Efficient Mining of Association Rules Based on Formal Concept Analysis

Lotfi Lakhal[1], Gerd Stumme[2]

[1] IUT d'Aix-en-Provence, Département d'Informatique
Avenue Gaston Berger, F-13625 Aix-en-Provence cedex, France
http://www.lif.univ-mrs.fr/EQUIPES/BD/
[2] Chair of Knowledge & Data Engineering, Department of Mathematics and Computer Science, University of Kassel, Wilhelmshöher Allee 73, D–34121 Kassel, Germany
http://www.kde.cs.uni-kassel.de

Abstract. Association rules are a popular knowledge discovery technique for warehouse basket analysis. They indicate which items of the warehouse are frequently bought together. The problem of association rule mining has first been stated in 1993. Five years later, several research groups discovered that this problem has a strong connection to Formal Concept Analysis (FCA). In this survey, we will first introduce some basic ideas of this connection along a specific algorithm, TITANIC, and show how FCA helps in reducing the number of resulting rules without loss of information, before giving a general overview over the history and state of the art of applying FCA for association rule mining.

1 Introduction

Knowledge discovery in databases (KDD) is defined as the non-trivial extraction of valid, implicit, potentially useful and ultimately understandable information in large databases [23]. For several years, a wide range of applications in various domains have benefited from KDD techniques and many work has been conducted on this topic. The problem of mining frequent itemsets arose first as a sub-problem of mining association rules [1], but it then turned out to be present in a variety of problems [24]: mining sequential patterns [3], episodes [32], association rules [2], correlations [12, 43], multi-dimensional patterns [26, 28], maximal itemsets [7, 54, 29], closed itemsets [47, 37, 38, 41]. Since the complexity of this problem is exponential in the size of the binary database input relation and since this relation has to be scanned several times during the process, efficient algorithms for mining frequent itemsets are required.

In Formal Concept Analysis (FCA), the task of mining frequent itemsets can be described as follows: Given a set G of objects, a set M of attributes (or items), a binary relation $I \subseteq G \times M$ (where $(g, m) \in I$ is read as "object g has attribute m"), and a threshold minsupp $\in [0, 1]$, determine all subsets X of M (also called *itemsets* or *patterns* here) where the *support* $\text{supp}(X) := \frac{\text{card}(X')}{\text{card}(G)}$ (with $X' := \{g \in G \mid \forall m \in X : (g, m) \in I\}$) is above the threshold minsupp.

The set of these *frequent itemsets* itself is usually not considered as final result of the mining process, but rather as intermediate step. Its most prominent use are certainly association rules. The task of mining association rules is to determine all pairs $X \to Y$ of subsets of M such that $\text{supp}(X \to Y) := \text{supp}(X \cup Y)$ is above the threshold minsupp, and the *confidence* $\text{conf}(X \to Y) := \frac{\text{supp}(X \cup Y)}{\text{supp}(X)}$ is above a given threshold minconf $\in [0, 1]$. Association rules are for instance used in warehouse basket analysis, where the warehouse management is interested in learning about products frequently bought together.

Since determining the frequent itemsets is the computationally most expensive part, most research has focused on this aspect. Most algorithms follow the way of the well-known Apriori algorithm [2]. It is traversing iteratively the set of all itemsets in a levelwise manner. During each iteration one level is considered: a subset of candidate itemsets is created by joining the frequent itemsets discovered during the previous iteration, the supports of all candidate itemsets are counted, and the infrequent ones are discarded. A variety of modifications of this algorithm arose [13, 34, 42, 48] in order to improve different efficiency aspects. However, all of these algorithms have to determine the supports of *all* frequent itemsets and of some infrequent ones in the database.

Other algorithms are based on the extraction of maximal frequent itemsets (i. e., from which all supersets are infrequent and all subsets are frequent). They combine a levelwise bottom-up traversal with a top-down traversal in order to quickly find the maximal frequent itemsets. Then, all frequent itemsets are derived from these ones and one last database scan is carried on to count their support. The most prominent algorithm using this approach is Max-Miner [7]. Experimental results have shown that this approach is particularly efficient for extracting maximal frequent itemsets, but when applied to extracting all frequent itemsets, performances drastically decrease because of the cost of the last scan which requires roughly an inclusion test between each frequent itemset and each object of the database. As for the first approach, algorithms based on this approach have to extract the supports of *all* frequent itemsets from the database.

While all techniques mentioned so far count the support of all frequent itemsets, this is by no means necessary. Using basic results from Formal Concept Analysis (FCA), it is possible to derive from some known supports the supports of all other itemsets: it is sufficient to know the support of all frequent concept intents. This observation was independently made by three research groups around 1997/98: the first author and his database group in Clermont–Ferrand [37], M. Zaki in Troy, NY [52], and the second author in Darmstadt [44].

The use of FCA allows not only an efficient computation, but also to drastically reduce the number of rules that have to be presented to the user, without any information loss. We present therefore some 'bases' for association rules which are non-redundant and from which all the others can be derived. Interestingly, up to now most researchers focus on the first aspect of efficiently computing the set of all frequent concept intents (also called *frequent closed sets*) and related condensed representations, but rarely consider condensed representations of the association rules.

