Towards the Tractable Discovery of Association Rules with Negations

Jean-François Boulicaut, Artur Bykowski, and Baptiste Jeudy

Institut National des Sciences Appliquées de Lyon, Laboratoire d'Ingénierie des Systèmes d'Information, Bâtiment 501, F-69621 Villeurbanne cedex, France.
{jfboulic|abykowsk|bjeudy}@lisi.insa-lyon.fr

Abstract. Frequent association rules (e.g., $A \wedge B \Rightarrow C$ to say that when properties A and B are true in a record then, C tends to be also true) have become a popular way to summarize huge datasets. The last 5 years, there has been a lot of research on association rule mining and more precisely, the tractable discovery of interesting rules among the frequent ones. We consider now the problem of mining association rules that may involve negations e.g., $A \wedge B \Rightarrow \neg C$ or $\neg A \wedge B \Rightarrow C$. Mining such rules is difficult and remains an open problem. We identify several possibilities for a tractable approach in practical cases. Among others, we discuss the active use of constraints. We propose a generic algorithm and discuss the use of constraints to mine the generalized sets from which rules with negations can be derived.

1 Introduction

The design of semiautomatic methods for locating interesting information in the masses of unanalyzed or underanalyzed data, the so-called data mining techniques, has become an important research area. Mining association rules [1] is a popular data mining technique that has been proved useful for real data analysis (see e.g., its application to basket analysis or [11] for alarm data analysis). During a typical association rule mining process, one first computes frequent patterns using (expensive) data mining algorithms. Then, one has to compute and/or query these collections for the needed post-processing phases (e.g., deriving rules and ranking them according to various objective measures of interestingness like confidence [2] or intensity of implication [8]).

A problem with such a process is that (a) the selection of interesting patterns has to be performed only on frequent patterns, and (b) standard association rules are not enough expressive for some applications. Mining all infrequent rules is known to be intractable, but is it interesting? Indeed, infrequent patterns can be considered as the result of noisy data. However, the frequency threshold that bounds the interest for an application might be too low from the computational complexity point of view. Looking for more expressive rules, we can consider various generalizations, e.g., the introduction of taxonomies on items [9], associations between multiple relations [6] or general boolean rules [12]. In this paper, we consider a special case of boolean rules, namely association rules with negations. We want to find associations

Proceedings of the Fourth Int. Conference on Flexible Query Answering Systems FQAS'00, Warsaw (PL), October 25-28, 2000. Physica-Verlag, Advances in Soft Computing. pp. 425-434. Physica-Verlag.

between positive attributes and negative ones. Mining frequent (generalized) rules turns to be intractable in practical cases, i.e., for frequency thresholds that enable the discovery of "standard" association rules. We would like to identify possibilities for a tractable approach to the computation of generalized (frequent) sets and their post-processing into generalized association rules. We believe that the inductive querying framework points out promising issues [10,5,7].

The contribution of this paper is as follows. First, we explain why a naive approach (i.e., the straightforward encoding of both positive and negative attributes and the use of standard algorithms like APRIORI) cannot be used. Then we consider how it is possible to derive some generalized rules using only the information about "positive" attributes. Finally, we consider inductive queries that return generalized sets (from which rules with negations can be derived) and discuss briefly their evaluation. Roughly speaking, this means that, given constraints on desired sets, one must identify which of them can be "pushed" efficiently into the discovery algorithm. Using results from [4], we consider relevant constraints for mining association rules with negations.

2 Inductive Databases and Inductive Queries

We consider the formalization of inductive databases [5] and the concept of inductive query in our context.

Definition 1 (schema and instance). The schema of an inductive database is a pair $\mathcal{R} = (\mathbf{R}, (\mathcal{L}, \mathcal{E}, \mathcal{V}))$, where $\mathbf{R}$ is a database schema, $\mathcal{L}$ is a countable collection of patterns, $\mathcal{V}$ is a set of *result values*, and $\mathcal{E}$ is the *evaluation function* that characterizes patterns. Given a database $\mathbf{r}$ over $\mathbf{R}$ and a pattern $\theta \in \mathcal{L}$, this function maps $(\mathbf{r}, \theta)$ to an element of $\mathcal{V}$. An instance $(\mathbf{r}, s)$ over $\mathcal{R}$ consists of a database $\mathbf{r}$ over the schema $\mathbf{R}$ and a subset $s \subseteq \mathcal{L}$.

