Actionability and Formal Concepts: A Data Mining Perspective

Jean-François Boulicaut[1] and Jérémy Besson[1,2]

[1] INSA-Lyon, LIRIS CNRS UMR5205, F-69621 Villeurbanne cedex, France
[2] UMR INRA/INSERM 1235, F-69372 Lyon cedex 08, France
{Firstname.Name}@insa-lyon.fr

Abstract. The last few years, we have studied different set pattern mining techniques from binary data. It includes the computation of formal concepts to support various knowledge discovery processes. For instance, when considering post-genomics, we can exploit Boolean data sets that encode a relation between some genes and the proteins that may regulate them. In such a context, it appears interesting to exploit the analogy between a putative transcriptional module (i.e., a typically important hypothesis for gene regulation understanding) and a formal concept that holds within such data. In this paper, we assume that knowledge nuggets can be captured by collections of formal concepts and we discuss the challenging issue of mining/selecting actionable patterns from these collections, i.e., looking for relevant patterns that really support knowledge discovery. Therefore, a major issue concerns the computation of complete collections of formal concepts that satisfy user-defined constraints. This is useful not only to avoid the computation of too small patterns that might be due to noise (e.g., using size constraints on both their intents and extents) but also to introduce some fault-tolerance. We discuss the pros and the cons of some recent proposals in that direction.

1 Introduction

Many application domains can provide possibly huge boolean matrices whose rows denote objects and columns denote attributes (see Table 1 for toy examples). Mining such binary data, or formal contexts in the terminology of Formal Concept Analysis (FCA) [1], has been studied extensively. Indeed, popular data mining techniques have been designed for set pattern extraction (e.g., mining frequent itemsets or association rules, mining frequent closed itemsets or other condensed representations of frequent patterns [2,3]). We are interested in bi-set mining, i.e., the computation of local patterns that are sets of objects and sets of attributes being somehow "associated". Clearly, a formal concept is an interesting type of bi-set that satisfy a local constraint: its attribute set (or intent) is the maximal set of attributes that are true for each object of its associated supporting set of objects (or extent). Here, locality refers to the fact that checking whether a bi-set is a formal concept or not can be performed independently of the other patterns holding in the data. An example of a formal concept in

R. Medina and S. Obiedkov (Eds.): ICFCA 2008, LNAI 4933, pp. 14–31, 2008.

$\mathbf{r}_1$ from Table 1 is $(\{o_1, o_2, o_3, o_4\}, \{p_1, p_2\})$. Notice that this paper does not consider FCA as such and that, for instance, we are not really interested in the underlying concept lattice itself. Instead, we consider collections of formal concepts as collections of patterns. Also, we do not use formal concepts as condensed representations for collections of association rules (see [4] for a recent survey covering such issues).

Table 1. $\mathbf{r}_1$ (left) - $\mathbf{r}_2$ (right)

	p_1	p_2	p_3	p_4
o_1	1	1	0	0
o_2	1	1	0	0
o_3	1	1	0	0
o_4	1	1	1	1
o_5	0	0	1	1
o_6	0	0	1	1

	p_1	p_2	p_3	p_4
o_1	1	1	0	0
o_2	1	0	1	0
o_3	1	1	0	1
o_4	1	1	1	1
o_5	0	0	1	0
o_6	0	0	1	1

Let us introduce a couple of motivating applications for our perspective on formal concept mining (see, e.g., [5]). The objects can denote biological samples and the attribute can denote boolean gene expression properties, e.g., the fact that a given gene is over-expressed in a given sample. In such a case, the boolean properties have to be derived from the continuous values measured by, e.g., the microarray technology, and a formal concept provides an hypothesis on a maximal group of genes that have the same expression property in a given group of biological samples. A second example would be to consider that some transcription factors (i.e., the proteins which regulate gene expression) are the studied objects for which we record whether they can bind or not on the promoter sequence of some studied genes. Here again, a formal concept can be interpreted as an hypothesis on a maximal set of genes whose co-expression might be explained by its associated set of transcription factors. Clearly, one of the motivations for collecting gene expression data is indeed to be able to discover such hypothesis that correspond, from a biological perspective, to putative synexpression groups, transcription modules, regulation pathways, etc.

In this paper, we are interested in the various application domains for which, given a binary data set, one can consider that its formal concepts are a priori interesting statements about the data. In theory, computing formal concepts is exponential in the smallest dimension of the data matrix (i.e., the number of objects or the number of attributes). An important question concerns the tractability of their computation for practical applications. Given the major effort of the last decade, it turns out that computing collections of formal concepts that hold in large binary matrices can be feasible. Researchers have designed algorithms that compute complete collections of formal concepts [6]. Since these patterns are built on closed sets, the extensive research on (frequent) closed set extraction has inspired constraint-based mining of formal concepts (see, e.g., [7]): every formal concept which furthermore satisfies a size constraint on one of its components (e.g., a minimal size for its intent or its extent) can be extracted

efficiently. This is however not really satisfactory when considering that our ultimate goal is to mine actionable patterns, i.e., relevant formal concepts that can indeed be interpreted by human experts to catalyze knowledge discovery. Real data sets can hold hundreds of thousands of formal concepts: it is clear that looking for actionable ones among the many spurious or irrelevant ones is extremely hard or even impossible. In fact, it is interesting to look at one fundamental limitation of Knowledge Discovery processes based on formal concepts. Within such local patterns, the strength of the association of the two set components is often too strong in real-life data. Indeed, errors of measurement and boolean encoding techniques can lead to "erroneous" zero or one values. Unexpected zero values give rise to a combinatorial explosion of the number of formal concepts because interesting patterns are split into less relevant ones. For example, let us consider the data from Table 1. Assume that $\mathbf{r}_1$ is a reliable representation of a phenomenon but that data collection and preprocessing lead to $\mathbf{r}_2$ instead (i.e., some noise has been introduced), the number of formal concepts in $\mathbf{r}_2$ is approximately twice larger than in $\mathbf{r}_1$. While this concerns zero values that may be one values, we can also consider what happens in the reverse situation: the intuition is that when some zero values have been encoded as one values by error, many "small" formal concepts may hold. Therefore, we need to avoid computing too small patterns but also we have to somehow relax that no exception (zero value) can be accepted, i.e., what we call fault-tolerance. For instance, a bi-set like $(\{o_1, o_2, o_3, o_4\}, \{p_1, p_2\})$ is not a formal concept in $\mathbf{r}_2$ but it may be considered as a relevant pattern: its objects and attributes are strongly associated (only one zero value) and, furthermore, its "outside" objects and attributes contain more than one zero value.

Our contribution here is to consider how some data mining researchers have designed more or less pragmatic methods to address these problems. We avoid to produce the technical details that are available from the referenced papers. We will discuss the DMiner proposal for constraint-based mining of formal concepts in the challenging case where, for instance, we want to "push" size constraints on both dimensions [8,5]. We will also consider different approaches for designing fault-tolerant patterns based on formal concepts [9,10,11]. The survey paper [12] is a discussion on the needed trade-off between extraction feasibility, completeness, relevancy, and ease of interpretation of such fault-tolerant pattern types. Notice also that [12] contains empirical results on both synthetic and real data.