This survey consists of two parts: In the next four sections, we provide an introduction to mining association rules using FCA, which mainly follows our own work,[1] before we provide a more general overview over the field in Section 6. First, we will first briefly relate the notions of Formal Concept Analysis to the association rule mining problem. Then we discuss how so-called iceberg concept lattices represent frequent itemsets, and how they can be used for visualizing the result. In Section 4, we will sketch one specific algorithm, called TITANIC, as an illustrative example for exploiting FCA theory for efficient computation. The reduction of the set of asociation rules to the relevant ones is the topic of Section 5. Section 6 gives an overview over the current state of the art.

2 Mining Frequent Itemsets with Formal Concept Analysis

Consider two itemsets X and Y such that both describe exactly the same set of objects, i. e., $X' = Y'$. So if we know the support of one of them, we do not need to count the support of the other one in the database. In fact, we can introduce an equivalence relation θ on the powerset $\mathfrak{P}(M)$ of M by $X\theta Y \iff X' = Y'$. If we knew the relation from the beginning, it would be sufficient to count the support of one itemset of each class only — all other supports can then be derived. Of course one does not know θ in advance, but one can determine it along the computation. It turns out that one usually has to count the support of more than one itemset of each class, but normally not of all of them. This observation leads to a speed-up in computing all frequent itemsets, since counting the support of an itemset is an expensive operation.

We will see below that it is not really necessary to compute all frequent itemsets for solving the association rule problem: it will be sufficient to focus on the frequent concept intents (here also called *closed itemsets* or *closed patterns*). As well known in FCA, each concept intent is exactly the largest itemset of the equivalence class of θ it belongs to. For any itemset $X \subseteq M$, the concept intent of its equivalence class is the set X''. The concept intents can hence be considered as 'normal forms' of the (frequent) itemsets. In particular, the concept lattice contains all information to derive the support of all (frequent) itemsets.

3 Iceberg Concept Lattices

While it is not really informative to study the set of all frequent itemsets, the situation changes when we consider the closed itemsets among them only. We call the concepts they belong to *frequent concepts*, and the set of all frequent concepts *iceberg concept lattice* of the context $\mathbb{K}$ for the threshold minsupp. We illustrate this by a small example. Figure 1 shows the iceberg concept lattice of

[1] This part summarizes joint work with Yves Bastide, Nicolas Pasquier, and Rafik Taouil as presented in [5, 40, 45, 46].

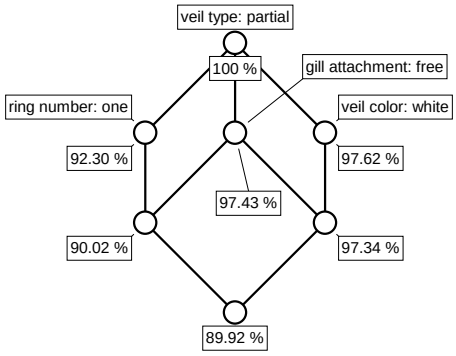

Fig. 1. Iceberg concept lattice of the mushroom database with minsupp = 85 %

the MUSHROOM database from the *UCI KDD Archive* [6] for a minimum support of 85 %.

The MUSHROOM database consists of 8,416 objects (mushrooms) and 22 (nominally valued) attributes. We obtain a formal context by creating one (Boolean) attribute for each of the 80 possible values of the 22 database attributes. The resulting formal context has thus 8,416 objects and 80 attributes. For a minimum support of 85 %, this dataset has 16 frequent itemsets, namely all 2^4 possible combinations of the attributes 'veil type: partial', 'veil color: white', 'gill attachment: free', and 'ring number: one'. Only seven of them are closed. The seven frequent concepts are shown in Figure 1.

In the diagram, each node stands for formal concept. The intent of each concept (i. e., each frequent closed itemset) consists of the attributes labeled at or above the concept. The number shows its support. One can clearly see that all mushrooms in the database have the attribute 'veil type: partial'. Furthermore the diagram tells us that the three next-frequent attributes are: 'veil color: white' (with 97.62 % support), 'gill attachment: free' (97.43 %), and 'ring number: one' (92.30 %). There is no other attribute having a support higher than 85 %. But even the combination of all these four concepts is frequent (with respect to our threshold of 85 %): 89.92 % of all mushrooms in our database have one ring, a white partial veil, and free gills. This concept with a quite complex description contains more objects than the concept described by the fifth-most attribute, which has a support below our threshold of 85 %, since it is not displayed in the diagram.

In the diagram, we can detect the implication

$$\{\text{ring number: one, veil color: white}\} \Rightarrow \{\text{gill attachment: free}\} \ .$$

It is indicated by the fact that there is no concept having 'ring number: one' and 'veil color: white' (and 'veil type: partial') in its intent, but not 'gill attachment: free'. This implication has a support of 89.92 % and is globally valid in the database (i. e., it has a confidence of 100 %).

