
Mining Association rules with Multiple Min-supports

Application to Symbolic Data

Tao WAN — Karine ZEITOUNI

PRISM Laboratory, University of Versailles,
45, avenue des Etats-Unis
78035 Versailles Cedex, France
[\[Tao.Wan, Karine.Zeitouni\]@prism.uvsq.fr](mailto:[Tao.Wan, Karine.Zeitouni]@prism.uvsq.fr)

RÉSUMÉ. *Les données symboliques sont un nouveau paradigme de description de données quand les données sont plus complexes que la forme tabulaire. Elles supportent des données plus structurées ayant des variations internes telles que des distributions, des règles telles que des taxonomies. Depuis son introduction [1], le problème d'extraction de règles d'association à partir de grandes bases de données a été le sujet de nombreuses études. Les méthodes connues cherchent les itemsets fréquents satisfaisant un seuil de support. Seulement, le seuil de support uniforme est mal adapté lorsque la fréquence des articles est très disparate. Cet article propose une nouvelle méthode qui vise, d'une part, de fouiller des règles d'association inter-dimensionnelles à partir des données symboliques, et d'autre part, de permettre le choix des différents seuils de support selon leur intérêt dans l'application.*

ABSTRACT. *Conventional data mining techniques apply well to simple data types. However, as more and more complex data are produced, such as spatial or multimedia datasets,, traditional techniques can no longer fulfil their analysis. Symbolic data is a new data description paradigm where the data are more complex than the tabular form. It holds more structured data having internal variations such as value distributions, and rules such as taxonomies. Since its introduction [1], the problem of mining association rules from large databases has been the subject of numerous studies. Popular methods of mining association rules involve finding all frequent itemsets satisfying a uniform support threshold. The uniform min-support approach, however, has some limitations if some important items are unlikely to occur as frequently as the others. This paper proposes a new method that aims to mine inter-dimensional association rules from symbolic data, while allowing specifying different min-supports for certain dimensions depending on their interest within actual application.*

MOTS-CLÉS: *fouille de données, données symboliques, recherche de motifs fréquents.*

KEYWORDS: *data mining, symbolic data, mining frequent itemsets.*

1. Introduction

As a new data description paradigm, symbolic data gives more complex data structures than classical or mono-valued data. It is better adapted to the actual explosively propagating data sources. In this context, we address the problem of association rules mining that is different to that in conventional data analysis. Conventional association rules mining requires finding all frequent itemsets that satisfy a uniform *min-support*. This limitation frequently results in specifying a high *min-support* that will miss interesting but less frequent patterns, as well as reducing *min-support* risks to produce a large result difficult to filter by the user. Mining association rules with multiple min-supports from symbolic data is an efficient algorithm that aims to:

1. Deal with complex data such as symbolic data where no equivalent method has been previously developed.
2. Fulfill, more precisely, the users' requirements by allowing them to specify multiple min-supports within the application.
3. Achieve a good performance due to existing efficient algorithm FP-growth.

This section introduces the context of this method and emphasizes its motivation.

1.1. Symbolic data

According to [12], a symbolic data table is an extension of a relational table where attribute values may contain distributions, intervals or several values linked by taxonomy and logical rules. As input, symbolic data can arise from many source types and aggregate large datasets according to their underlying concepts. Hence, the symbolic data concept allows us to summarize large and complex databases more naturally, with greater richness and a more manageable size. *Fig.1* shows a simple example of a symbolic data table generation procedure from some relational tables describing road accidents in their neighboring environments.

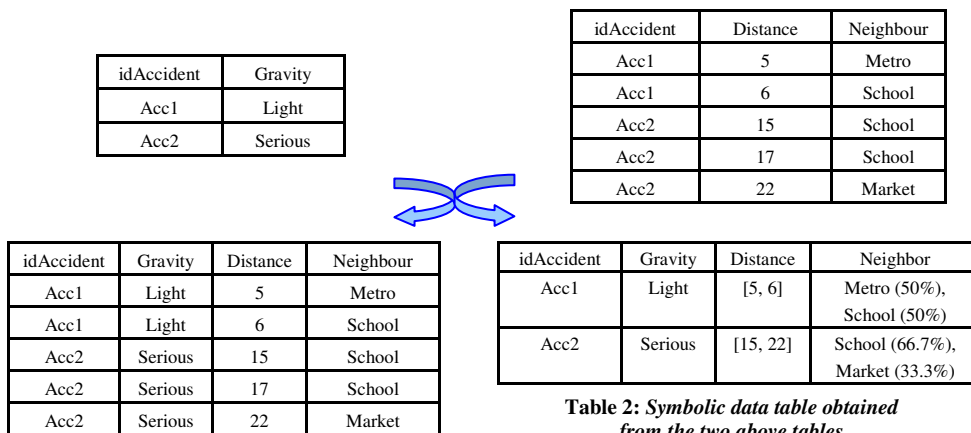


Table 1: Relational table obtained by joining the two above tables

Table 2: Symbolic data table obtained from the two above tables

Fig.1: A relational and symbolic data table arisen from two relational tables

In table 2, we can see that each individual (e.g. Acc1, Acc2) is described by only one summary tuple and attributes can be of interval type or multiple values linked by their distribution. While in table 1, each individual requires many tuples. Therefore, the size of the symbolic table is much smaller. Furthermore, this summary table captures more detailed information than what is obtained by relational aggregates, as it aggregates categorical attributes (such as neighbor) and keeps internal variation and numerical range. This is why the symbolic data paradigm allows us to summarize large and complex databases more naturally and effectively.