A typical KDD process operates on both of the components of an inductive database. Queries that concerns only the pattern part, called hereafter *inductive queries*, specify mining tasks.

Definition 2 (inductive query). Given an inductive database instance $(\mathbf{r}, s)$ whose schema is $(\mathbf{R}, (\mathcal{L}, \mathcal{E}, \mathcal{V}))$, an inductive query is denoted as $\sigma_{\mathcal{C}}(s)$ and specifies the sentences from s that are interesting. $\mathcal{C}$ is a conjunction of constraints that must be fulfilled by the desired patterns. Checking some of the conjuncts may need for the evaluation of $\mathcal{E}$ on $\mathbf{r}$ and involve data scans.

Example 1. Assume the minable view is `trans(Tid,Item)` i.e., a typical schema of data for basket analysis. Figure 1 provides a toy dataset under a boolean matrix format. For instance, if `trans(2,A)` and `trans(2,C)` define the transaction 2, row 2 contains true for columns A and C and false for column B. It also means that, in row 2 of the "complemented" matrix, $\bar{A}$ (to denote $\neg A$) and $\bar{C}$ are false while $\bar{B}$ is true.

Definition 3 (set mining task). We look for generalized sets $X \in 2^{\texttt{Items}}$ where `Items` is the set of all the values of attribute `Item` and its negative counterpart. Let $\mathcal{F}(X, \mathbf{r})$, the frequency of X, denotes the percentage of transactions that involve each attribute in X. A set X is γ-frequent in $\mathbf{r}$ if $\mathcal{F}(X, \mathbf{r}) \geq \gamma$. This constraint is denoted by $\mathcal{C}_{freq}$. The pattern part of the associated inductive database is $(\mathcal{L}, \mathcal{E}, [0, 1])$ where $\mathcal{L} = 2^{\texttt{Items}}$ and $\mathcal{E}$ returns the set frequency. Mining generalized sets leads to the evaluation of $\sigma_{\mathcal{C}}(2^{\texttt{Items}})$ where $\mathcal{C}$ is a conjunction of constraints. Mining frequent generalized sets means that $\mathcal{C}$ contains $\mathcal{C}_{freq}$.

Example 2. Given data from Fig. 1, $\texttt{Items}=\{A, B, C, \bar{A}, \bar{B}, \bar{C}\}$. If $\mathcal{C}_{freq}(X) = \mathcal{F}(X, \mathbf{r}) \geq 0.5$, the query $\sigma_{\mathcal{C}_{freq}}(2^{\texttt{Items}})$ returns $\{A, B, C, \bar{C}, AB, AC, B\bar{C}\}$. Assume that $\mathcal{C}_{size}$ (resp., $\mathcal{C}_{miss}$) denotes the constraint $|X| \leq 2$ (resp., $\{A, \bar{B}\} \cap X = \emptyset$) for a set X. $\sigma_{\mathcal{C}_{size} \wedge \mathcal{C}_{miss}}(2^{\texttt{Items}})$ returns $\{B, C, \bar{A}, \bar{C}, BC, B\bar{A}, B\bar{C}, C\bar{A}, C\bar{C}, \bar{A}\bar{C}\}$. $\sigma_{\mathcal{C}_{freq} \wedge \mathcal{C}_{size} \wedge \mathcal{C}_{miss}}(2^{\texttt{Items}})$ returns $\{B, C, \bar{C}, B\bar{C}\}$.