Section 2 discusses the DMiner solution for constraint-based mining of formal concepts. In Section 3, we consider the obstacles and present some available solutions for actionable pattern discovery based on formal concepts. Section 4 briefly concludes on some current open issues in that area.

2 Formal Concept Mining

Let $\mathcal{O}$ denotes a set of objects and $\mathcal{P}$ denotes a set attributes (or properties). In Table 1, $\mathcal{O} = \{o_1, \cdots, o_6\}$ and $\mathcal{P} = \{p_1, \cdots, p_4\}$. A data set is the materialization of a binary relation $\mathbf{r} \subseteq \mathcal{O} \times \mathcal{P}$. We write $(o_i, p_j) \in \mathbf{r}$ to denote that property

j holds for object i. Boolean matrices like $\mathbf{r}_1$ and $\mathbf{r}_2$ in Table 1 are classical representations for such relations.

Formal concepts can be considered as bi-sets, i.e., couples of sets (X, Y) from $2^{\mathcal{O}} \times 2^{\mathcal{P}}$ that satisfy a constraint denoted $\mathcal{C}_{FC}(X, Y)$. The definition of such a constraint might be expressed in terms of Galois operators and closures. In this paper, let us specify it in terms of the conjunction of a density constraint (first conjunct) and a relevancy constraint (second and third conjuncts), following the presentation from [11].

Definition 1 (Formal Concept). *A bi-set $(X, Y) \in 2^{\mathcal{O}} \times 2^{\mathcal{P}}$ is a formal concept in $\mathbf{r}$ if it satisfies the constraint $\mathcal{C}_{FC}(X, Y) \equiv (\forall x \in X, \forall y \in Y (x, y) \in \mathbf{r}) \wedge (\forall x \in \mathcal{O} \setminus X, \exists y \in Y \text{ s.t. } (x, y) \notin \mathbf{r}) \wedge (\forall y \in \mathcal{P} \setminus Y, \exists x \in X \text{ s.t. } (x, y) \notin \mathbf{r})$.*

Informally, it means that for a formal concept (X, Y), if we perform permutations of rows and columns such that all the elements from X (resp. Y) are contiguous, we observe a maximal rectangle of true values (no zero value inside, at least one zero value outside). Another useful analogy to understand the semantics of formal concepts is to consider them as bi-cliques in the bi-partite graphs represented by the boolean matrix. Computing every formal concept that holds within a boolean matrix is NP-hard. As soon as none of the dimensions is small, this extraction task is not feasible. Furthermore, when the computation is tractable, we often get a huge amount of formal concepts (e.g., millions) even in rather small data sets. As we mentioned in our introduction, this is definitively not acceptable for actionable pattern discovery. Constraint-based mining is a partial but impressive solution to both problems, i.e., computational complexity and relevancy. The idea is that the analyst can often exploit some background knowledge to specify declarative constraints that may hold for the extracted formal concepts of interest. It happens that some of these user-defined constraints can be exploited (say "pushed deeply") by the mining algorithm to prune efficiently the search space. For example, we may need patterns with a minimal number of elements on both dimensions (a counterpart of the popular minimal frequency constraint for itemset mining [13]) and/or patterns covering at least a given number of elements of $\mathbf{r}$ (intuitively, a minimum area constraint for the associated "rectangle"). As a result, when considering knowledge discovery processes based on formal concepts, we are generally computing collections of bi-sets that satisfy not only $\mathcal{C}_{FC}$ but also a user-defined constraint $\mathcal{C}_{UD}$:

$$\{(X, Y) \in 2^{\mathcal{O}} \times 2^{\mathcal{P}} \mid \mathcal{C}_{FC}(X, Y) \wedge \mathcal{C}_{UD}(X, Y)\}$$

Figure 1 provides examples of well-known user-defined constraints where α and β denote some thresholds, $a \in \mathcal{O}$, $b \in \mathcal{P}$, $E \subseteq \mathcal{O}$, and $E' \subseteq \mathcal{P}$ are constraint parameters. For instance, $\mathcal{C}^1_{size} \wedge \mathcal{C}^2_{size}$ or $\mathcal{C}_{area}$ are two different constraints to specify that patterns have to be "large enough". Also, $\mathcal{C}_{mean}$ is just one example of a constraint which enforces that the average of an external positive value associated to each element of the extent is greater than a given threshold.

Not every constraint can be processed efficiently. We have a special interest for monotonic and anti-monotonic constraints (see Definition 2) that have nice

$\mathcal{C}_{UD}(X,Y)$		
$\mathcal{C}^1_{size}$	$\equiv$	$\lvert X\rvert > \alpha$
$\mathcal{C}^2_{size}$	$\equiv$	$\lvert Y\rvert > \alpha$
$\mathcal{C}_{area}$	$\equiv$	$\lvert X\rvert \times \lvert Y\rvert > \alpha$
$\mathcal{C}_{mean}$	$\equiv$	$\sum_{i\in X} Val^+(i)/\lvert X\rvert > \alpha$
$\mathcal{C}_{member}$	$\equiv$	$a \in X \wedge b \in Y$
$\mathcal{C}_{inter}$	$\equiv$	$\lvert X \cap E\rvert > \alpha \wedge \lvert Y \cap E'\rvert < \beta$

Fig. 1. Examples of interesting constraints on bi-sets

properties when the search space is organized as a lattice structure thanks to a specialization relation.

Definition 2 (Monotonic constraints). *A constraint $\mathcal{C}$ is anti-monotonic w.r.t. the specialization order $\preceq$ on E iff $\forall a, b \in E$ s.t. $a \preceq b$ then $\neg\, \mathcal{C}(a) \Rightarrow \neg\, \mathcal{C}(b)$. $\mathcal{C}$ is monotonic w.r.t. $\preceq$ iff $\forall a, b \in E$ s.t. $a \preceq b$ then $\neg\, \mathcal{C}(b) \Rightarrow \neg\, \mathcal{C}(a)$.*

Following a path in an enumeration tree for candidate patterns, an anti-monotonic constraint is satisfied for all the patterns before a specific pattern and not satisfied afterwards. A popular example is the anti-monotonicity of a minimal frequency constraint which specifies that the size of the extent has to be greater than a given threshold (i.e., $\mathcal{C}^1_{size}$). This constraint can efficiently reduce the search-space and remove spurious patterns whose extent is too small. However, in the many applications where the data set is large on both dimensions and when the density in terms of true values is high, the only way to achieve tractability seems to be an increase of the minimal size threshold for the extent. Doing so, we clearly loose a priori interesting formal concepts (the larger the extent, the smaller the intent, and vice versa). Therefore, we may want to use other constraints like, for example, minimal size constraint on both the intents and the extents, i.e., conjunctions $\mathcal{C}^1_{size} \wedge \mathcal{C}^2_{size}$. Unfortunately, using the standard enumeration on formal concepts (enumeration of the intent and computation of the extend), most algorithms can only exploit anti-monotonic constraints on the intent and monotonic constraints on the extent, i.e., a conjunction like $|X| > \alpha \wedge |Y| < \beta$ (say $\mathcal{C}^1_{size} \wedge \neg\, \mathcal{C}^2_{size}$). Furthermore, it is not that simple to exploit constraints like $\mathcal{C}_{area}$, $\mathcal{C}_{mean}$, $\mathcal{C}_{member}$, and $\mathcal{C}_{inter}$. Even though they can be used to capture important expectation from the analysts, these constraints are neither anti-monotonic nor monotonic.