"Symbolic Data Analysis" (SDA) [12] is an extension of standard data analysis. It extends many data analysis and data mining techniques towards symbolic data.

1.2. Mining frequent patterns

Since its introduction [1], the problem of mining association rules from large databases has been the subject of numerous studies. Those studies can be broadly divided into three categories, as summarized in [26]:

Scalability: the central question is how to compute the conventional association rules as efficiently as possible. Studies in this category can be further classified into three subgroups: (i) fast algorithms based on the level wise Apriori framework [3,11]; (ii) partitioning [17,19] and sampling; and (iii) incremental updating and parallel algorithms [2,6,8,16].

Extensibility: the central question is how to extend association rules mining. Studies in this category can be further classified into two generations. Studies in the first generation attempt to extend this method to different data types and applications, e.g. mining association rules from time series data [22], web data [23], ...; while studies in the second generation apply larger logic formalism, such as datalog queries in [24].

Functionality: the central question is what kind of rules to compute. Studies in this category can be further classified into two types. Studies in the first type basically considered the data mining exercise in isolation. Examples include multi-level association rules [9], quantitative and multi-dimensional rules [7,14,21], correlations and causal structures [5,20], mining long patterns [4], and ratio rules [12]. Studies in the second type have explored how data mining can best interact with other key components in the broader framework of knowledge discovery. One key component is the Database Management System (DBMS), and some studies explored how association rule mining can handshake with the DBMS most effectively. For instance, the integration of association rule mining with relational DBMS [13], or query flocks [18]. Another component, which is arguably even more important when it comes to knowledge discovery, is the *human user*. Studies of this kind try to provide much better support for (i) user interaction: e.g. dynamically change of parameters midstream; and (ii) user guidance and focus: e.g. limit of the computation to what interests the user.

This study contributes to the field by exploring association rules from complex symbolic data. It is a novel method that has never been treated before. In particular, it belongs to the category of functionality, as it is centered upon the human user's d

objectives by allowing the user to specify different min-supports attributes in order to obtain more interesting information. Additionally, it avoids combinatorial explosion calculations.

1.3. Why mining association rules from symbolic data?

The advantage of symbolic data related to simple data is that they have not much loss of regular (but not frequent) events.

Thus, if we consider the case of a supermarket which is interested by the associations of products bought by its client. Conventional association rules mining allows to analyse the bought products at the same purchase and thus will introduce only the associations rules among these products. So there is a loss of association between the products which are not bought at the same time.

Let's take an example of a customer who buys the bread everyday, while the butter only one time per week. Since the purchase of butter is not very regular, it will not be taken in account at the time of generation of association rules on simple data. However the association between butter and bread is important...

That's why the symbolic data are interesting: they are interested by a set of purchases for each user (to take again the preceding example). We will thus be able to detect an association between the purchase of two products, and this association will be stronger if many customers buy them.

It is necessary nevertheless to pay attention to the period of purchases taken into account. Will one too long period make generate virtual associations and one too short will make miss associations which can be important...

1.4. Why mining association rules with multiple min-supports?

In most research concerning association rules mining, the min-support is fixed. The two following references are exceptions:

1. Mining multi-level association rules [9] is an effective method that can produce interesting information from low level to high level with potentially different *min-supports* between levels, while remaining the same for each level. However, as some combinations of items within a given level could be either very frequent or very rare, this method fails to find interesting rules. Indeed, specifying a high *min-support* will miss interesting but less rules, while reducing *min-support* risks to produce a large result difficult to filter by the user.

2. In [15], the authors argued that using a single *min-support* for the whole database is inadequate because it cannot capture the inherent natures and/or frequency differences of the items in the database. For this reason, they proposed an association rules mining method with *multiple min-supports*. This method, based on "Apriori", extends the association rule model by allowing the user to specify *multiple min-supports* to better reflect the different natures and/or frequencies of

items. By using these specified *multiple min-support*, we can prune certain itemsets in the phase of generating frequent patterns, while removing many rules in the phase of rules generation. In this extended model, the definition of association rules remains the same but the definition of minimum support is changed, each item in the database can have a minimum item support called *MIS* specified by the user. The authors' experiments proved that this model enables us to find rare item rules without producing a large number of meaningless rules with frequent items. However, in the definition of *frequent itemset* of [15], the authors consider that an itemset is frequent if the support of each item in this itemsets satisfies the smallest min-support of all its items. For example, an itemset $L = \{1, 2, 3\}$, $MIS(1) = 10\%$, $MIS(2) = 20\%$, $MIS(3) = 30\%$ and $support(L) = 15\%$ which is greater than $MIS(1)$ even if $support(2) < 20\%$, L is considered as frequent. This definition is not efficient enough since many non-meaningful yet frequent patterns are not removed. Indeed, the smallest MIS can easily be satisfied.

In our work, we borrow the proposed model MIS of [15], but we introduce a new definition for *frequent itemset*. This alternative definition is first explained by way of example 1.

Example 1: Consider an itemset $L = \{1, 2, 3\}$ and $MIS(1) = 10\%$, $MIS(2) = 20\%$, $MIS(3) = 30\%$. L is *frequent* if and only if $support(1) \geq 10\%$, $support(2) \geq 20\%$, $support(3) \geq 30\%$, $support(1, 2, 3) \geq 10\%$, and $support(2, 3) \geq 20\%$.

Definition 1: (*frequent itemset*) an itemset L is frequent if and only if every subset of L satisfies the smallest MIS of its items.