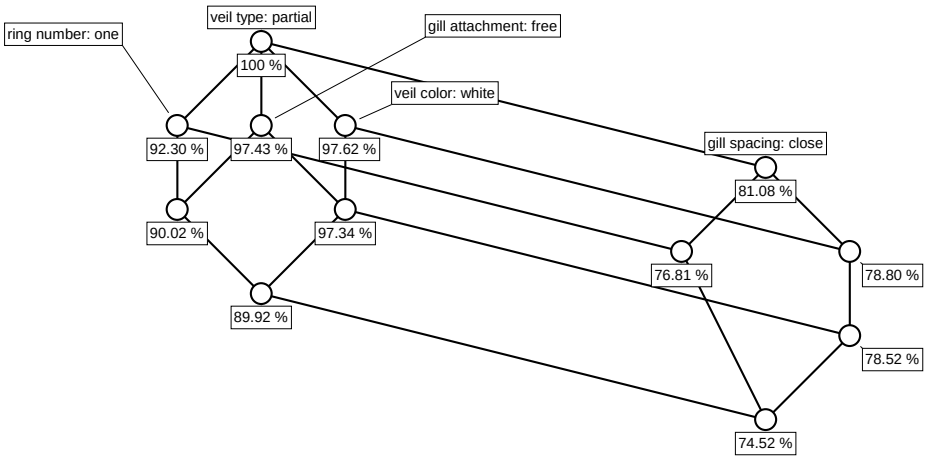

Fig. 2. Iceberg concept lattice of the mushroom database with minsupp = 70 %

If we want to see more details, we have to decrease the minimum support. Figure 2 shows the MUSHROOM iceberg concept lattice for a minimum support of 70 %. Its 12 concepts represent all information about the 32 frequent itemsets for this threshold. One observes that, of course, its top-most part is just the iceberg lattice for minsupp = 85 %. Additionally, we obtain five new concepts, having the possible combinations of the next-frequent attribute 'gill spacing: close' (having support 81.08 %) with the previous four attributes. The fact that the combination {veil type: partial, gill attachment: free, gill spacing: close} is not realized as a concept intent indicates another implication:

$$\{\text{gill attachment: free, gill spacing: close}\} \Rightarrow \{\text{veil color: white}\} \qquad (*)$$

This implication has 78.52 % support (the support of the most general concept having all three attributes in its intent) and — being an implication — 100 % confidence.

By further decreasing the minimum support, we discover more and more details. Figure 3 shows the MUSHROOMS iceberg concept lattice for a minimum support of 55 %. It shows four more partial copies of the 85 % iceberg lattice, and three new, single concepts.

The Mushrooms example shows that iceberg concept lattices are suitable especially for strongly correlated data. In Table 1, the size of the iceberg concept lattice (i. e., the number of all frequent closed itemsets) is compared with the number of all frequent itemsets. It shows for instance, that, for the minimum support of 55 %, only 32 frequent closed itemsets are needed to provide all information about the support of all 116 frequent itemsets one obtains for the same threshold.

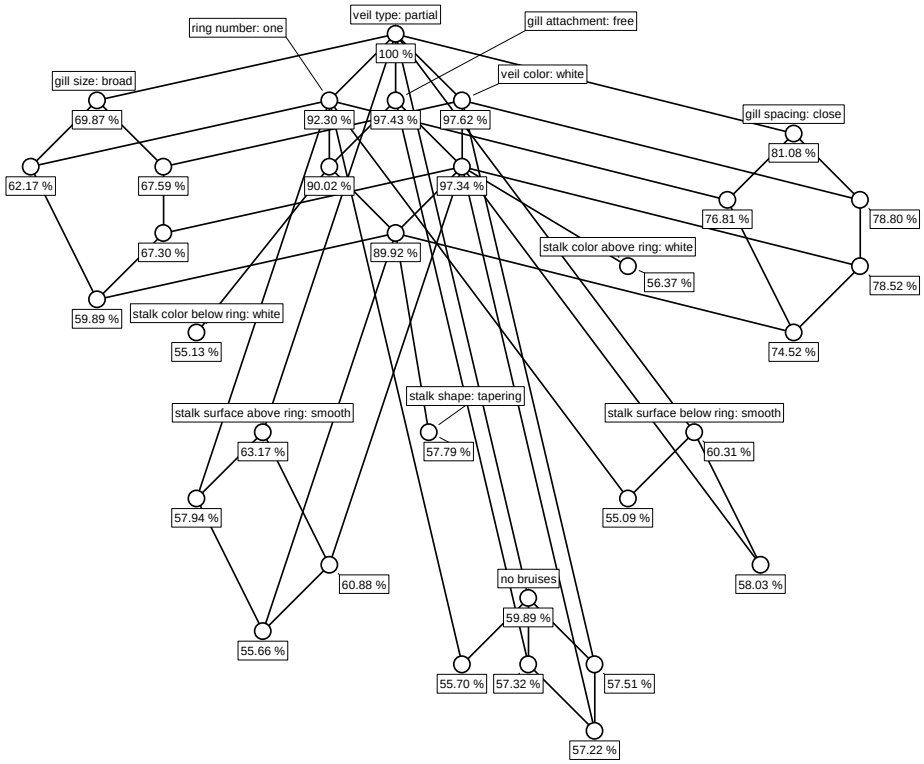

Fig. 3. Iceberg concept lattice of the mushroom database with minsupp = 55 %

Table 1. Number of frequent closed itemsets and frequent itemsets for the Mushrooms example

minsupp	# frequent closed itemsets	# frequent itemsets
85 %	7	16
70 %	12	32
55 %	32	116
0 %	32.086	2^{80}

4 Computing the Iceberg Concept Lattice with Titanic

For illustrating the principles underlying the algorithms for mining frequent (closed) itemsets using FCA, we sketch one representative called Titanic. For a more detailed discussion of the algorithm, we refer to [45].

