

# Tackling Closed Pattern Relevancy In $n$ -ary Relations

Jérémy Besson<sup>1</sup>, Loïc Cerf<sup>2</sup>,  
Rémi Thévenoux<sup>2</sup>, and Jean-François Boulicaut<sup>2</sup>

<sup>1</sup> Institute of Mathematics and Informatics, MII, Vilnius, Lithuania

<sup>2</sup> INSA-Lyon, LIRIS CNRS UMR5205, F-69621 Villeurbanne cedex, France  
{Firstname.Name}@insa-lyon.fr

**Abstract.** For the last decade, set pattern discovery from binary relations has been studied in depth. Today, many complete and efficient algorithms for frequent closed set mining are available. More recently, their extensions towards  $n$ -ary relation mining have been considered. In this paper, we consider the recent proposal for closed  $n$ -set pattern discovery and we discuss their relevancy. Indeed, starting with experiments on two real-life multidimensional data sets, we discuss the quality of the extracted local patterns thanks to an inside and outside perspective on the discovered closed  $n$ -sets. This original analysis enables to support the declarative specification of a priori relevant patterns thanks to the conjunction of primitive constraints (minimal size, minimal area,  $\delta$ -isolated and fault-tolerance constraints) they have to satisfy. Interestingly, some of these primitive constraints can be exploited within available solvers.

## 1 Introduction

For the last decade, set pattern discovery from binary relations has been extensively studied. Many complete and efficient algorithms are now available. This is in particular the case for (frequent) closed set mining (see, e.g., [13, 22, 14, 18]). Closed set mining has been proved useful across the many applications of frequent sets, e.g., association rule discovery, associative classification, pattern-based clustering, etc. Following a perspective that is now classical, constraint-based mining of closed patterns (or formal concepts) has been studied to improve both the efficiency and the relevancy of the extracted patterns (see, e.g., [17, 2, 1, 3]). Indeed, constraints (e.g., minimal size or typing constraints) are one obvious way of tackling some interestingness issues and avoiding the computation of thousands of uninteresting patterns.

Examples of binary relations that have been considered are the popular *transaction*  $\times$  *item* cases. More generally, it concerns the many application domains where Boolean properties can be recorded for a given set of objects, i.e., *object*  $\times$  *property* data sets. Recently, generalizations of these algorithms to  $n$ -dimensional settings ( $n \geq 3$ ) have been proposed. Handling  $n$ -ary relations looks promising. For instance, we would like to consider an *object*  $\times$  *property*  $\times$  *time*

ternary relation or a *customer*  $\times$  *product*  $\times$  *region*  $\times$  *time* 4-ary relation. This has given rise to a few proposals [11, 5, 10, 9].

In this paper, we address some issues raised by these new closed patterns. The whole discussion is valid in the binary case but it becomes crucial in the more difficult setting of  $n$ -ary relation mining. Indeed, all the difficulties we already know in the binary case (e.g., lack of fault-tolerance, explosion of the number of patterns, too many small patterns that appear to be false positive observations) are dramatically aggravated when  $n \geq 3$ . Starting from experiments with two real-life multidimensional data sets, we consider the quality of the extracted patterns thanks to an inside and outside perspective on the computed closed  $n$ -sets. In other words, we analyze the extracted patterns considering first the elements that are inside the patterns and, afterwards, the elements that do not belong to the patterns. To the best of our knowledge this is a rather original analysis of pattern quality. This is however a fairly natural idea when considering D. Hand’s perspective on local pattern discovery: “A local pattern is a data vector serving to describe an anomalously high local density of data points” [8]. Therefore, we will consider both the density issues inside a pattern but also outside vs. inside in order to provide a semantics to “anomalously”.

Constraint-based mining is a successful framework for supporting pattern discovery tasks. It leads to the declarative specification of the constraints that have to be satisfied by the extracted patterns. Such a specification has to combine primitive constraints that are more or less well-identified and studied. In this paper, we consider conjunctions of primitive constraints. Some of them like minimal size or minimal area constraints are quite popular. Some others like fault-tolerance or the  $\delta$ -isolated constraints are not really well understood. Our empirical study confirms that minimal size, area or volume constraints remain important in the general case of  $n$ -ary relations (inside view). We also show that extracted patterns can be split into different classes called pattern types w.r.t. their outside “slices” (outside view). Not only the use of new constraints is discussed but also we consider briefly their efficient processing within available solvers. Notice that the efficient processing of constraints (i.e., “pushing” them into the extraction phase) is much harder when considering  $n$ -ary relation mining with  $n \geq 3$  rather than the binary one [5].

The rest of the paper is organized as follows. Section 2 provides the definition of closed  $n$ -sets in  $n$ -ary relations as well as a state-of-the-art of related algorithms. In Section 3, we analyze closed  $n$ -sets extracted from two real data sets following both an inside and an outside view. Section 4 discusses how such an empirical study can help in extracting more relevant patterns. Section 5 briefly concludes.

## 2 Closed patterns in $n$ -ary relations

### 2.1 Preliminary definitions

Let  $A^1, \dots, A^n$  be  $n$  categorical attributes and assume their domains are respectively  $D^1, \dots, D^n$ .  $\mathcal{R}$  is a  $n$ -ary relation on these attributes, i.e,  $\mathcal{R} \subseteq D^1 \times \dots \times$

$D^n$ .  $n$ -sets are elements of  $2^{D^1} \times \dots \times 2^{D^n}$ . We use the  $\sharp$  operator to denote set cardinality.