$\mathbf{r} =$

	A	B	C
1	1	1	1
2	1	0	1
3	0	1	0
4	1	1	0

Its "complement" is

	$\neg A$	$\neg B$	$\neg C$
1	0	0	0
2	0	1	0
3	1	0	1
4	0	0	1

Fig. 1. A binary dataset $\mathbf{r}$

Definition 4 (standard association rule mining task). An association rule is an expression $X \Rightarrow Y$ where $X \subseteq \texttt{Items}$ and $Y \in \texttt{Items} \setminus X$. The typical "behavior" of these rules in an instance $\mathbf{r}$ is evaluated by means of two interestingness measures, namely the support and the confidence. The support of $X \Rightarrow Y$ in $\mathbf{r}$ is equal to $\mathcal{F}(X \cup \{Y\}, \mathbf{r})$ and its confidence is equal to its support divided by $\mathcal{F}(X, \mathbf{r})$. In terms of an inductive schema, $\mathcal{V}$ is [0,1] $\times$ [0,1] and $\mathcal{E}$ provides the support and the confidence. The standard association rule mining task concerns the discovery of the so-called *strong rules* whose support and confidence are greater or equal to user-given thresholds, resp., γ and ϕ. It corresponds to inductive queries on the language of rules whose constraint contains at least this selection criterion.

Example 3. With γ=0.5 and ϕ=0.9, $\bar{C} \Rightarrow B$ is discovered in data of Fig. 1 while $B \Rightarrow \bar{C}$ is not (its confidence is too low).

3 Mining Association Rules with Negations

Notations. When mining association rules, the expensive step concerns the computation of the (frequent) sets from which the rules are derived. We say that a set $T \in 2^{\texttt{Items}}$ satisfies a constraint $\mathcal{C}$ if $\mathcal{C}(T)$ evaluates to true. If $\mathcal{C}$ is a constraint, let $SAT_{\mathcal{C}}(\texttt{Items})$ denote the collection $\{\texttt{T} \subseteq \texttt{Items}, \texttt{T}$

satisfies $\mathcal{C}$}. In other terms, $SAT_{\mathcal{C}}(\texttt{Items})$ is the result of the inductive query $\sigma_{\mathcal{C}}(2^{\texttt{Items}})$. Most of the tractable queries involve $\mathcal{C}_{freq}$: given the threshold γ, $\mathcal{C}_{freq}(T) = \mathcal{F}(T, \mathbf{r}) \geq \gamma$. Let $\texttt{Items}^+$ (resp. $\texttt{Items}^-$) denote the set of positive (resp. negative) attributes in Items. A subset of $\texttt{Items}^+$ is called a positive set. $SAT_{\mathcal{C}_{freq}}(\texttt{Items})$ is the collection of frequent (generalized) sets and we assume that they are stored, with their frequencies in FS_γ. FS^+_γ denotes the collection of frequent positive sets with their frequencies. Notice that we need the frequency of every frequent set since our goal is to derive rules. Given a set $T \in 2^{\texttt{Items}}$, let $\texttt{NNT}(T)$ denote the number of negative elements in T, $\texttt{PT}(T)$ denote the set of positive elements in T, $\texttt{P}(T)$ denote the set of elements in T all turned to their positive counterparts. We often use a string notation for sets e.g., $\bar{A}\bar{B}C$ for $\{\bar{A}, \bar{B}, C\}$.

Example 4. $\texttt{NNT}(\bar{A}\bar{B}C) = 2$, $\texttt{PT}(\bar{A}\bar{B}C) = \{C\}$, $\texttt{P}(\bar{A}\bar{B}C) = ABC$.

3.1 The "Standard" Apriori Algorithm

We consider an abstract definition of the Apriori algorithm [2]. It takes a dataset $\mathbf{r}$ and, given the thresholds γ (frequency) and ϕ (confidence), outputs every strong association rule.

1. $C_1 :=$ `set-of-all-singletons(Items)`
2. k := 1
3. **while** $C_k \neq \emptyset$ **do**
4. Phase 1 - frequency constraint is checked - it needs a data scan
   $\mathcal{L}_k := SAT_{\mathcal{C}_{freq}}(C_k)$
5. Phase 2 - candidate generation for level k+1
   $C^g_{k+1} :=$ `generate`$(\mathcal{L}_k)$
6. Phase 3 - candidate safe pruning
   $C_{k+1} :=$ `safe-pruning-on`(C^g_{k+1})
7. $k := k + 1$
8. $FS_\gamma := \bigcup_{i=1}^{k-1} \mathcal{L}_i$
9. Phase 4 - rule generation
   for each $X \in FS_\gamma$ **do**
10. **for each** $Y \in X$ **do**
11. `test-for-output`$(X \setminus \{Y\} \Rightarrow \{Y\})$