Titanic is counting the support of so-called key itemsets (and of some candidates for key itemsets) only: A *key itemset* (or *minimal generator*) is every minimal itemset in an equivalence class of θ. Titanic makes use of the fact

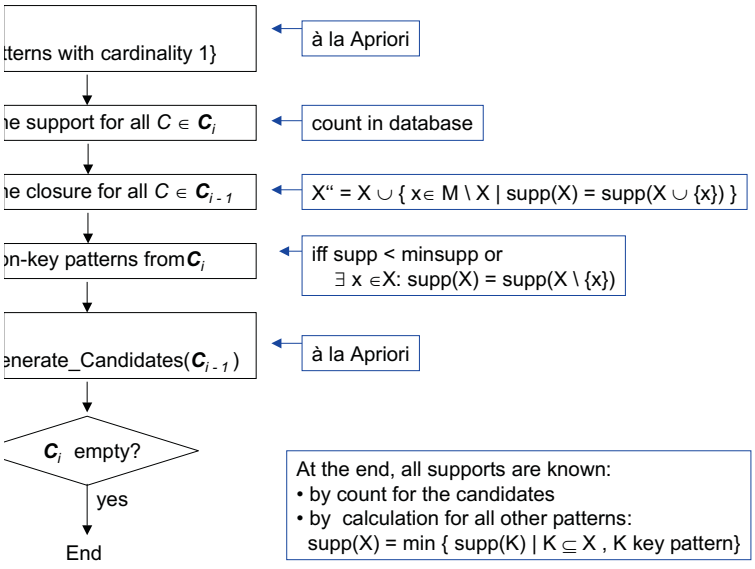

Fig. 4. The TITANIC algorithm

that the set of all key itemsets has the same property as the set of all frequent itemsets: it is an order ideal in the powerset of M. This means that each subset of a key itemset is a key itemset, and no superset of a non-key itemset is a key itemset. Thus we can reuse the pruning approach of Apriori for computing the supports of all frequent key itemsets. Once we have computed them, we have computed the support of at least one itemset in each equivalence class of θ, and we know the relation θ completely. Hence we can deduce the support of all frequent itemsets without accessing the database any more.

Figure 4 shows the principle of TITANIC. Its basic idea is as the original Apriori algorithm: At the ith iteration, we consider only itemsets with cardinality i (called i–*itemsets* for short), starting with $i = 1$ (step 1). In step 2, the support of all candidates is counted. For $i = 1$, the candidates are all 1–itemsets, later they are all i–itemsets which are potential key itemsets.

Once we know the support of all i–candidates, we have enough information to compute for all $(i-1)$–key itemsets their closure, i. e., the concept intent of their equivalence class. This is done in step 3, using the equation $X'' = X \cup \{x \in M \setminus X \mid \mathrm{supp}(X) = \mathrm{supp}(X \cup \{x\})$.

In step 4, all itemsets which are either not frequent or non-key are pruned. For the latter we use a characterization of key itemsets saying that a itemset is

a key itemset iff its support is different from the support of all its immediate subsets. In strongly correlated data, this additional condition helps pruning a significant number of itemsets.

At the end of each iteration, the candidates for the next iteration are generated in step 5. The generation procedure is basically the same as for Apriori: An $(i+1)$–itemset is a candidate iff all its i–subitemsets are key itemsets. As long as new candidates are generated, the next iteration starts. Otherwise the algorithm terminates.

It is important to note that — especially in strongly correlated data — the number of frequent key itemsets is small compared to the number of all frequent itemsets. Even more important, the cardinality of the largest frequent key itemset is normally smaller than the one of the largest frequent itemset. This means that the algorithm has to perform fewer iterations, and thus fewer scans of the database. This is especially important when the database is too large for main memory, as each disk access significantly increases computation time. A theoretical and experimental analysis of this behavior is given in [45], further experimental results are provided in [40].

5 Bases of Association Rules

One problem in mining association rules is the large number of rules which are usually returned. But in fact not all rules are necessary to present the information. Similar to the representation of all frequent itemsets by the frequent closed itemsets, one can represent all valid association rules by certain subsets, so-called *bases*. The computation of the bases does not require all frequent itemsets, but only the closed ones.

Here we will show by an example (taken from [46]), how these bases look like. A general overview is given in the next section.

We have already discussed how implications (i. e., association rules with 100 % confidence) can be read from the line diagram. The Luxenburger basis for approximate association rules (i. e., association rules with less than 100 % confidence) can also be visualized directly in the line diagram of an iceberg concept lattice. It makes use of results of [30] and contains only those rules $B_1 \rightarrow B_2$ where B_1 and B_2 are frequent concept intents and where the concept (B_1', B_1) is an immediate subconcept of (B_2', B_2). Hence there corresponds to each approximate rule in the Luxenburger base exactly one edge in the line diagram. Figure 5 visualizes all rules in the Luxenburger basis for minsupp = 70 % and minconf = 95 %. For instance, the rightmost arrow stands for the association rule {veil color: white, gill spacing: close} → {gill attachment: free}, which holds with a confidence of 99.6 %. Its support is the support of the concept the arrow is pointing to: 78.52 %, as shown in Figure 2. Edges without label indicate that the confidence of the rule is below the minimum confidence threshold. The visualization technique is described in more detail in [46]. In comparison with other visualization techniques for association rules (as for instance implemented in the IBM Intelligent Miner), the visualization of the Luxenburger basis within the iceberg concept