To summarize the difference between our approach and the one of [15], each MIS is directly taken into account in our definition whereas only the smallest MIS is considered in a candidate generation phase in [15]. Hence, this definition allows us to reduce more unnecessary itemsets without losing much interesting information.

According to the definition 1, every subset of an itemset must satisfy a special MIS which is possible no more the same. The classic efficient methods and the proposed method in [15] fail to trait this requirement. Moreover, the symbolic data form is more complex than classic or mono-valued data. In order to overcome these problems, while maintaining a good performance, we have adapted and extended the classic method. One of the mining frequent itemsets methods is the algorithm FP-growth; it consists first in building a compressed database called the Frequent Pattern Tree (FP-tree) [10] that retains the complete frequent items association information. This structure allows us to verify separately the support of each frequent item to generate the frequent itemsets. Since the performance of this algorithm has been proven, the issue of performance is less crucial. The remainder of the paper will introduce our extended method of FP-growth adapted to our presented problems. The paper is organized as follows. Section 2 introduces the method of mining frequent patterns from symbolic data. Section 3 presents our experimental study which demonstrates the interesting results of this new attempt. Section 4 summarizes our study and points out its contribution.

2. Mining association rules from symbolic data

The paper's method extends mining association rules from simple data to symbolic data. While initially only proposed for multi-valued data, section 2.1 expands the method to intervals and distributions. This section details this method and describes its different phases. These include the pre-processing phase, the underlying data structure building phase and the algorithm of mining frequent itemsets phase.

2.1. Conversion from symbolic data to transactional data

Mining frequent patterns in transaction databases has been widely studied in data mining research, and many good methods have been verified, such as *Apriori* [1], *FP-growth* [10], and *TreeProjection* [2]. If the form of symbolic data could be converted to transactional data, we can benefit from the advantages of these traditional methods.

To attain this objective, three *issues need* resolving:

1. **Performance:** the data of every individual are separated by the columns, looking for frequent itemsets one column after another, and then for all their possible combinations. This exponential calculation will inevitably decrease the performance.

Solution: a strategy is used to deal with this problem very easily. It consists of assigning a different code to the values of every attribute and removing their column bars, so that a value of each tuple appears as an item of a transaction and the classic methods become applicable.

2. **Data type:** transaction data is only Boolean data type, but Symbolic data has several types: multi-valued, multi-valued with weights (as distributions), and interval. How to mine frequent itemsets from interval type data?

Solution: interval data are aggregated from quantitative data at the beginning of symbolic producing. They can be converted to qualitative data by discretizing them. This process could be based on predefined concept hierarchies.

3. **The weights of values:** if an attribute is multi-valued type with weights, do we need to consider their weights in mining?

Solution: according to association mining definition, it is the occurrence frequency which is counted for every item support –as argued in [1]. Accordingly, we do not need to care about the weight of values.

With the solutions, we can easily derive transactional form from symbolic data. Example 2 shows this conversion.

Example 2 Table 3 below displays a symbolic data table with 10 individuals and 4 attributes. Furthermore the data type of attribute A is multi-valued with weights.

Table 4 includes the corresponding transactional data type. Each attribute value in Table 3 becomes a transaction item in Table 4, having the column code as a prefix.

# i	A	B	C	D
1	1 (50%), 2 (50%)	1, 2	1, 2, 3	1
2	1(20%), 2 (30%), 3 (50%)	1, 2, 3	1	1, 2
3	1(66%), 4 (34%)	2	2	1, 3
4	1(30%), 3 (70%)	2, 3	1, 3	3
.
.
.
10	1 (60%), 3 (40%)	1, 3	1, 2	1, 2, 3

# i	Items
T1	A1, A2, B1, B2, C1, C2, C3, D1
T2	A1, A2, A3, B1, B2, B3, C1, D1, D2
T3	A1, A4, B2, C2, D1, D3
T4	A1, A3, B2, B3, C1, C3, D3
.	.
.	.
.	.
T10	A1, A3, B1, B3, C1, C2, D1, D2, D3

Table 3: A symbolic data table

Table 4: A table of transformed symbolic data (a set of transactions)

Fig.2: Conversion of symbolic data to transactional type data

2.2. Construction of eXtended Frequent Pattern (XFP-tree)

The phase of conversion reduces the complexity of data types. Now the only difficulty for us is how to verify if an itemset is frequent. As previously indicated, FP-growth has been selected as the base of our method due to the special data structure FP-tree. Hence, we first build an extension of FP-tree called **XFP-tree** (eXtended Frequent Pattern Tree), established by a special relative order. This relative order can afterwards help us to better explore this structure. Further advantages of this special order are outlined in section 2.3.

Let $T = \{T_1, T_2, \dots, T_n\}$ be a **symbolic data table** where $T_i (i \in [1..n])$ is an individual, $Attr = \{A, B, C, \dots, N\}$ be its set of attributes and $I = \{A1, A2, \dots, Am1, B1, B2, \dots, Bm2, \dots, N1, N2, \dots, Nmn\}$ be the **set of all the values (or items)** of all attributes. To simplify the assignment of MIS for each item, we give a **MIS** to each attribute, i.e. $MIS(A) = a, MIS(B) = b, \dots, MIS(N) = n$, so each value of an attribute must have the same MIS. The **support** (or occurrence frequency) of a pattern (a set of items) P is the number of individuals containing P in T . P is a **frequent itemset** if it can satisfy the condition of *definition 1*.

Notice that in FP-tree, the frequent items are sorted in descending order of support. In XFP-tree, only the order is different, where, in our case, it is tailored to the optimization of the mining phase.