Intuitively, a  $n$ -set  $H = \langle X^1, \dots, X^n \rangle$  s.t.  $\forall i = 1 \dots n, X^i \subseteq D^i$  is a closed  $n$ -set iff (a) all elements of each set  $X^i$  are in relation with all the other elements of the other sets  $X^{j \neq i}$  in  $\mathcal{R}$  and (b)  $X^i$  sets cannot be enlarged without violating (a). Formally,  $H$  is a closed  $n$ -set iff it satisfies the conjunction of the following primitive constraints.










**Definition 1.** ( $\mathcal{C}_{connected}$ )  $H = \langle X^1, \dots, X^n \rangle$  satisfies  $\mathcal{C}_{connected}$  in  $\mathcal{R}$  iff

$$X^1 \times \dots \times X^n \subseteq \mathcal{R}$$

**Definition 2.** ( $\mathcal{C}_{closed}$ )  $H = \langle X^1, \dots, X^n \rangle$  satisfies  $\mathcal{C}_{closed}$  in  $\mathcal{R}$  iff

$\forall i = 1 \dots n, \forall x^i \in D^i \setminus X^i, \langle X^1, \dots, X^i \cup \{x^i\}, \dots, X^n \rangle$  does not satisfy  $\mathcal{C}_{connected}$

$\mathcal{C}_{connected}$  imposes patterns to be a sub-set of  $\mathcal{R}$ , i.e., in their Boolean representation they only cover "1" values.  $\mathcal{C}_{closed}$  forces the extracted patterns to be maximal w.r.t. every attribute. In a binary relation, a closed 2-set is a 2-set  $\langle X^1, X^2 \rangle$  satisfying  $\mathcal{C}_{connected} \wedge \mathcal{C}_{closed}$ . This is known as a *formal concept* according to the terminology introduced by R. Wille [20]. These closed 2-sets are associated closed sets from each attribute, e.g., a closed itemset and its supporting closed set of objects/transactions. Closed  $n$ -sets appear to be a straightforward generalization of formal concepts to  $n$ -ary relations when  $n > 2$ .

	A	B	C
1			
2			
3			

**Fig. 1.** A visual representation of  $\mathcal{R}_E \subseteq \{1, 2, 3\} \times \{A, B, C\} \times \{star, circle, square\}$ .

*Example 1.* Figure 1 provides a ternary relation  $\mathcal{R}_E \subseteq \{1, 2, 3\} \times \{A, B, C\} \times \{\star, \circ, \square\}$ . This may represent customers (1, 2, and 3) buying items (A, B and C) along three months ( $\star$ ,  $\circ$  and  $\square$ ). The 3-sets  $\langle (1, 3), (A, B, C), (\star) \rangle$  and  $\langle (1, 2, 3), (B), (\circ, \square) \rangle$  are examples of closed 3-sets in  $\mathcal{R}_E$ .  $\langle (1, 3), (A, B, C), (\star) \rangle$  shows that Customers 1 and 3 buy Items A, B and C during Month " $\star$ " ( $\mathcal{C}_{connected}$ ). Moreover, it is closed w.r.t. every attribute ( $\mathcal{C}_{closed}$ ):

- There is no other month for which they buy these three items.

- No other customer buys these three items during this month.
- No other item is simultaneously bought by these customers during this month.

The 3-set  $\langle (1, 3), (A, B, C), (\star, \square) \rangle$  violates  $\mathcal{C}_{connected}$  because  $(1, A, \square) \notin \mathcal{R}_E$  or  $(3, A, \square) \notin \mathcal{R}_E$ . The 3-set  $\langle (2), (B, C), (\circ) \rangle$  satisfies  $\mathcal{C}_{connected}$  but not  $\mathcal{C}_{closed}$  because  $(2, A, \circ) \in \mathcal{R}_E$ .

Notice that when we discuss about the pattern types, we consider that the  $n$ -ary relation is encoded as a collection of Boolean values that tell whether it contains a given tuple or not. This way,  $\mathcal{R}_E$  appears to be a cube (3-dimensional structure) of 0/1 values.

## 2.2 Computing closed patterns in $n$ -ary relations

We do not discuss further the many proposals in the binary case (see, e.g., [13, 22, 14, 18, 17, 2]). Recently, the two algorithms CUBEMINER [10] and TRIAS [9] have been proposed for closed pattern mining from 3-ary relations.

[10] proposes in fact two algorithms that compute closed 3-sets from ternary relations. The first one, called *Representative slice mining*, consists in enumerating all subsets of the smallest attribute domain. For each of them, the related binary relation is projected on the two other attributes using bitwise AND operations between elements. Then, closed 2-sets are extracted from these new binary relations. After adding elements of the enumerated attribute, a post-processing phase removes the 3-sets that are not closed. A second algorithm, called CUBEMINER, directly operates on the ternary relation. It consists in using cubes denoted  $X \times Y \times Z$  called *cutters* presenting the following particularity: none of their tuples are in relation. In other terms, it generalizes the closedness checking introduced in [2] for closed 2-set mining. [10] performs a depth-first ternary enumeration, i.e., each candidate is split into three new candidates: a first one without the elements of  $X$ , a second one without the elements of  $Y$  and a third one without the elements of  $Z$ . The main limitations of this algorithm are that (a) for each candidate, several checks are required to ensure its closedness and, (b) each candidate can be generated several times.