In [2], `generate`$(\mathcal{L}_k)$ provides the candidates by fusion of two elements from $\mathcal{L}_k$ that share the same $k-1$ first elements (in lexicographic order), and phase 2 and 3 are merged. `safe-pruning-on`(C^g_{k+1}) just eliminates the candidates for which a subset of length k is not frequent. In step 8, all the frequent sets and their frequencies are stored in FS_γ. In [2], `test-for-output` consists of testing if the confidence of the rule is high enough w.r.t. ϕ.

Example 5. Considering the data from Fig. 1 and the thresholds $\gamma = 0.5$ and $\phi = 1$, Apriori outputs only $C \Rightarrow A$ and $\bar{C} \Rightarrow B$.

APRIORI can work fine for huge sparse datasets. Its practical time complexity is $\mathcal{O}(n \times nc)$ where n is the number of transactions and nc the number of candidates (i.e., the size of FS_γ plus the size of its negative border $\mathcal{B}d^-(FS_\gamma)$ [13]). $\mathcal{B}d^-(FS_\gamma)$ is the collection of infrequent sets whose every subset is frequent. Deriving rules is cheap when checking the selection criterion needs only FS_γ (e.g., confidence evaluation for frequent rules).

3.2 Problems with Negations... and Potential Solutions

A Naive Approach Assume the use of APRIORI on a dataset that has been "complemented". Considering the boolean matrix representation, it means that for each column A, one adds a column $\bar{A}$. In a row, i.e., for a given transaction, the value of $\bar{A}$ is the boolean negation of the value for A.

The problem with this approach is that for reasonable frequency thresholds, the number of frequent sets explodes. For instance, assume you have 100 positive attributes with a maximum attribute frequency of 0.1 and a frequency threshold $\gamma = 0.05$. It turns out that every set of negative attributes up to size 9 is frequent. In that case, it leads to more than $\binom{100}{9} > 10^{12}$ frequent sets [12]. In fact, this encoding also introduces a high correlation in the dataset: many association rules with high confidence hold in it. In practice, it means that we have to take higher frequency thresholds, possibly leading to uninteresting mining phases. Furthermore, even if the extraction of the frequent sets remains tractable, most of them will involve only negative attributes and, most of the derived rules (with high confidence) will concern only negative attributes as well. This is unfortunate for many application domains. Notice also that the positive part of the data can be already highly-correlated so that APRIORI can even not tackle the computation of FS^+_γ i.e., subsets of $\texttt{Items}^+$ that are frequent.

"Approximation" of Association Rules with Negations Assume in this subsection that we compute FS^+_γ i.e., the collection of γ-frequent sets for the positive attributes only. We already know that this problem is easier to solve. Using only that (positive) information, it remains possible to mine some association rules with negations. Proofs of these theorems are not given due to space limitation.

Theorem 1. *The support of a generalized set T can be computed using the collection FS^+_γ if $\mathtt{P}(T) \in FS^+_\gamma$ and is equal to:*

$$\mathcal{F}(T, \mathbf{r}) = \sum_{\mathtt{PT}(T) \subseteq X \subseteq T} (-1)^{\mathtt{NNT}(X)} \mathcal{F}(\mathtt{P}(X), \mathbf{r}) \; .$$

Example 6. Considering data from Fig. 1, $FS^+_{0.5} = \{A, B, C, AB, AC\}$[1]. Since $\mathtt{P}(A\bar{B}) = AB \in FS^+_{0.5}$, $\mathcal{F}(A\bar{B}, \mathbf{r}) = \mathcal{F}(A, \mathbf{r}) - \mathcal{F}(AB, \mathbf{r})$ is known exactly

[1] Notice also that the frequency in $\mathbf{r}$ is associated to each frequent set in $FS^+_{0.5}$

$(= 0.25)$. $\mathtt{P}(AB\bar{C}) = ABC \notin FS^+_{0.5}$ and $\mathcal{F}(AB\bar{C}, \mathbf{r}) = \mathcal{F}(AB, \mathbf{r}) - \mathcal{F}(ABC, \mathbf{r})$ can not be evaluated exactly.