Definition 2 (Relative order) A **relative order** in XFP-tree is the arrangement of its nodes. It is built by placing, in ascending order, the frequent single items' MIS. But if two frequent single items have the same MIS, they are ordered by their support descending order.

Before designing the XFP-tree, let's first examine an example.

Example 3 builds an XFP-tree with a set of *MIS* of each attribute $MIS(A) = 40\%$, $MIS(B) = 20\%$, $MIS(C) = 45\%$, $MIS(D) = 30\%$, using the converted transactional table in example 1 (table 4 in **Fig.2**). In this example, for the item set $L = [A1:8, C1:6, B2:5, A3:5, D1:5]$, in which every element satisfies its *MIS*, the relative order will be computed as: $B2 \geq D1 \geq A1 \geq A3 \geq C1$.

After determining this relative order, XFP-tree construction can start following exactly the same procedure as FP-tree construction in [10].

Continuing example 3, to avoid repeatedly scanning the converted symbolic database, like the construction of FP-tree, the values of every individual are listed according to the relative order shown in Table 5.

TID	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
Item list	A1, B2, C1, D2	A2, B1, B2, D1	A1, C1, C2	A1, B2, C1	A1, B2, A3, C1, D1, D2	B2, D1	A1, A2, A3, C1	A1, A2, A3, C1, D1	A1, A3, C1, D1	A1, B2, A3, C2
(ordered) frequent items	B2, A1, C1	B2, D1	A1, C1	B2, A1, C1	B2, D1, A1, A3	B2, D1	A1, A3, C1	D1, A1, A3, C1	D1, A1, A3	A1, A3, C1

Table 5: A transformed symbolic data table as a running example according to relative frequent items order

We may create the root of the tree, labeled with "null". *Scan the converted symbolic database for a second time*. The scan of the values of the first individual T1 leads to the construction of the first branch of the tree: $\langle (B2:1), (A1:1), (C1:1) \rangle$. Notice that the frequent items in the transaction are ordered according to the relative order obtained by *definition 2*. For the second individual T2, since its (ordered) frequent item list $\langle B2, D1 \rangle$ shares a common node $\langle B2 \rangle$ with the existing path $\langle B2, A1, C1 \rangle$, the count of the node B2 is *incremented by 1*, and one new node $\langle D1 \rangle$ is created and linked as the child of $\langle B2 \rangle$. The scan of the third individual T3 leads to the construction of the second branch of the tree, $\langle (A1:1), (C1:1) \rangle$; and so on.

To facilitate tree traversal, an item header table is built in which each item points to its occurrence in the tree via a head of node-link. Nodes with the same item-name are linked in sequence via such node-link. After scanning the values of all individuals, the tree with the associated node-links is shown in **Fig.3**.

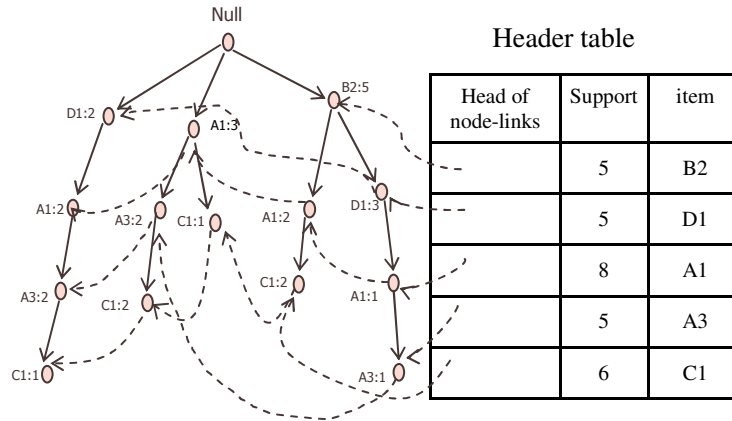


Fig.3: The XFP-tree of example 2

2.3. Mining Frequent Patterns using XFP-tree

XFP-tree construction with a relative order of frequent items makes this structure less compact than the original FP-tree. However, it is highly efficient since one avoids the *complicate combinatorial problem of candidate generation* when relative order is not employed. In this section, we will study the ways to explore the information stored in this XFP-tree structure when the *min-supports* are not uniform, and extend the efficient mining method FP-growth for mining our complete set of frequent patterns.

Example 4 This example examines the mining process based on the constructed XFP-tree shown in **Fig.3**. By using the header table, all the patterns where a node a_i participates are collected, by starting from head a_i and following a_i node-links. Thus, starting from the bottom of the header table we can easily perform the mining process.

For node C1, it derives a frequent pattern (C1:6) and four paths in the XFP-tree: $\langle B2:5, A1:2, C1:2 \rangle$, $\langle A1:3, C1:1 \rangle$, $\langle A1:3, A3:2, C1:2 \rangle$ and $\langle D1:2, A1:2, A3:2, C1:1 \rangle$. The first path indicates that string “ $\langle B2:5, A1:2, C1:2 \rangle$ ” appears twice in the database. Notice, however, that even if $\langle B2 \rangle$ itself appears five times, it only appears twice together with $\langle A1, C1 \rangle$. Thus to study, only C1 prefix path $\langle B2:2, A1:2, C1:2 \rangle$ counts. Similarly, the second path indicates string “ $\langle A1, C1 \rangle$ ” appears once in the set of Symbolic database, the third path indicates string “ $\langle A1, A3, C1 \rangle$ ” appears twice and the fourth path indicates string “ $\langle D1, A1, A3, C1 \rangle$ ” appears also once. These four prefix paths of C1, “ $\{(B2:2, A1:2), (A1:1), (A1:1, A3:2), (D1:1, A1:1, A3:1)\}$ ”, form C1 sub-pattern base, which is called **C1 extended conditional pattern base** (i.e. the sub-pattern base under the condition of C1 existence). Construction of an FP-tree on this conditional pattern base (called **C1 extended**

conditional XFP-tree) leads to a recursive mining of the XFP-tree base $\langle\{(B2:2, A1:2), (A1:1), (A1:2, A3:2), (D1:1, A1:1, A3:1)\}|C1\rangle$.