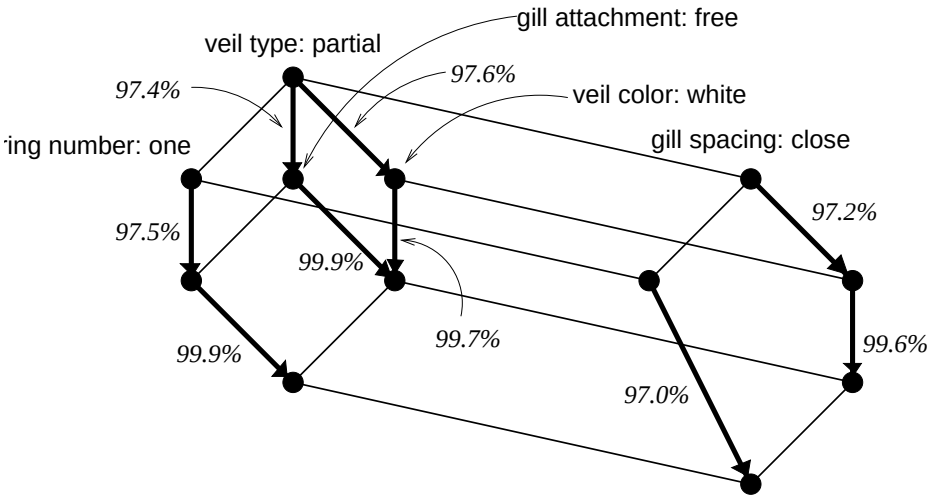

Fig. 5. Visualization of the Luxenburger basis for minsupp = 70 % and minconf= 95 %

lattice benefits of the smaller number of rules to be represented (without loss of information!), and of the presence of a 'reading direction' provided by the concept hierarchy. It might be worth to combine this advantage with the large effort spent in professional data mining tools to adher to human-computer interaction principles.

6 History and State of the Art in FCA-Based Association Rule Mining

Within the research area of Formal Concept Analysis, important results have been achieved for the association mining problem even before it has been stated. Because of their history, these results obviously did not consider all frequent itemsets, but rather the closed ones. At that point in time, the frequency was not used for pruning, i. e., implicitly a minimum frequency of 0 % was assumed. Early algorithms for computing the concept lattice / all closed itemsets were developed by Fay [19], Norris [33], Ganter [20], and Bordat [9].

Guigues and Duquenne [18] described a minimal set of *implications* (exact rules) from which all rules can be derived, the *Duquenne Guigues base* or *stem base*. In 1984, Ganter developed the Next Closure algorithm ([20], see also [21]) for its computation. Luxenburger was working on bases for *partial implications* (approximative rules) [30, 31]. From today's data mining perspective, all what these results are missing is the (rather straightforward) incorporation of the frequency threshold.

Five years after the first statement of the association mining problem in 1993, apparently the time was ripe for the discovery of its connection to Formal

Concept Analysis — it came up independently in several research groups at approximately the same moment [37, 52, 44]. Since then, the attention of FCA has largely increased within the data mining community, and many researchers joined the field.

Inspired by the observation that the frequent closed itemsets are sufficient to derive all frequent itemsets and their supports, the search for other *condensed / concise representations* began. The *key itemsets* as introduced in Section 4 serve the same purpose: from them (plus the minimal non-frequent key itemsets) one can derive the support of any frequent itemset. The key itemsets are also referred to as *minimal generators*, and nowadays more often as *free sets*.

A trend which goes beyond the notions discussed above aims at further condensing the set of free itemsets. The set of *disjunction-free sets* [15] / *disjunction-free generators* [27] is extending the set of free itemsets. A disjunction-free set is an itemset I where there do not exist i_1 and i_2 with $\text{supp}(I) = \text{supp}(I \setminus \{i_1\}) + \text{supp}(I \setminus \{i_2\}) - \text{supp}(I \setminus \{i_1, i_2\})$.[2] The frequent disjunction-free sets together with their support and the minimal non-frequent ones allow to compute the support of all frequent itemsets. This approach is further extended in [16] to *non-derivable itemsets*, where the previous equation is extended from the two elements i_1 and i_2 to an arbitrary number of elements.

A second trend is the analysis of *approximate representations*. One of them are *δ–free sets* [11], where an itemset is called δ–free if there are 'almost no dependencies' between its attributes.[3] They lead to *almost closures* [10], where an attribute m is in the almost–closure of an itemset X iff the supports of X and $X \cup \{m\}$ differ not more than a given threshold. (The usual closure is obtained with this threshold set to zero). The support of any frequent itemset can then be computed up to a certain error.

Amazingly, most of the recent work is focussing on condensed representations of the set of frequent itemsets rather than on condensed representations of the set of association rules. One reason may be that the former is the computationally expensive part, and thus considered as the more interesting problem. At least in the classical association rule setting, however, the analyst is confronted with a list of rules (and not a list of frequent itemsets), hence ultimately *this* list has to be kept small.

In the remainder of this section, we will present some results that have been obtained in FCA-based association rule mining since the late 1990ies. In the next subsection, we discuss algorithms for computing frequent closed and frequent key itemsets. While of course the development of algorithms for computing complete concept lattices goes on, they are out of the focus of the present survey. In the second subsection, we address condensed representations of association rules.

[2] Free sets are a special case with $i_1 = i_2$.

[3] A δ-free set is an itemset X such that all non-trivial rules $Y \rightarrow Z$ with $Y, Z \subseteq X$ have at least δ exceptions. Free sets are δ–free sets with $\delta = 0$, as there are no dependencies between their attributes. Because of this property which is known from database keys, free sets were originally called key sets.