TRIAS [9] also extracts closed 3-sets from 3-ary relations. It basically relies on closed 2-set extraction from two binary relations. From a relation on  $D^1 \times D^2 \times D^3$ , TRIAS first constructs a new binary relation on  $D^1 \times (D^2 \times D^3)$  whose columns correspond to couples of elements of  $D^2$  and  $D^3$ . Every closed 2-sets  $\langle A, B \rangle$ , extracted from this new binary relation, is such that  $B$  contains couples of  $D^2 \times D^3$  in relation with each of the elements of  $A \subseteq D^1$ . The set  $B$  stands for a relation which is not fully connected (i.e., there are "0" values in its Boolean representation). Thus, in a second step, TRIAS extracts every closed 2-sets from the relation generated from  $B$  and checks its closedness w.r.t.  $D^1$ . This verification is easily carried out: its closure must be  $A$ . Basically, this algorithm only works when  $D^1$  is small otherwise too many patterns that are closed on  $D^1 \times (D^2 \times D^3)$  but not on  $D^1 \times D^2 \times D^3$  are generated.

To the best of our knowledge, only the DATA-PEELER algorithm is able to extract closed patterns in  $n$ -ary relations under constraints when  $n \geq 3$  [5]. It performs a binary depth-first enumeration where the  $n$ -ary search-space is split into two disjointed partitions of the original search-space. The different attributes are considered in a symmetric way, i.e., any attribute can be enumerated. Again, the principle of [2] is used to check the closedness of patterns. In practice, DATA-PEELER outperforms the two other algorithms by orders of magnitude in terms of efficiency for 3-ary relations.

### 3 Empirical study of closed pattern relevancy

#### 3.1 Data sets

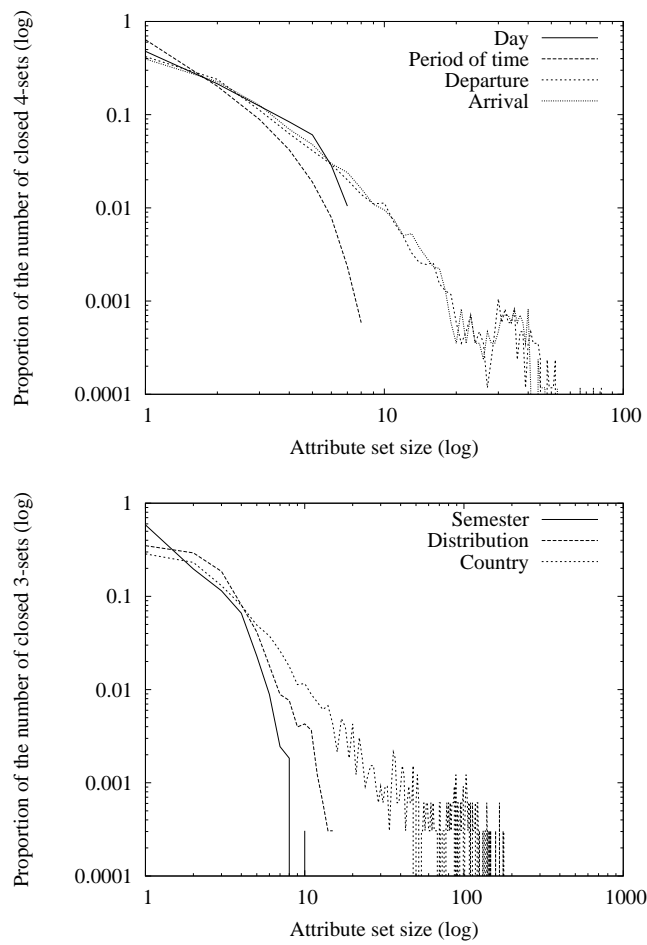
We have been considering two real-life multidimensional data sets. The first one is derived from the logs of the `DistroWatch.com` website. This popular web site gathers comprehensive information about GNU/Linux, BSD and Solaris distributions. Every distribution being described on a separate page, a visitor loading it is considered “interested” in the distribution. Every IP address contacting the server is analyzed so that the country the connection comes from is stored as well. Finally, time stamps enable to study the evolution of the interest granted to the different distributions along time. Data have been normalized so that every country and every time period has the same importance. They have been transformed in 0/1 data in the following way: for each distribution, we have kept the elements of  $\mathcal{R}$  containing this distribution and such that its normalized interest exceeds a threshold equals to one quarter of the maximal normalized interest for this distribution in  $\mathcal{R}$ . The final 3-ary data set gathers 10 semesters, 242 countries and 294 distributions. The density of this data set (ratio between the cardinality of the relation and the product of the cardinality of all attribute domain) is 0.5%.

The second data set contains information about the dynamics of a public bicycle network. Typical information at our disposal is which days of the week and at what time bicycles were rented to go from a given place to another one. This data set is in the form of a 4-ary relation with the following attributes: day of the week (7 elements), period of the day (8 elements - periods of 3 hours), departure place (245 elements) and arrival place (245 elements). For confidentiality reasons, we cannot provide more details about these data. The density of this data set is 0.35%.