This conditional XFP-tree construction can be designed based on the following Lemma:

Lemma 1(eXtended fragment growth) Let α be an itemset in XFP-tree, B be a conditional pattern base, and β be an item in B . If α is frequent and the support of β satisfies β *MIS*, the itemset $\langle \alpha, \beta \rangle$ is also frequent.

Rationale. According to the definition of conditional pattern base and XFP-tree, each subset in B occurs under the condition of the occurrence of α in the original converted symbolic database. If an item β appears in B n times, it appears with α in Symbolic database n times as well. As the definition of XFP-tree, β has a *MIS* smaller than that of α . If β support satisfies his *MIS*, $\alpha \cup \beta$ also satisfies this *MIS*. As α has already been frequent, the itemset $\langle \alpha, \beta \rangle$ is therefore also frequent.

From this Lemma, we can easily derive an important corollary.

Corollary 1 (eXtended pattern growth) Let α be an item in Symbolic database, B be a conditional pattern base, and β be an item in B . Then $\alpha \cup \beta$ is frequent in Symbolic database if and only if β is frequent in B .

Rationale. This lemma is the case when α is a frequent item but not an itemset. We verify nodes frequency from the nodes children to the nodes father. Since we have established the XFP-tree structure by the ascending order of the node *MIS*, a node father appears frequently together with its nodes children if it is frequent in the extended conditional pattern base of its frequent children.

Fig.4 shows mining $C1$ conditional pattern base “mine $\langle\{(B2:2, A1:2), (A1:1), (A1:2, A3:2), (D1:1, A1:1, A3:1)\}|C1\rangle$ ” involving the first mining 2-itemsets. The two items of the first 2-itemset $\langle A3, C1 \rangle$ appear together three times, $C1$ appears six times, satisfying his threshold, but this itemset is not frequent because $A3$ *MIS* is 4. The second 2-itemset $\langle A1, C1 \rangle$ appears six times, so it is frequent. Like the reason of the first 2-itemset, $\langle D1, C1 \rangle$ is not either frequent. The two items of the fourth 2-itemset $\langle B2, C1 \rangle$ appear together two times, satisfying the $B2$'s *MIS*. Since $C1$ has satisfied its *MIS*, this 2-itemset is frequent. Notice $C1$ has a threshold bigger than that of $B2$, according to Corollary 1, $\langle B2, C1 \rangle$ is frequent because $B2$ has satisfied its *MIS* and $C1$'s frequency been previously verified. Notice also a set of items is not frequent if one of its subsets is not frequent. So we only need to verify the frequency of 3-itemset $\langle B2, A1, C1 \rangle$ as $\langle A3, C1 \rangle$ and $\langle D1, C1 \rangle$ are not frequent. Since $\langle B2, A1, C1 \rangle$ appears twice, satisfying $B1$'s *MIS*, according to Lemma 1, this 3-itemset is frequent. Since $\langle A1, C1 \rangle$ and $\langle B2, C1 \rangle$ are included in $\langle B2, A1, C1 \rangle$, $C1$'s conditional XFP-tree is $\langle B2, A1, C1 \rangle$.

The mining conditional XFP-tree $\langle B2, A1 \rangle|C1$ is exactly the same as the conditional FP-tree mining in [10], so we do not introduce it here. The result of this mining is $\{\{B2, C1\}, \{A1, C1\}, \{B2, A1, C1\}\}$.