6.1 Algorithms for Computing Frequent Closed / Key Itemsets

The first set of algorithms that were explicitly designed to compute *frequent closed* itemsets were Close [39], Apriori–Close [37] and A-Close [38]. Inspired by Apriori [2], all these algorithms traverse the database in a level-wise approach. A-Close follows a two-step approach. In the first step, it performs a level-wise search for identifying the frequent key sets; in the second step, their closures are determined. Apriori-Close computes simultaneously the frequent closed and all frequent itemsets. Close computes first all frequent closed itemsets by computing the closures of all minimal generators. In a second step, it derives all frequent (closed or non-closed) itemsets. These algorithms put the emphasis on datasets which cannot be kept completely in the main memory. Their major concern is thus to decrease the number of necessary accesses to the hard disk.

ChARM [52] is also following a bottom-up approach. Contrary to A-Close it performs a depth-first search in the powerset of itemsets. It stores the frequent itemsets in a prefix tree in main memory. The algorithm traverses both the itemset and transaction search spaces. As soon as a frequent itemset is generated, the set of corresponding transactions is compared with those of the other itemsets having the same parent in the prefix tree. If they are equal, then their nodes are merged in the prefix tree, as both generate the same closure. The follow-up CHARM–L [56] of ChARM also computes the covering relation of the iceberg concept lattice.

Closet [41] and Closet+ [50] also compute the frequent closed sets and their supports in a depth-first manner, storing the transactions in a tree structure, called FP–tree, inherited from the FP–Growth algorithm [24]. Each branch of the FP–tree represents the transactions having the same prefix. They use the same merging strategy as ChARM.

Mafia [14] is an algorithm which mainly is intended for computing all maximal frequent itemsets (which, by basic results from FCA, are all closee). It also has an option to compute all frequent closed itemsets. Another very early algorithm for computing the maximal frequent itemsets which makes use of FCA is `MaxClosure` [17], but it does not provide a significant speed-up compared to Apriori.

Titanic [45] — as introduced above in detail — continues the tradition of the Close family. It computes in a level-wise manner all frequent key sets, and in the same step their closures. Pascal [5] differs from Titanic in that it additionally produces all frequent itemsets. [4] discusses efficient data structures for the algorithms Pascal and Titanic.

There are also some approaches which combine the closeness constraint with other constraints. In [25], an algorithm is presented which determines, for given natural numbers k and $\min\_l$, the k most frequent closed itemsets which have no less than $\min\_l$ elements. In [8], only itemsets within a pre-defined order interval of the powerset of the set of items are considered; the closure operator is modified to this scenario (but is no longer a closure operator in the usual sense). Bamboo [51] is an algorithm for mining closed itemsets following a 'length-

decreasing support constraint', which states that large itemsets need a lower support to be considered relevant than small ones.

[49] provides an overview over some FCA based algorithms for mining association rules and discusses the data structures underlying the implementations. In particular, they address updates and compositions of the database / the formal context.

6.2 Bases of association rules

All existing proposals for bases of association rules distinguish between exact and approximate rules, and present separate bases for each of them. From the combination of both bases one can derive all frequent association rules — by some calculus which is not always stated explicitly by the authors.

As mentioned above, Duquenne/Guigues [18] and Luxenburger [30] presented bases of exact and approximate (frequent or not) association rules rather early. Their results were rediscovered from the data mining community [36, 53] and adapted to the frequency constraint [37, 44, 47, 55]. This approach is described in detail in [46], and sketched above in Section 5. Other approaches comprise the following:

The *min-max basis* [35, 40] is an alternative to the couple of Duquenne/Guigues and Luxenburger base. It is based both on free and on closed itemsets. The min-max base is also divided into two parts. For the exact rules, the set of all rules of the form $A \to B$ with B being a closed set and being A a generator of B with $A \neq B$ is a basis. For the approximate rules, the set of all rules of the form $A \to B$ with B being a closed set and being A a generator of some closed set strictly contained in B is a basis. As the min-max basis needs both the free and the closed sets, TITANIC is an appropriate algorithm for its computation.

Depending on the derivation calculus one allows, both the Luxenburger base (as discussed here) and the min-max base can still be reduced further. In fact, Luxenburger showed in [31] that it is sufficient to consider a set of rules which is a spanning tree of the concept lattice. E. g., the rule labeled by 99.7 % in Fig. 5 can be deduced from the rest. Since the deduction rules for this computation turn out to be rather complex for a knowledge discovery application, we decided in [46] to work with a simpler calculus.

In [22], the non-derivable itemsets [16] as described above are used to define a set of *non-derivable association rules*. This set is a lossless subset of the min-max basis, but the reduction comes with the cost of more complex formulae for computing support and confidence of the derivable rules by determining upper and lower bounds, based on set inclusion-exclusion principles.

Based on the disjunction-free generators mentioned above, [27] extends the notion of bases of association rules to rules where disjunction is allowed in the premise.