The DMiner algorithm is a depth-first search algorithm inspired by both Ganter's algorithm [14] and DualMiner [15]. The principle of Ganter's algorithm enables to identify from an extracted formal concept the smallest formal concept that may follow it. Doing so, we can avoid the generation of many sets that are not closed, i.e., which can not correspond to formal concepts. On the other hand, for efficiency purposes, we have to follow the order related to the standard set inclusion. Given the data from Table 2, it is far more efficient to generate the formal concept $(\{o_1,\ o_2,\ o_3\}, \{p_1,\ p_2,\ p_3,\ p_4\})$ from $(\{o_1,\ o_2,\ o_3,\ o_5\}, \{p_1,\ p_3\})$ than from $(\{o_2,\ o_3,\ o_4\}, \{p_9,\ p_{10}\})$. The pattern $(\{o_1, o_2, o_3, o_5\}, \{p_1, p_3\})$ already contains a lot of information that can be used to

Table 2. A boolean context $\mathbf{r}_3$

	Attributes									
	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}
o_1	1	1	1	1	0	1	1	0	0	0
o_2	1	1	1	1	0	0	0	0	1	1
o_3	1	1	1	1	0	0	0	0	1	1
o_4	0	0	0	0	1	1	1	1	1	1
o_5	1	0	1	0	1	1	1	1	0	0

generate $(\{o_1, o_2, o_3\}, \{p_1, p_2, p_3, p_4\})$, e.g., we know that $(\{o_1, o_2, o_3\}, \{p_1, p_3\})$ holds in $\mathbf{r}_3$ because it is "included" in $(\{o_1, o_2, o_3, o_5\}, \{p_1, p_3\})$ which is a formal concept.

It means that we do not have to scan all the data corresponding to the new patterns to be extracted. In our running example, we only need to scan the data corresponding to $(\{o_1, o_2, o_3\}, \{p_2, p_4\})$. It is even more crucial when large data sets are considered. Thus, we adopt a binary enumeration of the smallest set ($\mathcal{O}$ or $\mathcal{P}$) and the other set is computed by means of the Galois connection. This enumeration combines both Ganter's principle and a prefix-based enumeration.

It is important to exploit constraints not only to increase the relevancy of the computed patterns but also to increase computational efficiency. Most of the available algorithms can push monotonic and/or anti-monotonic constraints according to set inclusion on one dimension. We however argued in the previous section that this is not enough. Unlike most of the formal concept mining algorithms, DMiner does not consider that each candidate is only represented by means of two sets, i.e., the intent and the extent. The enumerated set, let us say the intent, is split into two sets, the first one representing the set of elements that belong to any formal concept extracted from the current candidate and the second one containing the elements that still have to be enumerated (see the inspiring principle in [15]). The two sets are the bottom and the top of a lattice which represents the current search space. For example, we may have a candidate $(o_1o_2o_3o_5, (p_1, p_1p_2p_3))$ where the intent is represented by two sets $\{p_1\}$ and $\{p_1p_2p_3\}$ instead of only one set $\{p_1p_2p_3\}$. By this way during the enumeration we always know precisely which search-space is related to the current candidate and thus increase the number of constraints the algorithm can handle. In our example the candidate is supported by the extent $\{o_1, o_2, o_3, o_5\}$ and it represents all the attribute sets Y (intents) such that $\{p1\} \subseteq_L Y \subseteq_L \{p_1, p_2, p_3\}$ where $(X, Y) \subseteq_L (X', Y') \Rightarrow X \subset X' \wedge Y \subset Y'$, i.e., the attribut sets $\{p_1\}$, $\{p_1, p_2\}$, $\{p_1, p_3\}$ and $\{p_1, p_2, p_3\}$ in our example. Notice that a candidate of the form $(O, (X, X))$ denotes the bi-set (O, X).

This pattern representation enables to push a larger class of constraints than only the anti-monotonic constraints. Indeed, each candidate denotes a search space in the form of an attribute lattice with its associated object set. For example, let us consider candidate $C = (o_1o_2o_3o_5, (p_1, p_1p_2p_3p_4))$ in $\mathbf{r}_3$, each formal concept derived from C which contains p_1 contains at most the attributes from $\{p_1, p_2, p_3, p_4\}$ and its associated object set is included in $\{o_1, o_2, o_3, o_5\}$. It

enables to push difficult constraints like $\mathcal{C}_{area}$ and $\mathcal{C}_{mean}$ which are neither monotonic nor anti-monotonic. Indeed, since we have an attribute lattice, we can compute bounds for some constraints. For instance, we can see that the area of C is between $|\{o_1, o_2, o_3, o_5\}| \times |\{p_1\}| = 4$ and $|\{o_1, o_2, o_3, o_5\}| \times |\{p_1, p_2, p_3, p_4\}| = 16$. If we are looking for formal concepts with an area of size at least 17 or of size at most 3, then pattern C can be pruned safely, i.e., it can not lead to acceptable formal concepts.

We adopted simple arrays as data structure to store the sets of objects and attributes. A time complexity analysis shows that it is as efficient as the other more complex data structures used in depth-first search algorithms. Finally, to check whether a set X is closed, DMiner does not have to scan all the sets from $\mathcal{P} \setminus X$. Indeed, only attributes which have been removed from the search space by enumeration have to be checked. Instead of going into much details, let us provide two DMiner executions (see Figure 2 and Figure 3) on the data sets given in Table 3. In Figure 2, the algorithm starts with the candidate $(o_1o_2o_3, (\emptyset, p_1p_2))$ representing the set of all possible patterns of $\mathbf{r}_4$. Then the attribute p_1 is selected to proceed the enumeration. Two new candidates $(o_1o_2o_3, (\emptyset, p_2))$ and $(o_1o_2o_3, (p_1, p_1p_2))$ are generated. After enumerating the attribute p_2, four formal concepts are extracted $(o_1o_2o_3, \emptyset)$, (o_1o_2, p_2), (o_2o_3, p_1) and (o_2, p_1p_2). Figure 3 provides an other example of DMiner execution.