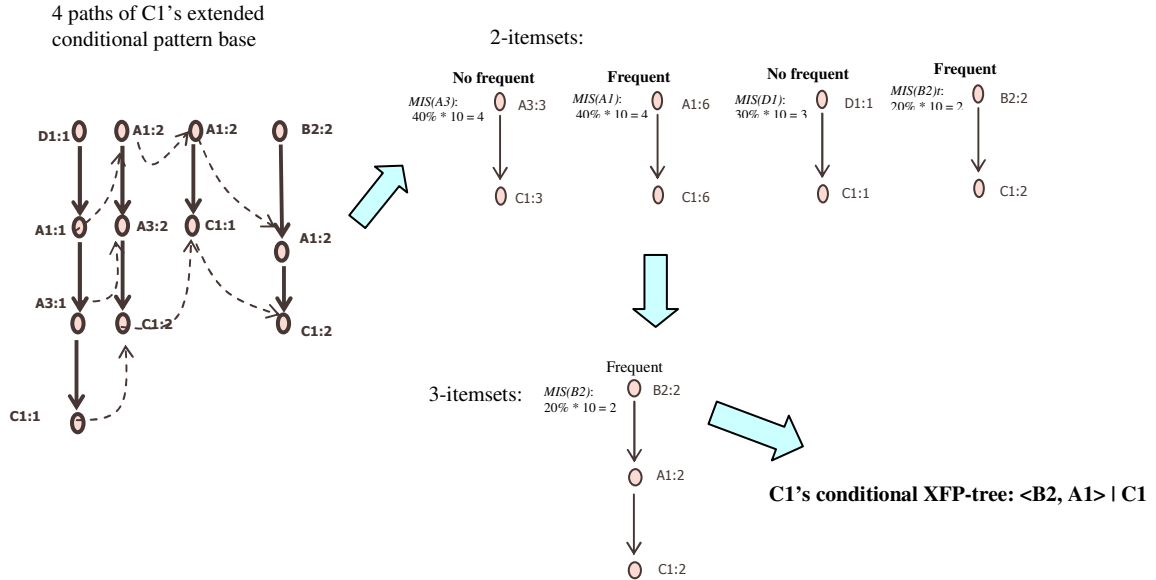


Fig.4: A conditional XFP-tree built from XFP-base for a node C1

Similarly to the generation of itemsets of C1, we obtain a table of mining all the frequent patterns of Example 3 in table 6.

Item	Conditional pattern base	Conditional XFP-tree	Generated Itemsets
C1	<B2, A1>:2, <A1>:1, <A1, A3>:2 <D1, A1, A3>:1	{B2, A1} C1	{B2, C1}, {A1, C1} {B2, A1, C1}
A3	<B2, D1, A1>:1, <A1>:2, <D1, A1, A3>:2	{D1, A1} A3	{D1, A3}, {A1, A3}, {D1, A1, A3}
A1	<B2>: 2, <D1>: 2	{B2} A1	{B2, A1}
D1	<B2>:3	{B2} D1	{B2, D1}

Table 6: Mining all-patterns by creating conditional XFP-tree

Based on Corollary 1 and Lemma 1, mining can be easily performed like FP-growth in [10]. But this advantage is based on the relative order. Without this relative order at the beginning of XFP-tree construction, the tree structure would be inevitably more difficult. Let's see an example.

Example 5 If we construct a FP-tree according the method of construction in [10] where the frequent items are sorted in their descending order frequency as shown in

Fig.5 and the information pertaining to the supports and *MIS*s of each node is located in the right header table.

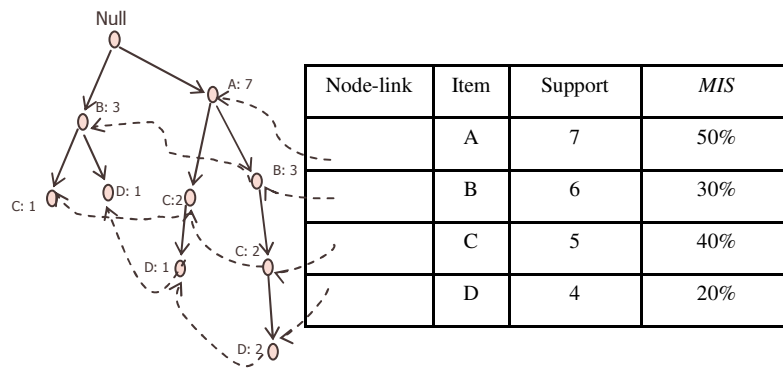


Fig.5: the FP-tree of example 4

In the conditional pattern base $\{ \langle A:2, B:2, C:2 \rangle, \langle A:1, C:1 \rangle, \langle B:1 \rangle \}$, we analyze first the frequency of all the 2-itemsets. $\langle C, D \rangle$ appears three times, satisfying D's *MIS*. As C has a *MIS* bigger than that of D, we must also verify if C's support is greater than its *MIS*. Finally we find that $\langle C, D \rangle, \langle B, D \rangle, \langle A, D \rangle$ are frequent 2-itemsets. The next step involves verifying the frequency of the 3-itemsets. For 3-itemset $\langle B, C, D \rangle, \langle C, D \rangle$ appears three times, $\langle B, C, D \rangle$ twice, satisfying D's *MIS*. However, the frequency of $\langle B, C \rangle$ must be found in another conditional pattern base. Due to the lengthy verification procedure, this renders the process slow and complicated. Finally $\langle B, C, D \rangle$ is not frequent, as the $\langle B, C \rangle$ frequency is 3 that does not satisfy C's *MIS*. The three items of the second 3-itemset $\langle A, C, D \rangle$ appear together 3 times, satisfying D's *MIS*, and the frequency of $\langle A, C \rangle$ is four, equally satisfying the *MIS* of C, so $\langle A, C, D \rangle$ is frequent. And so on.

From example 5 we can easily see that the process of mining conditional pattern bases is more complicated when the frequent items are not sorted by the relative order. Since every frequent item no longer has the same *MIS*, if a node child has a *MIS* smaller than its ancestor, we must verify not only its frequency with every ancestor, but also the frequency of every combination of all ancestors. By sorting these frequent items by their *relative order* first, this becomes very useful in mining the XFP-tree, as proved in Lemma 1.

3. Experiments study

The section evaluates the extended model XFP-tree and the algorithm XFP-growth. The performance and advantages of this model will be justified by the comparison of XFP-growth with the classical frequent pattern mining algorithm and the proposed approach in [15] "*MsApriori*".

All the experiments are performed on a 2.4GHz Pentium PC machine with 256 DDR megabytes main memory, running on Microsoft Windows/XP. All the programs are written in Microsoft/Visual C++6.0. We implement other algorithms to the best of our knowledge based on the published reports on the same machine and compare in the same running environment.