#### 3.2 Inside view of patterns

Closed patterns in  $n$ -ary relations as for itemsets and formal concepts in binary relations, can depict different types of associations between sets of elements (e.g., one type of pattern concerns sets of elements of small size associated to sets of large size). During a local pattern mining task, the minimal size constraint enforces that the size of a given attribute set is larger than a threshold. It is the

multidimensional counterpart of the popular minimal frequency constraints in the binary case. Using this constraint, only the patterns that are large enough – according to one or several attributes – are extracted. If we do not want to consider patterns describing associations between small sets of attributes, then the interpretation is easier. In addition to this, we may also consider a minimal volume constraint forcing the patterns to cover at least a given number of tuples in the original data set. Such a constraint may be more relevant than a conjunction of minimal size constraints on the different attributes.

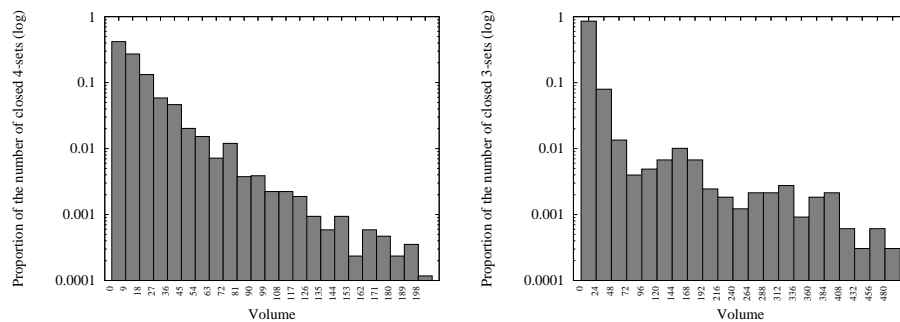


**Fig. 2.** Proportion of closed  $n$ -set having a given number of elements in one attribute (public bicycle network data at the top and DistroWatch.com data at the bottom).

Closed local pattern mining in  $n$ -ary relations has been recently studied and an empirical study of the impact on such constraints in real cases is still missing. This is a good opportunity to grasp the difficulty at considering such data sets and to see what are the shapes (size and volume) of closed  $n$ -sets that hold in them. On the two data sets, we have been using our implementation of DATA-PEELER [5] to compute all (unconstrained) closed  $n$ -sets. Figure 2 presents for each attribute the number of patterns having a given size (for this attribute). Unsurprisingly, most of them have a small number of elements per attribute (left part of the curves). These patterns are commonly considered as a priori uninteresting for numerous real-life extraction tasks. This confirms how useful a minimal size constraint is.

More interestingly, there are patterns with large sets of attributes that seem to emerge from these distributions. Indeed, some large sets of attributes are over-represented w.r.t. the global trends of the distributions, patterns that can be qualified as unexpected. Intuitively, it confirms the important role that minimal size constraints can have for local pattern discovery from  $n$ -ary relations.

To provide a complementary inside view of the extracted closed patterns, Figure 3 provides the number of closed patterns having a given “volume” (product of sizes of every attribute set) for the two data sets. First of all, we can see that a lot of patterns have a small volume, i.e., they contain only small sets of attributes. Then, the global trend seems to be much more regular in comparison with size distributions. This observation suggests that a volume constraint is less discriminant. The volume constraint should be used in addition to the minimal size constraint. This way, we can slightly reduce the minimal size thresholds so that we do not miss many potentially relevant patterns while keeping by means of the minimal volume constraint an enough stringent criterion for “kicking out” many patterns.



**Fig. 3.** Proportion of patterns having a given volume (public bicycle network data on the left and Distrowatch.com data on the right).

### 3.3 Outside view of patterns

We are now going to study closed pattern relevancy w.r.t. the outside of patterns. Assessing how much a local pattern is good at representing strong associations between sets of attributes w.r.t. the whole data set is assessed is rather new [6, 5]. The idea is to analyze for each pattern its outside slices (see Definition 3) and to see how different/similar they are to the inside slices. In short, if a pattern represents a strong association between its sets of attributes then outside slices of the pattern must be either filled by "0" values or at least must contain a lot of "0" values. In other words, patterns are putatively relevant if inside and outside slices are really different from each others in terms of the number of "0" values.

**Definition 3 (outside and inside slices).** Let  $X = \langle X^1, \dots, X^n \rangle$  be a  $n$ -set.  $Y = \langle Y^1, \dots, Y^n \rangle$  is a slice of  $X$  iff there exists  $i \in [1, n]$  s.t.  $\#Y^i = 1$  and  $\forall j \in 1 \dots n \setminus i, Y^j = X^j$ . If  $Y^i \subseteq X^i$  (respectively  $Y^i \subseteq D^i \setminus X^i$ ),  $Y$  is called an inside (resp. outside) slice.

In the best case, we would want patterns whose outside slices contain only "0" values. This would mean that patterns capture all the associations contained in the data set that involved elements of  $n - 1$  attributes belonging to the pattern. Figure 4 depicts an example of such a best-case pattern. We can see that all their slices are either filled by "0" (for outside slices) or by "1" (for inside ones). We denote this pattern type  $\mathcal{T}_{bc}$ .

Worst-case patterns have almost no significant difference between inside and outside slices in terms of the number of "0" values they contain. In other words, there are almost as many slices filled by "1" values (inside ones), as slices with one "0" values, as slices with two "0" values, etc. In that case, we cannot say that these patterns represent strong associations between their sets of attributes. Figure 5 presents three examples of such patterns. This pattern type is denoted  $\mathcal{T}_{wc}$ .

