *XIX Simpósio Brasileiro de Bancos de Dados, 18-20 de Outubro, 2004,Brasília, Distrito Federal, Brasil, Anais/Proceedings*, pages 48–62, 2004.

[4] T. Dietterich and R. Michalski. Discovering patterns in sequences of events. *Artificial Intelligence*, 25(2):187–232, 1985.

[5] J. Han. OLAP mining: Integration of olap with data mining. In *DS-7*, pages 3–20, 1997.

[6] J. Han and Y. Fu. Mining multiple-level association rules in large databases. *IEEE Trans. Knowl. Data Eng.*, 11(5):798–804, 1999.

[7] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *SIGMOD Conference*, pages 1–12, 2000.

[8] C.-H. Lee. An entropy-based approach for generating multi-dimensional sequential patterns. In *PKDD*, pages 585–592, 2005.

[9] H. Mannila, H. Toivonen, and A. Verkamo. Discovering frequent episodes in sequences. In *Proc. of Int. Conf. on Knowledge Discovery and Data Mining*, pages 210–215, 1995.

[10] 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.

[11] 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.

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

[13] M. Plantevit, Y. W. Choong, A. Laurent, D. Laurent, and M. Teisseire. M^2SP: Mining sequential patterns among several dimensions. In *PKDD*, pages 205–216, 2005.

[14] M. Plantevit, A. Laurent, and M. Teisseire. $HYPE$: Prise en compte des hiérarchies lors de l'extraction de motifs séquentiels multidimensionnels.(french version). In *EDA*, pages 155–173, 2006.

[15] S. Sahar. Interestingness via what is not interesting. In *KDD*, pages 332–336, 1999.

[16] R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *EDBT*, pages 3–17, 1996.

[17] 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.

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