For comparing these methods, we need to assign MIS values to items in the datasets. Here, we use the method of MIS values distribution in [15] for each item, the formulas is following:

$$\text{MIS}(i) = \begin{cases} \text{M}(i) & \text{M}(i) > LS \\ LS & \text{Otherwise} \end{cases}$$

$$\text{M}(i) = \text{BF}(i)$$

$F(i)$ is the frequency of the actual items. LS is the lowest minimum item support that is allowed by users. B ($0 \leq B \leq 1$) is a controlling parameter of MIS which is relative to its frequency. If $B = 0$, we have only one min-support, LS , which is the same to the conventional association rules mining. Here, for illustration purposes, we show the results from one dataset. The other datasets are similar and are thus omitted. This dataset is generated with 1K items, 100,000 transactions, the average transaction size average maximal potentially frequent itemsets size are set to 15 and 10. In this dataset, the deviation of item frequencies in the data is 1.58% (expressed in the percentage of the total data size). Like in [15], we use also three very low LS values, 0.1%, 0.2%, and 0.3% respectively. For the value B , we let $B = 1/\alpha$ and α varies from 1 to 15.

From section 1.4, we know that our approach and the *MsApriori* allow together the user to specify multiple min-supports in order to find rare item rules yet without producing a huge number of meaningless rules with frequent items. However, our method gives another definition of frequent itemsets which directly takes into account each MIS in every candidate itemset. **Fig 6** shows the number of large itemsets found. The thick lines give the numbers of large itemsets found by existing approach with a uniform min-support at 0.1%, 0.2%, and 0.3% respectively.

From **Fig 6** We can see that the number of large itemsets is significantly reduced by the method *MsApriori* and it reduced moreover by our approach when α is not too large. When α becomes larger, the results of the three method get closer because α becomes larger and reaches LS .

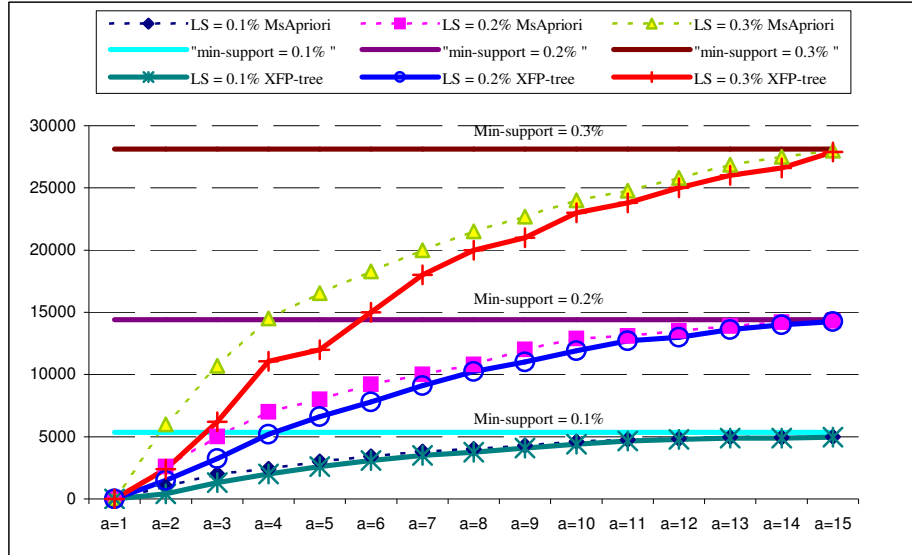


Fig.6: Number of large itemsets found

Since our method checks the MIS of all the subsets of each itemset to verify if this itemset is frequent, the procedure is more complicated than *MsApriori*. However, our method is based on FP-growth where we don't need to generate candidate itemsets, so the issue performance is less crucial. Fig7 shows the execution time comparison of XFP-tree and *MsApriori*.

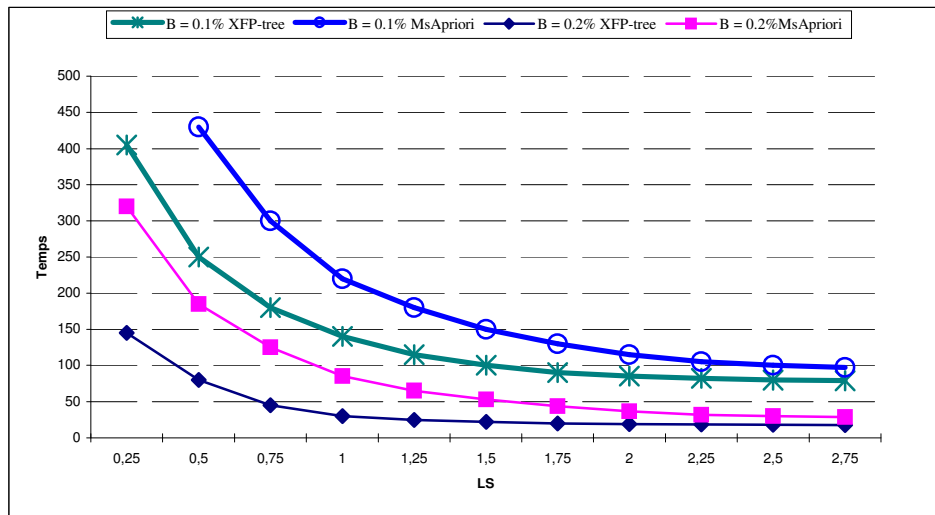


Fig.7: Scalability with LS

In Fig 7, B is set to 0.1% and 0.2%, while LS varies from 0 to 3. We can see that the execution time of our method is significantly faster than that of *MsApriori* when LS is small and B is fixed.