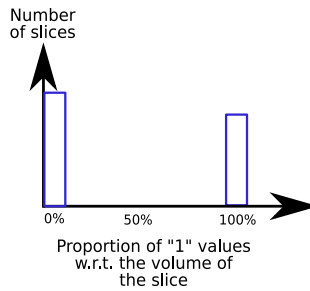
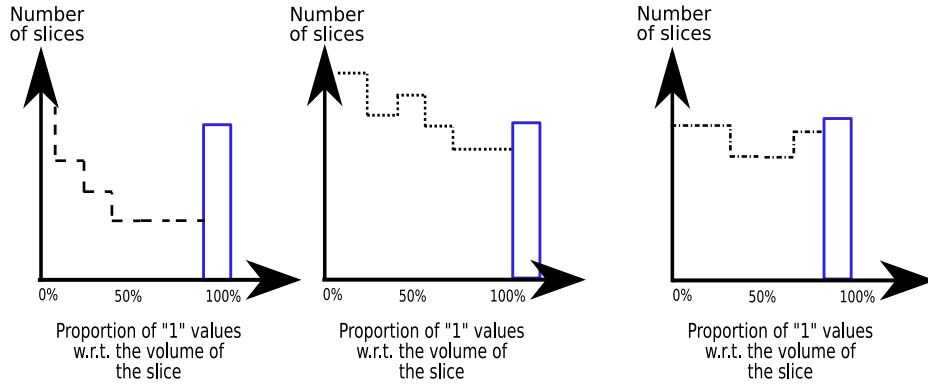


Fig. 4. Best-case pattern type  $\mathcal{T}_{bc}$

Two other types of patterns are of interest. They are presented in Figure 6 (left and middle). The first pattern type  $\mathcal{T}_{gc_1}$  has slices with at the same time



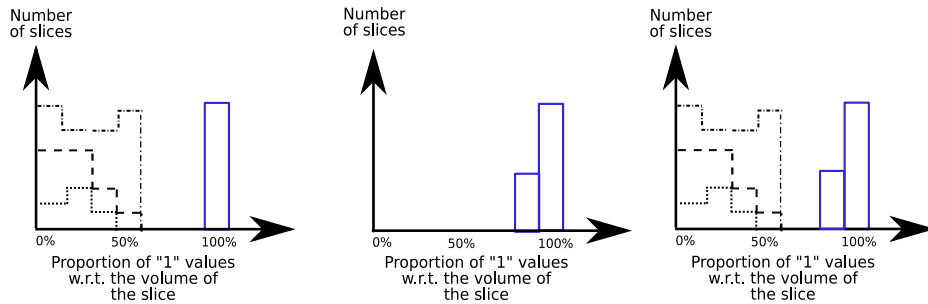


**Fig. 5.** Worst-case patterns  $\mathcal{T}_{wc}$

"0" and "1" values. But when they are not filled by "1", they contain a lot of "0" w.r.t. the volume of the slices. The second pattern type  $\mathcal{T}_{gc_2}$  highlights a problem with most local set patterns: data sets often contain values improperly set to "0" (false negative) or "1" (false positive). It commonly leads to an explosion of the number of extracted patterns and pattern presented in Figure 6 (middle) shows that some outside slices contain only a few of "0" values. Intuitively, we wish that this kind of outside slices can join the inside slices and become part of a fault-tolerant pattern.

The last pattern type  $\mathcal{T}_{oi}$  is somehow a fusion of pattern types we described as potentially interesting, i.e.,  $\mathcal{T}_{bc}$ ,  $\mathcal{T}_{gc_1}$  and  $\mathcal{T}_{gc_2}$ .

Our goal is to discuss if it could be possible to declaratively specify and then compute the following pattern types:  $\mathcal{T}_{bc}$ ,  $\mathcal{T}_{gc_1}$ ,  $\mathcal{T}_{gc_2}$  and  $\mathcal{T}_{oi}$ .



**Fig. 6.** Pattern types  $\mathcal{T}_{gc_1}$ ,  $\mathcal{T}_{gc_2}$  and  $\mathcal{T}_{oi}$  considered as interesting

### 3.4 Experimentation

We want to study how the extracted closed  $n$ -sets split into the different pattern types. Once closed patterns have been computed on our two real data sets, we assigned them to its closest pattern type considering each attribute separately. Details of this slightly ad-hoc procedure are omitted, we are only interested in the trends. Figure 7 presents the number of patterns that are assigned to each pattern type for each attribute. First, we can see that there is an abundance of worst-case patterns ( $\mathcal{T}_{wc}$ ). In several cases, they even represent the majority of patterns. The best-case pattern type  $\mathcal{T}_{bc}$  only represents a small proportion of patterns. Interestingly, patterns of type  $\mathcal{T}_{oi}$  are very present in our data sets.

Pattern type	Public bicycle network data				Distrowatch.com data		
	Day	Period	Departure	Arrival	Semester	Distribution	Country
$\mathcal{T}_{wc}$	2973	1406	3994	4263	857	1676	2800
$\mathcal{T}_{bc}$	306	776	179	148	192	20	7
$\mathcal{T}_{gc_1}$	2999	5502	3975	3730	2131	1464	460
$\mathcal{T}_{gc_2}$	695	6	0	0	0	0	0
$\mathcal{T}_{oi}$	1539	822	364	371	91	111	4

Fig. 7. Distribution of pattern types for each attribute and for the two data sets.

Figures 8 and 9 show examples of extracted patterns of each type with the same graphical representation as for Figures 4, 5, and 6.