Theorem 2. *The support of all subsets of a set T can be derived from the collection FS^+_γ if $\mathtt{P}(T) \in FS^+_\gamma$.*

The support of some generalized sets can be computed exactly from the positive frequent sets only. The number of terms of the sum is growing exponentially with the number of negative elements in the set. However, it can not be higher than $|FS^+_\gamma|$, which is the number of all possible terms of the sum. Therefore, the computation of the support of a generalized set remains tractable when the computation of FS^+_γ is tractable.

Consider now the generation of rules from a frequent set $X \in FS^+_\gamma$. Let A_X be a generalized set such that $\mathtt{P}(A_X) = X$. There are $2^{|X|}$ such generalized sets from which the Phase 4 of the Apriori algorithm can be performed. This way, we can generate a lot of generalized rules, but the collection is not complete w.r.t. the support and confidence criteria: there is no guarantee that it computes all the strong generalized rules. Our experiments have shown that it can even be far from the intended result.

A special case of this method can be performed for free during the standard association rule discovery. It concerns a restricted form of generalized rule: rules with a set included in $\mathtt{Items}^+$ at the left–hand side and an attribute from $\mathtt{Items}$ (positive or negative) at the right–hand side. After the computation of FS^+_γ, for each frequent set $X \in FS^+_\gamma$ and for each $Y \in X$ (all of them are frequent, too), it is possible to test the confidence of $X \setminus \{Y\} \Rightarrow \{Y\}$ and $X \setminus \{Y\} \Rightarrow \{\bar{Y}\}$ as well. Testing the confidence of the second rule needs for the evaluation of $1 - (\mathcal{F}(X, \mathbf{r})/\mathcal{F}(X \setminus \{Y\}, \mathbf{r})) \geq \phi$ i.e., $\mathcal{F}(X, \mathbf{r})/\mathcal{F}(X \setminus \{Y\}, \mathbf{r}) \leq 1 - \phi$. However, even limited to this form, some strong rules might be missed by the method. To be complete, we need to know the frequency of all generalized frequent sets and we saw that is is not realistic. However, it is possible to trade precision against completeness.

Consider now the possibility to compute imprecise interestingness measures for generalized rules by using only positive information. The idea is to substitute to the unknown terms an interval that bounds the possible values for the support and the confidence. Thus, it gives rise to an incertitude to the values of these measures. Let us consider a second dataset in Fig. 2.

Example 7. Consider the rule $A\bar{E} \Rightarrow B$. $\text{support}(A\bar{E} \Rightarrow B, \mathbf{r}) = \mathcal{F}(AB, \mathbf{r}) - \mathcal{F}(ABE, \mathbf{r})$ and $\text{confidence}(A\bar{E} \Rightarrow B, \mathbf{r}) = \text{support}(A\bar{E} \Rightarrow B, \mathbf{r})/(\mathcal{F}(A, \mathbf{r}) - \mathcal{F}(AE, \mathbf{r}))$. $\mathcal{F}(ABE, \mathbf{r})$ is not available (because ABE is not in $FS_{4/9}$ neither in $\mathcal{B}d^-(FS_{4/9})$) but it can be estimated within the interval $[0, 3/9]$: $\text{support}(A\bar{E} \Rightarrow B, \mathbf{r}) \in [1/9, 4/9]$ and $\text{confidence}(A\bar{E} \Rightarrow B, \mathbf{r}) \in [1/4, 1]$.