To investigate the efficiency of DMiner, we studied its complexity using the time delay, i.e., the complexity to go from one solution to the next one [16]. DMiner time delay is in the worst case equal to $O(n^2m)$ where n is the size of the enumerated set and m is the size of the other one. This complexity is the same as

Table 3. Extraction contexts $\mathbf{r}_4$ (left) and $\mathbf{r}_5$ (right)

	p_1	p_2
o_1	0	1
o_2	1	1
o_3	1	0

	p_1	p_2	p_3
o_1	0	0	1
o_2	0	1	0
o_3	1	0	1

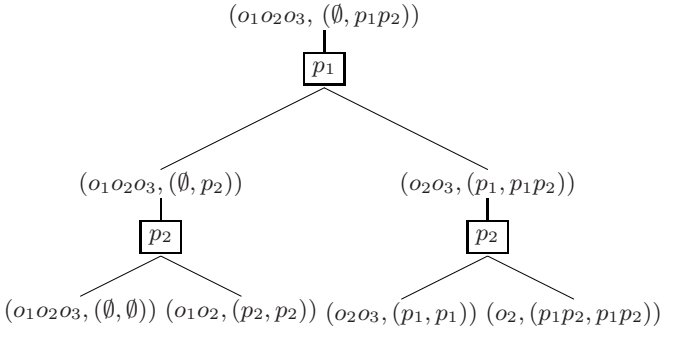

Fig. 2. Formal concept extraction on $\mathbf{r}_4$

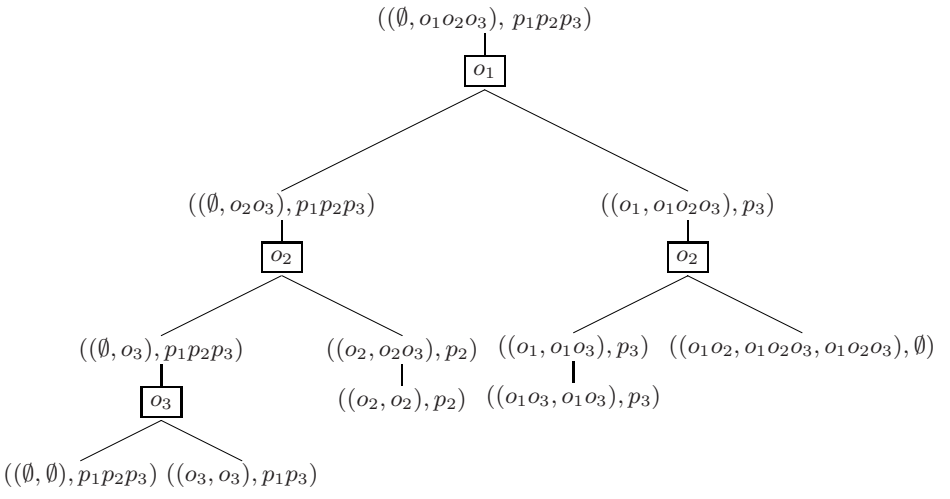

Fig. 3. Formal concept extraction on $\mathbf{r}_5$

for Ganter's algorithm [14]. The time delay in average is $O(n-\log_2(K)+1)O(mn)$ where K is the number of formal concepts in the data set. It is between $O(nm)$ and $O(n^2m)$ according to K. This is an interesting result when considering the use of DMiner on data sets in which many formal concepts hold.

To refer to one concrete example, let us recall the application described in [5]. It concerns the analysis of a data set that records (a) the existence of putative binding sites of 94 transcription factors on the promoter sequences of 304 genes (selection of human genes), and (b) the over-expression property of these same genes in 10 biological simples (individuals). In other terms, the boolean context implies 104 objects (94 transcription factors and 10 biological situations, more precisely 5 for healthy individuals and 5 for diabetic patients) and 304 genes. Formal concept discovery from such a boolean context is already hard. Notice also that the obtained boolean context was rather dense in terms of true values (17% of the cells containing a true value). In such a situation, even efficient algorithms for mining frequent closed sets can turn to be intractable. This data set is particular: there are very few frequent formal concepts with a relative frequency threshold above 0.1 on genes (5 534 patterns) and then the number of formal concepts increases very fast. Without any constraint, we get more than five million formal concepts within a few minutes. In this context, extracting actionable formal concepts needs for a very low frequency threshold, otherwise almost no formal concept can be computed. Notice that actionable patterns corresponding to interesting biological hypothesis have been found by means of formal concepts holding in this data set [16,5].

Figure 4 shows the running time of formal concepts extractions in the biological data set with varying the minimal frequency threshold. The competitors in the experiment are three different algorithms used for computing frequent closed sets, namely `ac-miner` [17], `closet` [18] and Charm [19].

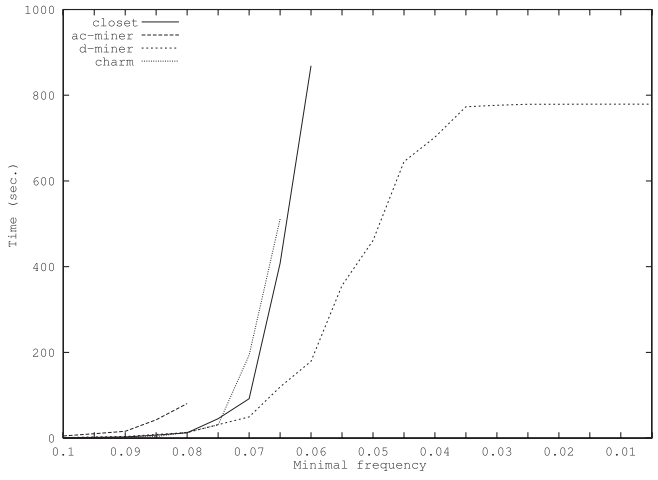

Fig. 4. An application [5]

3 Looking for Actionable Patterns

Let us now come back on our main goal that is actionable pattern discovery based on collections of formal concepts. The challenge is to mine relevant formal concepts that can indeed be interpreted by human experts to catalyze knowledge discovery. In case this is not possible, we may also look for application-independent and/or application-dependent post-processing techniques that can be applied on formal concepts to support the discovery of actionable patterns.

- When, among other things, a data set captures correctly a phenomenon of interest, collections of formal concepts can be huge and they may contain large collections of spurious patterns (false positives) and/or irrelevant patterns w.r.t. domain knowledge. This is somehow inherent to (unsupervized) local pattern discovery techniques. The two main directions of research to improve this situation concern (a) the use of randomization techniques for statistical validity assessment for the extracted patterns (see, e.g., [20]) and (b) the use of user-defined constraints to specify subjective interestingness issues based on domain knowledge (e.g., using $\mathcal{C}_{member}$ or $\mathcal{C}_{inter}$ primitive constraints).
- When considering the problem of erroneous true values (the property should not be satisfied but we record that it is satisfied), assuming that this is fundamentally rare, many "small" formal concepts may hold. Therefore, we need to avoid computing these too small patterns by using minimal size or minimal area constraints (See Section 2). In practice, using these constraints $\mathcal{C}^1_{size}$, $\mathcal{C}^2_{size}$ and/or $\mathcal{C}_{area}$ can be extremely efficient.
- When considering the problem of erroneous false values (the property is satisfied but we record that it is not satisfied), assuming again that this can not be too frequent, the number of extracted formal concepts will increase

fast w.r.t. what it should be if the data were correct. This is a clear need for fault-tolerance and we discuss several proposals hereafter (see Section 3.1).
– The number of patterns, even if only relevant ones have been collected, is indeed a problem. Knowledge discovery needs for human expert assessment of patterns and browsing or inspecting thousands of patterns is definitively not possible. The solution can come from (a) the design of pattern databases and advanced querying tools, and (b) summarization techniques based on, for instance, clustering methods. Considering problems and solutions for pattern database management is related to the emergent topic of inductive databases and will not be discussed further in this paper. Considering summarization techniques has been successfully applied (see Section 3.2).