7 Conclusion

In this survey, we have shown that Formal Concept Analysis provides a strong theory for improving both performance and results of association rule mining algorithms. As the current discussion of 'condensed representations' and – more general – of other applications of Formal Concept Analysis within Knowledge Discovery (e. g., conceptual clustering or ontology learning) show, there remains still a huge potential for further exploitation of Formal Concept Analysis for data mining and knowledge discovery. On the other hand we expect that this interaction will also stimulate further results within Formal Concept Analysis.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD international conference on Management of Data (SIGMOD'93)*, pages 207–216. ACM Press, May 1993.
2. R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th international conference on Very Large Data Bases (VLDB'94)*, pages 478–499. Morgan Kaufmann, September 1994.
3. R. Agrawal and R. Srikant. Mining sequential patterns. In *Proceedings of the 11th International Conference on Data Engineering (ICDE'95)*, pages 3–14. IEEE Computer Society Press, March 1995.
4. Y. Bastide. *Data Mining: algorithmes par niveau, techniques d'implementation et applications*. PhD thesis, Université de Clermont-Ferrand II, 2000.
5. Y. Bastide, R. Taouil, N. Pasquier, G. Stumme, and L. Lakhal. Mining frequent patterns with counting inference. *SIGKDD Explorations, Special Issue on Scalable Algorithms*, 2(2):71–80, 2000.
6. S.D. Bay. The UCI KDD Archive. Technical report, University of California, Department of Information and Computer Science, Irvine, 99. http://kdd.ics.uci.edu.
7. R. J. Bayardo. Efficiently mining long patterns from databases. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of Data (SIGMOD'98)*, pages 85–93. ACM Press, June 1998.
8. Francesco Bonchi and Claudio Lucchese. On closed constrained frequent pattern mining. In *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM 2004)*, pages 35–42. IEEE Computer Society, 2004.
9. J. P. Bordat. Calcul pratique du treillis de galois d'une correspondance Galois. *Math. Sci. Hum.*, 96:31–47, 1986.
10. Jean-Francois Boulicaut and Artur Bykowski. Frequent closures as a concise representation for binary data mining. In *PADKK '00: Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Current Issues and New Applications*, pages 62–73, London, UK, 2000. Springer-Verlag.
11. Jean-Francois Boulicaut, Artur Bykowski, and Christophe Rigotti. Approximation of frequency queries by means of free-sets. In *PKDD '00: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 75–85, London, UK, 2000. Springer-Verlag.
12. S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets : Generalizing association rules to correlation. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of Data (SIGMOD'97)*, pages 265–276. ACM Press, May 1997.

13. S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proceedings of the 1997 ACM SIGMOD international conference on Management of Data (SIGMOD'97)*, pages 255–264. ACM Press, May 1997.
14. D. Burdick, M. Calimlim, and J. Gehrke. Mafia: A maximal frequent itemset algorithm for transactional databases. In *Proc. of the 17th Int. Conf. on Data Engineering*. IEEE Computer Society, 2001.
15. Artur Bykowski and Christophe Rigotti. A condensed representation to find frequent patterns. In *PODS '01: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 267–273, New York, NY, USA, 2001. ACM Press.
16. Toon Calders and Bart Goethals. Mining all non-derivable frequent itemsets. In *PKDD '02: Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 74–85, London, UK, 2002. Springer-Verlag.
17. D. Cristofor, L. Cristofor, and D.A. Simovici. Galois Connections and Data Mining. *Journal of Universal Computer Science*, 6(1):60–73, January 2000.
18. V. Duquenne and J.-L. Guigues. Famille minimale d'implications informatives résultant d'un tableau de données binaires. *Mathématiques et Sciences Humaines*, 24(95):5–18, 1986.
19. G. Fay. An algorithm for finite Galois connections. Technical report, Institute for Industrial Economy, Budapest, 1973.
20. Bernhard Ganter. Two basic algorithms in concept analysis. FB4–Preprint 831, TH Darmstadt, 1984.
21. Bernhard Ganter and Klaus Reuter. Finding all closed sets: a general approach. *Order*, 8:283–290, 1991.
22. Bart Goethals, Juhu Muhonen, and Hannu Toivonen. Mining non-derivable association rules. In *Proc. SIAM International Conference on Data Mining*, Newport Beach, CA, April 2005.
23. J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, Sept. 2000.
24. J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proc. ACM SIGMOD Int'l Conf. on Management of Data*, pages 1–12, May 2000.
25. Jiawei Han, Jianyong Wang, Ying Lu, and Petre Tzvetkov. Mining top-k frequent closed patterns without minimum support. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002)*, pages 211–218. IEEE Computer Society, 2002.
26. M. Kamber, J. Han, and Y. Chiang. Metarule-guided mining of multi-dimensional association rules using data cubes. In *Proc. of the 3rd KDD Int'l Conf.*, August 1997.
27. Marzena Kryszkiewicz. Concise representation of frequent patterns based on disjunction-free generators. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 305–312, Washington, DC, USA, 2001. IEEE Computer Society.
28. B. Lent, R. Agrawal, and R. Srikant. Discovering trends in text databases. In *Proceedings of the 3rd international conference on Knowledge Discovery and Data mining (KDD'97)*, pages 227–230. AAAI Press, August 1997.
29. D. Lin and M. Kedem. A new algorithm for discovering the maximum frequent set. In *Proceedings of the 6th Int'l Conf.on Extending Database Technology (EDBT)*, pages 105–119, March 1998.