4. Conclusion

The paper contributes to the extant literature by extending functionally association rules mining. Specifically, our paper extends and contributes to the research field in three main ways; i) by dealing with complex data, such as symbolic data, where no equivalent method has been previously developed; ii) by more accurately fulfilling the user's requirements, by allowing the specification of different *MISs* depending on the frequency/nature of data; and finally iii), to gain a good performance by being based on existing efficient algorithm FP-growth. Concerning symbolic data analysis, a conversion phase to transaction data has also been proposed. This method is not only tailored to symbolic tables but also applies to tabular data when some important items are unlikely to occur as frequently as others. The proposed method allows specifying different *min-supports* for certain dimensions (i.e. attributes) depending on their interest within the actual application. With different *min-supports* according to the users' requirements, we may obtain interesting information that we could not find by using the classical methods. The novel definition of *frequent itemsets* reduces the frequent patterns numbers without losing important information. By re-using the tree structure, FP-tree, we can separately verify every subset support of an itemset, resulting in the attainment of a good performance when the data is sufficiently dense.

Its principle is similar to that of the FP-growth algorithm. It first extends the FP-tree data structure to XFP-tree by changing the tree nodes order. Thus, the mining process could reuse the FP-growth algorithm with minimal changes.

This algorithm has been implemented, and the experimental results have been obtained, in the context of road traffic safety analysis, which is a spatial data mining application. It demonstrates the **interest of** our method and a comparison of our method to that employed in [15].

Reference

- [1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in massive databases. In *Proc. 1993 SIGMOD*, pp. 207-216.
- [2] R. C. Agarwal, C. Aggarwal and V. V. V. Prasad. A tree projection algorithm for generation of frequent itemset. In *J. of Parallel and Distributed Computing*, 61(3), pp350-371, 2001.
- [3] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. 1994 VLDB*, pp. 487-499.
- [4] R.J. Bayardo. Efficiently mining long patterns from databases. In *Proc. 1998 SIGMOD*, pp. 85-93.
- [5] S. Brin, R. Motwani, and C. Silverstein. Beyond market basket: Generalizing association rules to correlations. In *Proc. 1997 SIGMOD*, pp. 265-276.
- [6] D.W. Cheung, J.Han, V.T.Ng, and C.Y. Wong. Maintenance of discovered association rules in large databases: An incremental updating technique. In *Proc. 1996 ICDE*, pp. 106-114.
- [7] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. In *Proc. 1996 SIGMOD*, pp.13-23.

- [8] E.-H. Han, G. Karypis, and V. Kumar. Scalable parallel data mining for association rules. In *Proc. 1997 SIGMOD*, pp. 277-288.
- [9] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *Proc. 1995 VLDB*, pp. 420-431.
- [10] J. Han, J. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation. In *Proc. SIGMOD, 2000*.
- [11] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A.I. Verkamo. Finding interesting rules from large sets of discovered association rules. In *Proc. 1994 CIKM*, pp. 401-408.
- [12] H.-H. Bock and E. Diday. *Analysis of Symbolic Data - Exploratory methods for extracting statistical information from complex data*. Springer Verlag, Heidelberg, 425 pages, ISBN 3-540-66619-2, 2000.
- [13] S. Sarawagi, S. Thomas, and R. Agrawal. Integrating association rule mining with relational data base systems; Alternatives and implications. In *Proc. 1998 SIGMOD*, pp. 343-354.
- [14] R.J. Miller and Y. Yang. Association rules over interval data. In *Proc. 1997 SIGMOD*, pp. 13-24.
- [15] B. Liu, W. Hsu and Y. Ma, Mining association rules with multiple minimum supports. In *Proc ACM Intl. Conference KDD, 1999*.
- [16] J.S. Park, M.-S. Lakshmanan, J. Han, and A. Pang. Exploratory mining and pruning optimizations of constrained associations rules. In *Proc. 1995 CIKM*, pp. 31-36.
- [17] J.S. Park, M.-S. Chen, and P.S. Yu. Using a hash-based method with transaction trimming for mining association rules. *IEEE TKDE*, 9(5), pp. 813-825, Sept./Oct. 1997.
- [18] D. Tsur, J. D. Ullman, S. Abiteboul, C. Clifton, T. Motwani, S. Nestorov, and A. Tosenthal. Query flocks: A generalization of association-rule mining. In *Proc. 1998 SIGMOD*, pp. 1-12.
- [19] A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in large databases. In *proc. 1995 VLDB*, pp. 432-443.
- [20] C. Silverstein, S. Brin, P. Motwani, and J. Ullman. Scalable techniques for mining causal structures. In *Proc. 1998 VLDB*, pp. 594-602.
- [21] H. Toivonen. Sampling large databases for association rules. In *Proc. 1996 VLDB*, pp. 134-145.
- [22] J. Yang, W. Wang and Philip S. Yu. Mining asynchronous periodic patterns in time series data. In *Proc. 2000 KDD*, pp. 275-279.
- [23] José Borges and Mark Levene. Mining association rules in hypertext databases. In *Proc, 1998, KDD*, pages 149-153. New York, USA, August 1998.
- [24] L. Dehaspe, H. Toivonen, and R. D. King. Finding frequent substructures in chemical compounds, In *Proc, 1998, KDD*, pages 30-36. New York.
- [25] Mannila, H. "Database methods for data mining." KDD-98, tutorial, 1998.
- [26] L. V. S. Lakshmanan, C. K.-S. Leung, R. T. Ng. The Segment Support Map: Scalable Mining of Frequent Itemsets. In *Proc. 2000 SIGKDD*, pp. 21-27.