3.1 Introducing Fault-Tolerance

We first revisit the definition of formal concepts for a fairly natural introduction of fault-tolerance. In the following, we say that a bi-set (X, Y) is included in a bi-set (X', Y') denoted $(X, Y) \subseteq (X', Y')$ iff $(X \subseteq X') \wedge (Y \subseteq Y')$.

Definition 3. *Assume $\mathcal{Z}_l(x, Y)$ denotes the number of false values of an object x on the attributes in Y, i.e., $|\{y \in Y | (x, y) \notin \mathbf{r}\}|$. Similarly, let $\mathcal{Z}_c(y, X) = |\{x \in X | (x, y) \notin \mathbf{r}\}|$ be the number of false values of an attribute y on the objects in X.*

Definition 4 (FC). *A bi-set $(X, Y) \in 2^{\mathcal{O}} \times 2^{\mathcal{P}}$ is a formal concept in $\mathbf{r}$ iff*
(2.1) $\forall x \in X,\ \mathcal{Z}_l(x, Y) = 0 \ \wedge \ \forall y \in Y,\ \mathcal{Z}_c(y, X) = 0$
(2.2) $\forall x \in \mathcal{O} \setminus X,\ \mathcal{Z}_l(x, Y) \geq 1 \ \wedge \ \forall y \in \mathcal{P} \setminus Y,\ \mathcal{Z}_c(y, X) \geq 1$

Sub-constraint 2.1 expresses that a formal concept contains only true values. Sub-constraint 2.2 denotes that formal concept relevancy is enhanced by a maximality property. It is now straightforward to introduce a declarative specification of fault-tolerance.

Definition 5 (DRBS [11]). *Given integer parameters δ and ϵ, a bi-set $(X, Y) \in 2^{\mathcal{O}} \times 2^{\mathcal{P}}$ is called a* DRBS *pattern (Dense and Relevant Bi-Set) in $\mathbf{r}$ iff*
(3.1) $\forall x \in X,\ \mathcal{Z}_l(x, Y) \leq \delta \ \wedge \ \forall y \in Y,\ \mathcal{Z}_c(y, X) \leq \delta$
(3.2) $\forall e \in \mathcal{O} \setminus X,\ \forall x \in X,\ \mathcal{Z}_l(e, Y) \geq \mathcal{Z}_l(x, Y) + \epsilon$
$\wedge \ \forall e' \in \mathcal{P} \setminus Y,\ \forall y \in Y,\ \mathcal{Z}_c(e', X) \geq \mathcal{Z}_c(y, X) + \epsilon$
(3.3) It is maximal, i.e., $\nexists (X', Y') \in 2^{\mathcal{O}} \times 2^{\mathcal{P}}$ s.t. (X', Y') is a DRBS *pattern and $(X, Y) \subseteq (X', Y')$.*

DRBS patterns have at most δ false values per object and per attribute (Sub-constraint 3.1) and are such that each outside object (resp. attribute) has at least ϵ false values plus the maximal number of false values on the inside objects (resp. attributes) according to Sub-constraint 3.2. The size of a DRBS pattern increases with δ such that, when $\delta > 0$, it happens that several bi-sets are included in each other. Only maximal bi-sets are kept (Sub-constraint 3.3). Notice that δ and ϵ can be chosen differently on objects and on attributes. It is clear that when $\delta = 0$ and $\epsilon = 1$, DRBS $\equiv$ FC.

Table 4. A boolean context $\mathbf{r}_6$

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
o_1	1	0	1	0	1	0	0
o_2	1	1	1	1	0	1	0
o_3	0	1	1	1	1	1	1
o_4	0	0	0	1	1	1	0
o_5	1	0	0	0	0	1	0
o_6	1	1	1	1	1	0	0
o_7	1	1	1	1	1	0	0

Table 5. Permutations on $\mathbf{r}_6$ to illustrate Example 1

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
o_1	1	0	1	0	1	0	0
o_2	1	1	1	1	0	1	0
o_3	0	1	1	1	1	1	1
o_4	0	0	0	1	1	1	0
o_6	1	1	1	1	1	0	0
o_7	1	1	1	1	1	0	0
o_5	1	0	0	0	0	1	0

Example 1. *If* $\delta = \epsilon = 1$, $(X, Y) = (\{o_1, o_2, o_3, o_4, o_6, o_7\}, \{p_3, p_4, p_5\})$ *is a* DRBS *pattern in* $\mathbf{r}_6$ *(see Table 4). Columns* p_1, p_2, p_6 *and* p_7 *contain at least two false values on* X, *and* o_5 *contains three false values on* Y *(see Table 5).*

Collections of DRBS patterns can be computed in rather small data sets by using the correct and complete algorithm DR-Miner [11]. It is again based on the Dual-Miner principle [15]. Notice that a preliminary approach for specifying "symmetrical" fault-tolerant formal concepts had been introduced in [9] (i.e., the so-called $\alpha\beta$-concepts) and that it has been compared with the DRBS pattern domain in [12].

Let us now consider a rather different (and say pragmatic) extension of formal concepts which is not symmetrical. It has been designed thanks to the previous work on one of the few approximate condensed representations of frequent sets, the so-called δ-free sets [17]. δ-free sets are some kind of generators whose counted frequencies enable to infer the frequency of many sets (sets included in their so-called δ-closures) without further counting but with a bounded error. The δ-freeness constraint on attributes sets has been formalized in terms of the size of the supported sets of objects (this so-called frequency has to be different from the frequency of all its subsets by at least δ). Notice also that the so-called generators in [21] or key patterns in [22] are special cases of δ-free sets ($\delta = 0$). The 0-closure is the classical closure operator and applying it on each 0-free set is one way to produce every closed set and thus every formal concept.

The idea for the so-called FBS patterns (Free set Based Bi-Set) is to consider bi-sets built on the δ-closure of δ-free attribute sets associated to their supporting

sets of objects [23,10]. The intuition is that it will provide strong associations between sets of objects and sets of attributes. To avoid technical details, let us just comment an example.