This method has been proposed in [12]. It gives rise to several problems. First, a rule can be interesting (w.r.t. interestingness criteria), uninteresting or unresolved. The last case arises when intervals for support and/or confidence cross minimum support and/or confidence thresholds. It happens

$\mathbf{r} =$

	A	B	C	D	E
1	1	1	1	0	0
2	0	0	1	1	1
3	0	1	0	0	1
4	1	1	1	0	1
5	1	1	1	0	0
6	1	1	0	1	0
7	1	0	0	0	1
8	0	1	1	0	0
9	1	0	0	1	0

Set	Frequency
$\{A\}$	6/9
$\{B\}$	6/9
$\{C\}$	5/9
$\{D\}$	3/9
$\{E\}$	4/9
$\{A, B\}$	4/9
$\{A, C\}$	4/9
$\{A, E\}$	2/9
$\{B, C\}$	4/9
$\{B, E\}$	2/9
$\{C, E\}$	1/9
$\{A, B, C\}$	3/9

Fig. 2. A dataset and the frequency for each set $\in FS^+_{4/9} \cup \mathcal{B}d^-(FS^+_{4/9})$

in Ex. 7. By substituting an interval of possible values to unknown terms in the support formula of a generalized set, some sets can not be classified as frequent or infrequent. Hence, for a complete strong rule generation in the worst case, one has to enumerate every generalized set that is not known as infrequent (i.e., that is frequent or unresolved). If we introduce "large" intervals, it gives rise to a huge amount of unresolved sets. Lot of them might be infrequent, but it is not possible to identify which ones.

Example 8. Continuing Ex. 7, from $FS^+_{4/9} \cup \mathcal{B}d^-(FS^+_{4/9})$, we infer 14 frequent generalized sets, 19 unresolved ones, and 210 infrequent ones. On this toy example the number of unresolved generalized sets is above the number of frequent generalized sets. This would be worse if we had $FS^+_{4/9}$ only, because these numbers become respectively 11, 48 and 184.

Problems related to incertitude on set frequency amplify with the computation of the confidence or other interestingness measures. Providing better estimations is important and worthwile.

Using Constraints A third direction of research is the effective use of constraints during the set mining task. It concerns the effective computation of $SAT_{\mathcal{C}}$ where $\mathcal{C}$ is a conjunction of atomic constraints that specify the interesting sets from which "interesting" rules are to be derived. We saw that APRIORI uses the constraint $\mathcal{C}_{freq}$ to prune the search space. Beside that, it is possible to have a "generate and test" approach for the other constraints (first generate all frequent generalized sets and then test other constraints on them) but this is clearly intractable in many cases. The solution might come from the study of constraint–based discovery of frequent sets [14,4].

4 A Constraint–Based Approach

In this section, we apply part of the work presented in [4] to our problem. A simple observation is that the APRIORI pruning is not only applicable to the frequency constraint but also to all anti-monotone constraints.

Definition 5 (anti-monotonicity). A constraint $\mathcal{C}$ is *anti-monotone* if and only if for all sets S, S': S' $\subseteq$ S $\wedge$ S satisfies $\mathcal{C}$ $\Rightarrow$ S' satisfies $\mathcal{C}$

Example 9. A systematic study of anti-monotone constraints on sets is in [14]. Continuing our running example, $\{A, B, C, D\} \supset \mathtt{S}$ and $\mathtt{S} \cap \{A, B, C\} = \emptyset$ are anti-monotone constraints on $\mathtt{S}$. Indeed, $\mathcal{C}_{freq}$ is anti-monotone. Another interesting anti-monotone constraint is $|S \cap \mathtt{Items}^-| \leq n$ which states that every itemset can contain up to n negative attributes.

Conjunctions of anti-monotone constraints are anti-monotone and it is easy to prove that pushing an anti-monotone constraint inside the search space exploration leads to less computations. What is challenging is the possibility of pushing other constraints. This is difficult because the generation and pruning steps must be rewritten to be complete. Moreover, pushing non anti-monotone constraints can decrease the performance of the whole process [4].

4.1 A Generic Algorithm $\mathcal{G}$

We consider a generalization of APRIORI to emphasize the potential for optimization. Assume the constraint $\mathcal{C}$ can be written as $\mathcal{C}_1 \wedge \mathcal{C}_2 \wedge \mathcal{C}_3 \wedge \mathcal{C}_4 \wedge \mathcal{C}_g \wedge \mathcal{C}_{dbs}$ where $\mathcal{C}_{dbs}$ denotes constraints that need a database scan at checking time and $\mathcal{C}_g$ denotes constraint checked during the generation step.