30. M. Luxenburger. Implications partielles dans un contexte. *Mathématiques, Informatique et Sciences Humaines*, 29(113):35–55, 1991.
31. Michael Luxenburger. *Implikationen, Abhängigkeiten und Galois–Abbildungen.* PhD thesis, TH Darmstadt, 1993. Shaker Verlag, Aachen, 1993. In english language, beside the introduction).
32. H. Mannila. Methods and problems in data mining. In *Proceedings of the 6th biennial International Conference on Database Theory (ICDT'97)*, Lecture Notes in Computer Science, Vol. 1186, pages 41–55. Springer-Verlag, January 1997.
33. Eugene M. Norris. An algorithm for computing the maximal rectangles in a binary relation. *Rev. Roum. Math. Pures et Appl.*, 23(2):243–250, 1978.
34. J. S. Park, M.-S. Chen, and P. S. Yu. An efficient hash based algorithm for mining association rules. In *Proceedings of the 1995 ACM SIGMOD international conference on Management of Data (SIGMOD'95)*, pages 175–186. ACM Press, May 1995.
35. N. Pasquier. Extraction de bases pour les règles d'association à partir des itemsets fermés fréquents. In *Actes du 18ème congrès sur l'Informatique des Organisations et Systèmes d'Information et de Décision (INFORSID'2000)*, May 2000.
36. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Pruning closed itemset lattices for association rules. In *Actes des 14èmes journées Bases de Données Avancées (BDA'98)*, pages 177–196, Octobre 1998.
37. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Closed set based discovery of small covers for association rules. In *Actes des 15èmes journées Bases de Données Avancées (BDA'99)*, pages 361–381, Octobre 1999.
38. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *Proceedings of the 7th biennial International Conference on Database Theory (ICDT'99)*, Lecture Notes in Computer Science, Vol. 1540, pages 398–416. Springer-Verlag, January 1999.
39. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Efficient mining of association rules using closed itemset lattices. *Information Systems*, 24(1):25–46, 1999.
40. Nicolas Pasquier, Rafik Taouil, Yves Bastide, Gerd Stumme, and Lotfi Lakhal. Generating a condensed representation for association rules. *Journal of Intelligent Information Systems*, 24(1):29–60, 2005.
41. J. Pei, J. Han, and R. Mao. Closet: An efficient algorithm for mining frequent closed itemsets. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 21–30, 2000.
42. A. Savasere, E. Omiecinski, and S. Navathe. An efficient algorithm for mining association rules in larges databases. In *Proceedings of the 21st international conference on Very Large Data Bases (VLDB'95)*, pages 432–444. Morgan Kaufmann, September 1995.
43. C. Silverstein, S. Brin, and R. Motwani. Beyond market baskets : Generalizing association rules to dependence rules. *Data Mining and Knowledge Discovery*, 2(1):39–68, January 1998.
44. G. Stumme. Conceptual knowledge discovery with frequent concept lattices. FB4-Preprint 2043, TU Darmstadt, 1999.
45. G. Stumme, R. Taouil, Y. Bastide, N. Pasqier, and L. Lakhal. Computing iceberg concept lattices with titanic. *J. on Knowledge and Data Engineering*, 42(2):189–222, 2002.
46. G. Stumme, R. Taouil, Y. Bastide, N. Pasquier, and L. Lakhal. Intelligent structuring and reducing of association rules with formal concept analysis. In F. Baader, G. Brewker, and T. Eiter, editors, *KI 2001: Advances in Artificial Intelligence* Proc. KI 2001, volume 2174 of *LNAI*, pages 335–350. Springer, Heidelberg, 2001.

47. R. Taouil. *Algorithmique du treillis des fermés : application à l'analyse formelle de concepts et aux bases de données.* PhD thesis, Université de Clermont-Ferrand II, 2000.
48. H. Toivonen. *Discovery of frequent patterns in large data collection.* PhD thesis, University of Helsinki, 1996.
49. P. Valtchev, R. Missaoui, and R. Godin. Formal concept analysis for knowledge discovery and data mining: The new challenges. In Peter W. Eklund, editor, *Concept Lattices*, volume 2961 of *Lecture Notes in Computer Science*, pages 352–371. Springer, 2004.
50. Jianyong Wang, Jiawei Han, and Jian Pei. Closet+: searching for the best strategies for mining frequent closed itemsets. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 236–245, New York, NY, USA, 2003. ACM Press.
51. Jianyong Wang and George Karypis. Bamboo: Accelerating closed itemset mining by deeply pushing the length-decreasing support constraint. In Michael W. Berry, Umeshwar Dayal, Chandrika Kamath, and David B. Skillicorn, editors, *Proceedings of the Fourth SIAM International Conference on Data Mining*. SIAM, 2004.
52. M. J. Zaki and C.-J. Hsiao. Chaarm: An efficient algorithm for closed association rule mining. technical report 99–10. Technical report, Computer Science Dept., Rensselaer Polytechnic, October 1999.
53. M. J. Zaki and M. Ogihara. Theoretical foundations of association rules. In *DMKD'98 workshop on research issues in Data Mining and Knowledge Discovery*, pages 1–8. ACM Press, June 1998.
54. M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In *Proceedings of the 3rd international conference on Knowledge Discovery and Data mining (KDD'97)*, pages 283–286. AAAI Press, August 1997.
55. Mohammed J. Zaki. Generating non-redundant association rules. In *Proc. KDD 2000*, pages 34–43, 2000.
56. Mohammed J. Zaki and Ching-Jui Hsaio. Efficient algorithms for mining closed itemsets and their lattice structure. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):462–478, April 2005.