Example 2. *If* $\delta = 1$, $\{p_2\}$ *is a* 1*-free set and* $(\{o_2, o_3, o_6, o_7\}, \{p_1, p_2, p_3, p_4, p_5\})$ *is a* FBS *pattern in* $\mathbf{r}_6$ *(see Table 6). The* 1*-closure of* $\{p_2\}$ *is* $\{p_1, p_2, p_3, p_4, p_5\}$ *because* p_1*,* p_3*,* p_4*, and* p_5 *are the attributes which are almost always true (*1 *exception is accepted) for the objects that have the property denoted by* p_2*, i.e., objects* o_2*,* o_3*,* o_6*, and* o_7*.*

When δ=0, attribute 0-free sets are the minimal elements of the equivalence classes of the relation "has the same supporting set of objects". Then, the 0-closure provides a closed set of attributes that associated to its supporting set of object gives a formal concept. In other terms, when $\delta = 0$, FBS $\equiv$ FC.

Table 6. A permutation on $\mathbf{r}_6$ to illustrate Example 2

	p_1	p_2	p_3	p_4	p_5	p_6	p_7
o_2	1	1	1	1	0	1	0
o_3	0	1	1	1	1	1	1
o_6	1	1	1	1	1	0	0
o_7	1	1	1	1	1	0	0
o_1	1	0	1	0	1	0	0
o_4	0	0	0	1	1	1	0
o_5	1	0	0	0	0	1	0

The extraction of FBS patterns can be extremely efficient thanks to δ-freeness anti-monotonicity. Notice that FBS patterns are bi-sets with a bounded number of exception per column but every bi-set with a bounded number of exception per column is not necessarily a FBS pattern.

One crucial issue that can explain the added-value of formal concepts is the ease of interpretation thanks to the Galois connection. What happens with these DRBS and FBS extensions? For each bi-set (X, Y), do we have a function which associates X and Y? If a function exists which associates to each set X (resp. Y) at most a unique set Y (resp. X), the interpretation of each bi-set is much easier. Furthermore, if the two functions are monotonically decreasing, i.e. when the size of X (resp. Y) increases, the size of its associated set Y (resp. X) decreases. This property is meaningful since the more we have objects inside a bi-set, the less there are attributes that can be associated to describe them (or vice versa).

In a FBS pattern, we have no function from $2^{\mathcal{P}}$ to $2^{\mathcal{O}}$ but we have a function from $2^{\mathcal{O}}$ to $2^{\mathcal{P}}$. The definition of this pattern is indeed not symmetrical. In many data sets, including huge and dense ones, complete collections of FBS can be extracted efficiently but we need for a better characterization of more relevant FBS patterns which might remain easy to extract from huge databases, e.g., what is the impact of different δ values for the δ-free-set part and the δ-closure

computation? How can we avoid an unfortunate distribution of the false values among the same objects?

By construction, a DRBS has been defined such that we have a bounded number of exceptions per object and per attribute. Two interesting properties have been proven [11,16]. First, when $\epsilon > 0$, DRBS patterns are embedded by two functions ϕ (resp. ψ) which associate to X (resp. Y) a unique set Y (resp. X). Then, for a fixed δ, we have monotonicity properties of ϕ and ψ. Unfortunately, the functions loose this property on the whole DRBS collection. Furthermore, we have not identified yet an intentional definition of these functions.

Notice that [12] contains an empirical evaluation of these different pattern domains on both artificially noised data sets and a real-life medical data set. These extensions of formal concepts have been specified in a constraint-based mining framework, i.e., we have a declarative specification of the constraints on the patterns such that we can work on correct and complete implementations for computing them. Notice however that it is computationally challenging to work with the most elegant extension, i.e., DRBS. Other researchers have considered fault-tolerant pattern mining. To the best of our knowledge, most of the related work has concerned mono-dimensional patterns and/or the use of heuristic techniques [24,25]. [26] is one of the interesting proposal for geometrical tile mining (i.e., dense bi-sets which involve contiguous elements given orders on both dimensions). More recently, other attempts to relax closeness have been considered [27,28]. Fault-tolerance in general and its application to closed sets and formal concepts in particular definitively appears as an important topic for real-life data mining. Let us now consider another pragmatic approach for finding actionable patterns based on collections of formal concepts.

3.2 Post-processing Collections of Formal Concepts

The number of formal concepts which hold in a data set and which can be computed thanks to, for instance, user-defined constraints, can be huge. We already pointed out that many factors can have a dramatic impact on the number of formal concepts. For instance, when an error of measurement or an intrinsic variability of the observed phenomenon lead to a zero value whereas the value true should be obtained, we have to face with an explosion of the number of formal concepts. The fault-tolerant extensions discussed above are definitively not the ultimate solution: indeed, when tractable, the extractions still provide too many patterns. One simple idea is to post-process the formal concepts and more generally the extracted bi-sets to group the ones which are similar enough. It is rather straightforward to design similarity measures between bi-sets and one can perform, for instance, a hierarchical clustering method to group formal concepts. These groups can be interpreted by computing some kind of "quasi-formal concepts" that are in fact the bi-sets made of the union of the intents and the union of the extents of all the formal concepts that belong to a cluster [29]. This has been applied successfully to a real application in the domain of human gene expression data analysis [30]. In this application, mining a 90×5327 boolean gene expression matrix has given rise to 64 836 formal concepts. When

considering size constraints that have enforced the formal concepts to imply at least five biological samples and five genes, only 1 669 patterns have been selected: this is however too much for a human interpretation for each of them. We applied the hierarchical clustering and then we have built about 50 quasi-synexpression groups, i.e., sets of genes strongly associated to sets of biological samples. One of them has been carefully interpreted and this has given rise to interesting biological new hypothesis [30].

Following the same ideas, we can also consider the possibility to build clusters or co-clusters by exploiting the formal concepts. A co-clustering (see, e.g., [31]) provides linked partitions on both dimensions (objects and attributes) and, in Boolean data, it tends to compute rectangles with mainly true (resp. false) values. Heuristic techniques (i.e., local optimization) enable to compute one bi-partition. In fact, a bi-clustering provides a global structure over the data while fault-tolerant extensions of formal concepts are typical local patterns which can lead to the discovery of unexpected but yet relevant local associations. In [32], co-clusters are computed from collections of bi-sets like formal concepts: it is a KMeans like clustering that does not work on objects or attributes but on bi-sets. Notice also that once a bi-partition has been computed, by such a technique or with another co-clustering approach, we can again use the bi-sets for characterization purposes [10]. Notice that computing clusters or co-clusters based on formal concepts is different from selecting formal concepts that may constitute a collection of clusters or co-clusters (see, e.g., [33]).

In a conceptual clustering framework, Mineau et al. [34] present simple pre-pruning and post-pruning techniques that can be applied in reasonable time on large classification structures. The paper presents three such techniques: one is based on the definition of constraints over the generalization language, the other two are based on discrimination metrics applied on links between classes or on the classes themselves.

Another interesting relationship that may be studied further is the formal analogies between tiling as considered in [35] and co-clustering. Also, the quite active area of subspace clustering is clearly related to local pattern detection and constraint-based mining of bi-sets (see [36,37] for surveys).