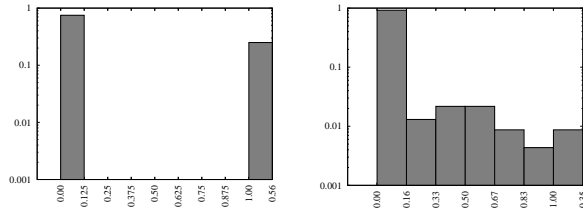


Fig. 8. Examples of patterns of types  $\mathcal{T}_{bc}$  and  $\mathcal{T}_{wc}$ .

Now, the challenge is to declaratively define patterns presented in Figure 4 and Figure 6 by means of primitive constraints on  $n$ -sets. We also want to exploit minimal and volume constraints and reduce, as much as possible, the number of extracted patterns that are putatively spurious while keeping potentially relevant ones.

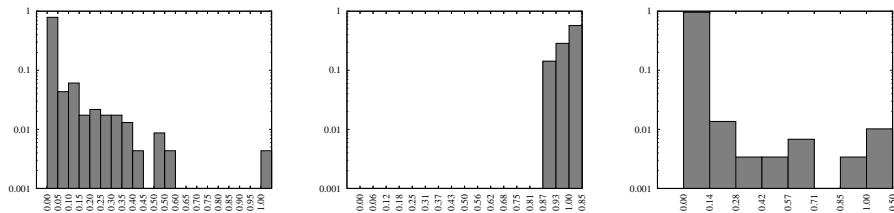


Fig. 9. Examples of patterns of types  $\mathcal{T}_{gc_1}$ ,  $\mathcal{T}_{gc_2}$  and  $\mathcal{T}_{oi}$ .

## 4 Tackling the closed pattern relevancy issue

### 4.1 Using minimal size and area constraints

As explained in Section 3.2, it may be interesting to extract patterns with sets of attributes of a minimal size or that cover a minimal number of tuples of the input data set. For the volume constraint, it would be wiser, as it has been done for the minimal size constraint, to compute a kind of relative volume, i.e., relative w.r.t. the size of the attribute domains.

**Definition 4.** *The (relative) minimal size constraint on  $n$ -sets in a  $n$ -ary relation is defined as  $\mathcal{C}_{\mu\text{-size}}(\langle X^1, \dots, X^n \rangle) \equiv \#X^1 \geq \mu_1 \times \#D^1 \wedge \dots \wedge \#X^n \geq \mu_n \times \#D^n$ . The volume constraint on  $n$ -sets in a  $n$ -ary relation is defined as  $\mathcal{C}_{\nu\text{-volume}}(\langle X^1, \dots, X^n \rangle) \equiv \prod_{i=1}^n \frac{\#X^i}{\#D^i} \geq \nu$ . Parameters  $\mu_1, \dots, \mu_n$  and  $\nu$  are percentages.*

These definitions are simple extensions of their counterpart studied for binary relations. Minimal size and volume constraints are anti-monotonic, what enables search-space pruning and constraint propagation. However, it is much more difficult to efficiently exploit them in an  $n$ -ary setting. The nature of the  $n$ -set search-space and of these constraints forces to consider all the attributes in a symmetric way. Indeed, they are connected to each others by  $n$  Galois connections and not only one as in the binary case. It means that at least  $n - 1$  attributes have to be enumerated. The size of the  $n - 1$  sets of attributes can then increase or decrease all along the enumeration. So we cannot consider any attribute as more important than another one while this was not the case for binary relation mining. Finally, every attribute must be enumerable and data sets must be accessible from any attribute (especially slices). All these requirements make  $n$ -set mining algorithms difficult to design and to implement. To the best of our knowledge, DATA-PEELER is the only one that currently exploits efficiently such constraints [5].

### 4.2 $\delta$ -isolated constraints

To extract patterns of type  $\mathcal{T}_{gc_1}$ , we need to design a constraint which enforces that every outside slices of closed patterns contain much more "0" values than

inside patterns. For this purpose, we can use the constraint  $\mathcal{C}_{\delta\text{-isolated}}$  (see Definition 5) which was first introduced in [5].

**Definition 5.** ( $\mathcal{C}_{\delta\text{-isolated}}$ ) A  $n$ -set  $X = \langle X^1, \dots, X^n \rangle$  is isolated w.r.t. the attribute  $X^i$ , denoted  $\mathcal{C}_{\delta\text{-isolated}}(X, i)$ , iff  $\forall x \in D^i \setminus X^i, \#(K \setminus \mathcal{R}) > \delta \times \#K$  where  $K = X^1 \times \dots \times \{x\} \times \dots \times X^n$  and  $\delta \in [0, 1[$  is a user-defined parameter.

The parameter  $\delta$  enables to set the required difference between inside and outside slices. It is the minimal proportion of "0" values allowed in an outside slice. The higher  $\delta$ , the more likely extracted closed patterns represent strong associations between the sets of attributes and the less patterns are extracted. Notice that  $\mathcal{C}_{\delta\text{-isolated}}$  ensures that patterns are closed. Indeed when  $\delta = 0$ , we have  $\mathcal{C}_{\delta\text{-isolated}} \equiv \mathcal{C}_{\text{closed}}$ .

Constraint  $\mathcal{C}_{\delta\text{-isolated}}$  is not (anti)-monotonic meaning that traditional techniques which relies on properties of (anti)-monotonicity cannot be applied. Fortunately it is piecewise (anti)-monotonic and DATA-PEELER can exploit this class of constraints to safely prune the search-space. To exploit a piecewise (anti)-monotonic constraint, the  $n$ -set search-space must be explicitly computed and stored for each generated candidate in the enumeration tree as done in [4]. Processing a piecewise (anti)-monotonic constraint consists in rewriting it so that, if the new form is satisfied, some candidate(s) deriving from the current one (i.e., leaves of the enumeration tree whose ancestor is the current candidate) may satisfy the original constraint. If not, safe pruning occurs. In the case of  $\mathcal{C}_{\delta\text{-isolated}}$  (see Definition 5), the occurrence of  $K$  on the left side of  $<$  is replaced by all elements in the current candidate *and* in the search space (upper-bound), whereas its occurrence on the right side of  $<$  is only replaced by all elements in the current candidate (lower-bound). For more detailed information, please refer to [5].

### 4.3 Fault-tolerant patterns

Pattern type  $\mathcal{T}_{gc_2}$  addresses a fundamental limitation of local pattern mining methods. Within local patterns, the strength of the association between the sets of attributes is often too strong in real-life data. Indeed, errors of measurement and/or Boolean encoding techniques may lead to erroneous zero values which will give rise to a combinatorial explosion of the number of extracted patterns. Based on our expertise in real-life data mining, it is now clear that the extraction of local patterns, their post-processing and their interpretation is not that relevant in noisy data which encode measured and/or computed relationships. Our hypothesis is that the extraction of local patterns in binary or  $n$ -ary relations containing also some "0" values might be useful and should be considered as a valuable alternative for actionable pattern discovery [3]. The extraction of fault-tolerant patterns in  $n$ -ary relations is a difficult problem, which has not been tackled yet. Here is a short overview of some fault-tolerant pattern mining algorithms focusing on the binary case.

The first proposed approaches concerned mono-dimensional patterns and/or the use of heuristic techniques. In [21], the frequent set mining task is extended towards fault-tolerance. A level-wise algorithm is proposed but their fault-tolerant property is not anti-monotonic while this is needed to achieve tractability. Therefore, [21] provides a greedy algorithm leading to an incomplete computation. [16] revisits this work. It looks for an anti-monotonic constraint such that a level-wise algorithm can provide every set whose density of one values is greater than  $\delta$  in at least  $\sigma$  situations. Anti-monotonicity is obtained by enforcing that every subset of extracted sets satisfies the constraint as well. In [7], the authors are interested in geometrical tiles (i.e., dense bi-sets which involve contiguous elements given orders on both attributes). Their local optimization algorithm is not deterministic and thus can not guarantee the global quality of the extracted patterns. Furthermore, the hypothesis on built-in orders can not be accepted on many data. Some fault-tolerant extensions of formal concepts have been proposed afterwards.

The proposal in [15] concerns an extension which can be computed efficiently but none of the appreciated properties are available. In [1], fault-tolerant formal concepts are defined as maximal rectangles with a bounded number of "0" values per object and per item. The proposed algorithm is sound and complete. [19] introduces a "zooming" approach on concept lattices. The so-called  $\alpha$ -Galois lattices exploit a partition on the objects to reduce the collection of the extracted bi-sets. [12] statistically analyzes, the effect of a noise – following a Bernoulli distribution – on the sizes and the numbers of exact itemsets. Both theoretically and experimentally, the authors have proved that the application of such a noise tends to generate exponentially more itemsets of sizes on the order of the logarithm of the original sizes. The AFI (Approximate Frequent Itemset) model, they proposed, copes with this issue by allowing a pattern – named ETI (Error Tolerant Itemset) – to contain, at most, a given proportion of '0' per transaction, and, another, per item. Then, the authors have successfully identified an anti-monotonic property able to prune sub-lattices of the search space, empty of ETI. Finally, they built on it an extractor displaying a rather good behavior in various synthetic and real-life settings.

## 5 Conclusion

Based on an empirical study and a background on local pattern discovery from binary relations, we have been considering the declarative specification of a priori relevancy for closed patterns that hold in  $n$ -ary relations. It has led to the definition of the pattern type denoted  $\mathcal{T}_{oi}$  that satisfies the constraint  $\mathcal{C}_{fault.tolerant}(X) \wedge \mathcal{C}_{\delta-isolated}(X) \wedge \mathcal{C}_{\mu-size}(X) \wedge \mathcal{C}_{\nu-volume}(X)$ .

The constraint  $\mathcal{C}_{fault.tolerant}(X)$  specifies that we are looking for fault-tolerant patterns, i.e.,  $n$ -sets covering mostly "1" values but with potentially some "0". While such constraints have been studied in binary relations, it remains an open problem in the general case of  $n$ -ary relations. We are currently studying it.

Constraint  $\mathcal{C}_{\delta\text{-isolated}}(X)$  enables to fix the difference between the number of "0" values inside and outside the patterns. In addition to this, it enforces the closedness property. This constraint being piecewise (anti)-monotonic, can be exploited adopting a specific search-space enumeration (see [5]).

Finally, we still consider the useful minimal size and volume constraints  $\mathcal{C}_{\mu\text{-size}}(X)$  and  $\mathcal{C}_{\nu\text{-volume}}(X)$ . We saw that they can be efficiently exploited when all the attributes are considered in a symmetric manner.