4 Conclusion

We have discussed several aspects of actionable pattern discovery from collections of formal concepts. Thanks to constraint-based mining algorithms, computing complete collections of formal concepts that satisfy user-defined constraints is feasible for some useful constraints. This framework has been used to specify fault-tolerant extensions of formal concepts. This is definitively needed for mining large and noisy Boolean data sets. The DRBS pattern domain appears as a well-designed class but the price to pay is its computational complexity. The good news are that (a) DRBS pattern extractions may involve further user-defined constraints which can be used for efficient pruning, and (b) one can look for more efficient data structures and thus a more efficient DR-Miner implementation. A pragmatic usage could be to extract some bi-sets, e.g., formal concepts, and

then select some of them (say $B = (X, Y)$) for further extensions towards fault-tolerant patterns: the computation of a DRBS patterns (say $B' = (X', Y')$) such that the constraint $B \subseteq B'$ is enforced. We also mentioned post-processing techniques that, for instance, cluster patterns like formal concepts not only to decrease the number of hypothesis that have to be interpreted by human experts but also to enhance their relevancy, e.g., achieving some kind of fault-tolerance.

Let us now conclude this paper by introducing a few open questions related to both Formal Concept Analysis and constraint-based mining of patterns.

Constraints and formal concept mining. One important direction of research remains constraint-based mining of formal concepts for hard constraints, that is constraints that are neither monotonic nor anti-monotonic and for which new enumeration strategies have to be designed. Constraints that refer to statistical measures (e.g., based on standard deviation), are typical examples of such hard constraints. Even though some types of hard constraints have been studied on simple pattern domains like itemsets (e.g., optimization constraints that look for the best k patterns w.r.t. an objective measure), so far, few researchers have tried to upgrade these algorithms to the 2-dimensional case (i.e., for bi-set mining). New constraint types must be designed as well. For instance, [38] has proposed to increase the relevancy of set patterns by means of constraints that exploit textual resources in the context of biological data analysis. This is a promising direction of research.

Extensions of formal concepts. The constraint-based mining framework supports the exploitation of domain knowledge to increase a priori relevancy of patterns. We notice however that in more and more applications, the data can hardly be presented as a single binary relation. Let us take an example: assume that we have genes with a lot of information about them (e.g., functions, associated transcription factors), biological experiments enabling to measure the expression of these genes and finally details about the experiments. Consider now that we want to extract the sets of house-keeping genes that are over-expressed in the same biological experiments that are related to muscle tissues. The extraction task sounds like a formal concept extraction, same words same ideas, but here the data are structured into multiple n-ary relations ($n > 2$). One hot topic is to revisit the principles of pattern mining in binary relation when considering n-ary relations. For instance, the counterpart of formal concepts in cubes (3-ary relations) has been recently investigated [39,40]. Also, extending the ideas of closed set and condensed representation mining in a multi-relational setting is a timely challenge. [41] presents a formal concept analysis-based class hierarchy design that can be viewed as normal forms for class hierarchies where each normal form addresses particular design goals. An overview of work in the area is presented by highlighting the formal concept analysis notions that are involved. [42] introduces a first formalization of a network of contexts, i.e., the data is represented by the means of several contexts. Descriptions of conceptual coherences within the formalized network of contexts are introduced.

Storing and querying collections of patterns. Following the inductive database perspective [43,2], patterns may be considered as first-class citizens and we have to design databases that not only contain data but also patterns like formal concepts or clustering results. Technically, many challenges concern the efficient management of large collections of set patterns because relational databases are not targeted towards this type of data. Also, the design of general-purpose query languages to support knowledge discovery by means of queries remains an open problem.

Local-to-Global. Many global patterns (e.g., classifiers, clusterings) can be considered as collections of local patterns that satisfy some kind of global constraints. These local patterns are themselves satisfying local constraints. The popular association-based classification approach [44] is an obvious example of a Local to Global (L2G) scheme: standard association rules are the local patterns (i.e., local constraints are the minimal frequency and minimal confidence constraints). The various proposals for building classifiers from them are then based on different global constraints on these collections of association rules. Clustering can be considered within a L2G framework as well (see, e.g;, [32]). A better understanding of cross-fertilization between local pattern detection and global pattern discovery is an extremely active research direction that may deliver interesting insights in the next few years. Among others, the exciting challenge of constrained clustering may benefit from such a Local to Global approach (see, e.g., [45] for a constrained co-clustering based on formal concepts).

Acknowledgements. The authors thank Céline Robardet and Ruggero G. Pensa for their participation to most of the research results discussed in this paper. This research is partially funded by EU contract IST-FET IQ FP6-516169.

References

1. Ganter, B., Stumme, G., Wille, R. (eds.): Formal Concept Analysis. LNCS (LNAI), vol. 3626. Springer, Heidelberg (2005)
2. Boulicaut, J.F.: Inductive databases and multiple uses of frequent itemsets: the cInQ approach. In: Meo, R., Lanzi, P.L., Klemettinen, M. (eds.) Database Support for Data Mining Applications. LNCS (LNAI), vol. 2682, pp. 1–23. Springer, Heidelberg (2004)
3. Calders, T., Rigotti, C., Boulicaut, J.F.: A survey on condensed representations for frequent sets. In: Boulicaut, J.-F., De Raedt, L., Mannila, H. (eds.) Constraint-Based Mining and Inductive Databases. LNCS (LNAI), vol. 3848, pp. 64–80. Springer, Heidelberg (2006)
4. Valtchev, P., Missaoui, R., Godin, R.: Formal concept analysis for knowledge discovery and data mining: The new challenges. In: Eklund, P.W. (ed.) ICFCA 2004. LNCS (LNAI), vol. 2961, pp. 352–371. Springer, Heidelberg (2004)
5. Besson, J., et al.: Constraint-based formal concept mining and its application to microarray data analysis. Intelligent Data Analysis 9(1), 59–82 (2005)
6. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. Experimental and Theoretical Artificial Intelligence 14(2-3), 189–216 (2002)

7. Stumme, G., et al.: Computing iceberg concept lattices with titanic. Data & Knowledge Engineering 42, 189–222 (2002)
8. Besson, J., Robardet, C., Boulicaut, J.F.: Constraint-based mining of formal concepts in transactional data. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 615–624. Springer, Heidelberg (2004)
9. Besson, J., Robardet, C., Boulicaut, J.F.: Mining formal concepts with a bounded number of exceptions from transactional data. In: Goethals, B., Siebes, A. (eds.) KDID 2004. LNCS, vol. 3377, pp. 33–45. Springer, Heidelberg (2005)
10. Pensa, R., Boulicaut, J.F.: From local pattern mining to relevant bi-cluster characterization. In: Famili, A.F., et al. (eds.) IDA 2005. LNCS, vol. 3646, pp. 293–304. Springer, Heidelberg (2005)
11. Besson, J., Robardet, C., Boulicaut, J.F.: Mining a new fault-tolerant pattern type as an alternative to formal concept discovery. In: Schärfe, H., Hitzler, P., Øhrstrøm, P. (eds.) ICCS 2006. LNCS (LNAI), vol. 4068, pp. 144–157. Springer, Heidelberg (2006)
12. Besson, J., et al.: Constraint-based mining of fault-tolerant patterns from boolean data. In: Bonchi, F., Boulicaut, J.-F. (eds.) KDID 2005. LNCS, vol. 3933, pp. 55–71. Springer, Heidelberg (2006)
13. Goethals, B., Zaki, M.: Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations FIMI 2003, Melbourne, USA (2003)
14. Ganter, B.: Two basic algorithms in concept analysis. Technical report, Technische Hochschule Darmstadt, Germany, Preprint 831 (1984)
15. Bucila, C., et al.: DualMiner: A dual-pruning algorithm for itemsets with constraints. Data Mining and Knowledge Discovery 7(4), 241–272 (2003)
16. Besson, J.: Découvertes de motifs pertinents pour l'analyse du transcriptome: application à l'insulino-résistance. PhD thesis, INSA-Lyon, 69621 Villeurbanne cedex, France. (in French) (2005)
17. Boulicaut, J.F., Bykowski, A., Rigotti, C.: Free-sets: a condensed representation of boolean data for the approximation of frequency queries. Data Mining and Knowledge Discovery 7(1), 5–22 (2003)
18. Pei, J., Han, J., Mao, R.: CLOSET an efficient algorithm for mining frequent closed itemsets. In: Proceedings ACM SIGMOD Workshop DMKD 2000 (2000)
19. Zaki, M.J., Hsiao, C.J.: CHARM: An efficient algorithm for closed itemset mining. In: Proceedings SIAM DM 2002, Arlington, USA (2002)
20. Gionis, A., et al.: Assessing data mining results via swap randomization. In: Proceedings ACM SIGKDD 2006, Philadelphia, USA, pp. 167–176 (2006)
21. Pasquier, N., et al.: Efficient mining of association rules using closed itemset lattices. Information Systems 24, 25–46 (1999)
22. Bastide, Y., et al.: Mining frequent patterns with counting inference. SIGKDD Explorations 2, 66 (2000)
23. Pensa, R., Boulicaut, J.F.: Towards fault-tolerant formal concept analysis. In: Bandini, S., Manzoni, S. (eds.) AI*IA 2005. LNCS (LNAI), vol. 3673, pp. 212–223. Springer, Heidelberg (2005)
24. Yang, C., Fayyad, U., Bradley, P.S.: Efficient discovery of error-tolerant frequent itemsets in high dimensions. In: Proceedings ACM SIGKDD, pp. 194–203. ACM Press, New York (2001)
25. Seppänen, J.K., Mannila, H.: Dense itemsets. In: Proceedings ACM SIGKDD 2004, Seattle, USA, pp. 683–688. ACM Press, New York (2004)
26. Gionis, A., Mannila, H., Seppänen, J.K.: Geometric and combinatorial tiles in 0-1 data. In: Boulicaut, J.-F., et al. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 173–184. Springer, Heidelberg (2004)

27. Cheng, J., Ke, Y., Ng, W.: Delta-tolerance closed frequent itemsets. In: Proceedings IEEE ICDM 2006, Hong Kong, China, pp. 139–148 (2006)
28. Cheng, H., Yu, P.S., Han, J.: Ac-close: Efficiently mining approximate closed itemsets by core pattern recovery. In: Proceedings IEEE ICDM 2006, Hong Kong, China, pp. 839–844 (2006)
29. Robardet, C., et al.: Using classification and visualization on pattern databases for gene expression data analysis. In: Proceedings PaRMa'04 co-located with EDBT 2004, Heraclion-Crete, Greece. CEUR Proceedings, vol. 96, pp. 107–118 (2004)
30. Blachon, S., et al.: Clustering formal concepts to discover biologically relevant knowledge from gene expression data. Silico Biology 7(0033), 1–15 (2007)
31. Dhillon, I.S., Mallela, S., Modha, D.S.: Information-theoretic co-clustering. In: Proceedings ACM SIGKDD 2003, Washington, USA, pp. 89–98. ACM Press, New York (2003)
32. Pensa, R., Robardet, C., Boulicaut, J.F.: A bi-clustering framework for categorical data. In: Jorge, A.M., et al. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 643–650. Springer, Heidelberg (2005)
33. Durand, N., Crémilleux, B.: Ecclat: a new approach of clusters discovery in categorical data. In: Proceedings ES 2002, Cambridge, UK, pp. 177–190. Springer, Heidelberg (2002)
34. Mineau, G., Godin, A.B.,, R.: Simple pre- and post-pruning techniques for large conceptual clustering structures. In: Electronic Transactions on Artificial Intelligence (ETAI), pp. 1–20 (2000)
35. Geerts, F., Goethals, B., Mielikäinen, T.: Tiling databases. In: Suzuki, E., Arikawa, S. (eds.) DS 2004. LNCS (LNAI), vol. 3245, pp. 278–289. Springer, Heidelberg (2004)
36. Parsons, L., Haque, E., Liu, H.: Subspace clustering for high dimensional data: a review. ACM SIGKDD Exploration Newsletter 6, 90–105 (2004)
37. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: A survey. IEEE/ACM Trans. Comput. Biol. Bioinf. 1, 24–45 (2004)
38. Kléma, J., et al.: Mining plausible patterns from genomic data. In: Proceedings IEEE CBMS 2006, Salt Lake City, USA, pp. 183–190 (2006)
39. Jaschke, R., et al.: TRIAS: An algorithm for mining iceberg tri-lattices. In: Proceedings IEEE ICDM 2006, Hong Kong, China, pp. 907–911 (2006)
40. Ji, L., Tan, K.L., Tung, A.K.H.: Mining frequent closed cubes in 3D datasets. In: Proceedings VLDB 2006, Seoul, Korea, pp. 811–822 (2006)
41. Godin, R., Valtchev, P.: Formal concept analysis-based class hierarchy design in object-oriented software development. In: Ganter, B., Stumme, G., Wille, R. (eds.) Formal Concept Analysis. LNCS (LNAI), vol. 3626, pp. 192–207. Springer, Heidelberg (2005)
42. Wille, R.: Conceptual structures of multicontexts. In: 4th Int. Conf. on Conceptual Structures ICCS 1996, pp. 23–39. Springer, Heidelberg (1996)
43. Imielinski, T., Mannila, H.: A database perspective on knowledge discovery. Communications of ACM 39, 58–64 (1996)
44. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: Proceedings KDD 1998, pp. 80–86. AAAI Press, Menlo Park (1998)
45. Pensa, R.G., Robardet, C., Boulicaut, J.F.: Constraint-driven Co-Clustering of 0/1 Data. In: S.B., et al. (eds.) Constrained Clustering: Advances in Algorithms, Theory and Applications. Data Mining and Knowledge Discovery Series, Chapman & Hall/CRC Press (to appear, 2008)