**Acknowledgments.** This work is partly funded by EU contract IST-FET IQ FP6-516169 ("Inductive Queries for Mining Patterns and Models") and by the French contract ANR-07-MDCO-014 Bingo2 ("Knowledge Discovery For and By Inductive Queries").

## References

1. Jérémy Besson, Céline Robardet, and Jean-François Boulicaut. Mining a new fault-tolerant pattern type as an alternative to formal concept discovery. In *ICCS'06: Proceedings of the 14th International Conference on Conceptual Structures*, pages 144–157, 2006.
2. Jérémy Besson, Céline Robardet, Jean-François Boulicaut, and Sophie Rome. Constraint-based formal concept mining and its application to microarray data analysis. *Intelligent Data Analysis*, 9(1):59–82, 2005.
3. Jean-François Boulicaut and Jérémy Besson. Actionability and formal concepts: a data mining perspective. In *ICFCA'08: Proceedings of the Sixth International Conference on Formal Concept Analysis*, pages 14–31, 2008.
4. Cristian Bucila, Johannes E. Gehrke, Daniel Kifer, and Walker White. Dualminer: A dual-pruning algorithm for itemsets with constraints. In *SIGKDD'02: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 42–51. ACM Press, 2002.
5. Loïc Cerf, Jérémy Besson, Céline Robardet, and Jean-François Boulicaut. Data peeler: Constraint-based closed pattern mining in  $n$ -ary relations. In *SDM'08: Proceedings of the Eighth SIAM International Conference on Data Mining*, pages 37–48. SIAM, 2008.
6. Aristides Gionis, Heikki Mannila, Taneli Mielikäinen, and Panayiotis Tsaparas. Assessing data mining results via swap randomization. *ACM Transactions on Knowledge Discovery from Data*, 1(3), 2007.
7. Aristides Gionis, Heikki Mannila, and Jouni K. Seppänen. Geometric and combinatorial tiles in 0-1 data. In *PKDD'04: Proceedings of the Eighth European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 173–184. Springer, 2004.
8. David Hand. Pattern detection and discovery. In *Proceedings of the ESF Exploratory Workshop on Pattern Detection and Discovery*, pages 1–12. Springer, 2002.
9. Robert Jaschke, Andreas Hotho, Christoph Schmitz, Bernhard Ganter, and Gerd Stumme. TRIAS—an algorithm for mining iceberg tri-lattices. In *ICDM'06: Proceedings of the Sixth IEEE International Conference on Data Mining*, pages 907–911. IEEE Computer Society, 2006.

10. Liping Ji, Kian-Lee Tan, and Anthony K. H. Tung. Mining frequent closed cubes in 3D data sets. In *VLDB'06: Proceedings of the 32nd international conference on Very large data bases*, pages 811–822. VLDB Endowment, 2006.
11. Daxin Jiang, Jian Pei, Murali Ramanathan, Chun Tang, and Aidong Zhang. Mining coherent gene clusters from gene-sample-time microarray data. In *KDD'04: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 430–439. ACM Press, 2004.
12. Jinze Liu, Susan Paulsen, Xing Sun, Wei Wang, Andrew B. Nobel, and Jan Prins. Mining approximate frequent itemsets in the presence of noise: Algorithm and analysis. In *SDM'06: Proceedings of the Sixth SIAM International Conference on Data Mining*. SIAM, 2006.
13. Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Efficient mining of association rules using closed itemset lattices. *Information Systems*, 24(1):25–46, 1999.
14. Jian Pei, Jiawei Han, and Runying Mao. CLOSET an efficient algorithm for mining frequent closed itemsets. In *DMKD'00: Proceedings of the 2000 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*. ACM Press, 2000.
15. Ruggero G. Pensa and Jean-François Boulicaut. Towards fault-tolerant formal concept analysis. In *AI\*IA 2005: Advances in Artificial Intelligence, Proceedings of the Ninth Congress of the Italian Association for Artificial Intelligence*, pages 212–223. Springer, 2005.
16. Jouni K. Seppänen and Heikki Mannila. Dense itemsets. In *KDD'04: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 683–688. ACM Press, 2004.
17. Gerd Stumme, Rafik Taouil, Yves Bastide, Nicolas Pasquier, and Lotfi Lakhal. Computing iceberg concept lattices with TITANIC. *Journal on Data and Knowledge Engineering*, 42:189–222, 2002.
18. Takeaki Uno, Masashi Kiyomi, and Hiroki Arimura. LCM ver.3: Collaboration of array, bitmap and prefix tree for frequent itemset mining. In *OSDM'05: Proceedings of the First International Workshop on Open Source Data Mining*, pages 77–86. ACM Press, 2005.
19. Véronique Ventos, Henry Soldano, and Thibaut Lamadon. Alpha galois lattices. In *ICDM'04: Proceedings of the Fourth IEEE International Conference on Data Mining*, pages 555–558. IEEE Computer Society, 2004.
20. Rudolf Wille. Restructuring lattice theory: an approach based on hierarchies of concepts. In Ivan Rival, editor, *Ordered sets*, pages 445–470. Reidel, 1982.
21. Cheng Yang, Usama Fayyad, and Paul S. Bradley. Efficient discovery of error-tolerant frequent itemsets in high dimensions. In *KDD'01: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 194–203. ACM Press, 2001.
22. Mohammed J. Zaki and Ching-Jui Hsiao. CHARM: An efficient algorithm for closed itemset mining. In *SDM'02: Proceedings of the Second SIAM International Conference on Data Mining*. SIAM, 2002.