Algorithm $\mathcal{G}$: computation of $\sigma_{\mathcal{C}}(2^{\mathtt{Items}})$

1. $k := 1$; $\mathcal{L}_0 := \{\emptyset\}$
2. **repeat**
3. $C_k^g := \mathtt{generate}(\mathcal{L}_{k-1})$ # Candidate generation for level k and Test $\mathcal{C}_g$
4. $C_k^1 := C_k^g \cap SAT_{\mathcal{C}_1}$ # Test $\mathcal{C}_1$
5. $C_k^2 := \mathtt{prune}(C_k^1)$ # Safe pruning using anti-monotone constraints
6. $C_k^3 := C_k^2 \cap SAT_{\mathcal{C}_2}$ # Test $\mathcal{C}_2$
7. $C_k^4 := C_k^3 \cap SAT_{\mathcal{C}_{dbs}}$ #Test $\mathcal{C}_{dbs}$
8. $\mathcal{L}_k := C_k^4 \cap SAT_{\mathcal{C}_3}$ # Test $\mathcal{C}_3$
9. $k := k + 1$
9. **until** there is no more candidates
10. **return** $\bigcup_{i=1}^{k-1} \mathcal{L}_i \cap SAT_{\mathcal{C}_4}$ # Test $\mathcal{C}_4$

One problem is to choose how to split $\mathcal{C}$ into $\mathcal{C}_1 \wedge \mathcal{C}_2 \wedge \mathcal{C}_3 \wedge \mathcal{C}_4 \wedge \mathcal{C}_g \wedge \mathcal{C}_{dbs}$ (some of these can be the "true" constraint that always evaluates to true), i.e., at what step of the algorithm should a constraint be tested. First, a constraint can be pushed at the candidate generation step (step 3) if it is possible to have an optimized generation procedure (e.g., with the constraint $\mathcal{C}(S) = A \in S$). Second, it is possible to test constraints on steps 4, 6, 7, 8 and 10. Constraint checking that needs database scans, e.g., for $\mathcal{C}_{freq}$, is performed as step 7. Step 10 corresponds to a "generate and test approach". Depending on these choices, the generation and pruning steps might be rewritten to ensure completeness [4]. The kind of data we have to process when mining generalized sets is dense and highly-correlated. The CLOSE algorithm [15,3] makes the frequent set discovery tractable in such difficult cases. However,

the original CLOSE algorithm can only find frequent sets and is not designed to mine other constrained itemsets. We have revisited this algorithm in [4]. In fact, the optimization mechanism in CLOSE can be formalized as a new pruning criterion due to a new anti-monotone constraint.

Definition 6 (anti-monotone constraint $\mathcal{C}_{Close}$). $\mathcal{C}_{Close}(S) = S' \subset S \Rightarrow S \not\subseteq \texttt{closure}(S')$ where closure(S) is the maximal (for set inclusion) superset of S which has the same frequency as S.

This anti-monotone constraint gives rise to a new safe pruning criterion (besides the APRIORI trick based on frequency testing). An important property of the CLOSE algorithm is its completeness, which means that it is possible to find $SAT_{\mathcal{C}_{freq}}$ knowing $SAT_{\mathcal{C}_{Close} \wedge \mathcal{C}_{freq}}$[2].

Example 10. Let us find $\texttt{closure}(BC)$ on example of Fig. 2. Items B and C are simultaneously present in transactions 1, 4, 5 and 8. We can see that the maximal set of attributes ($\in \texttt{Items}$) that are true within these transactions is $\{B, C, \bar{D}\}$. Thus $\texttt{closure}(BC) = BC\bar{D}$. By the definition of closures, $\mathcal{F}(BC, \mathbf{r}) = \mathcal{F}(BC\bar{D})$ $(= 4/9)$. Since $\texttt{closure}(B) = B$ and $\texttt{closure}(C) = C$, $\mathcal{C}_{Close}(\texttt{BC})$ is true.

Considering the CLOSE algorithm as an instance of our generic algorithm $\mathcal{G}$ (with the constraint $\mathcal{C}_{Close}$) allow one to take advantage of CLOSE's improvements over APRIORI together with the ability to "push" constraints. To keep the completeness of CLOSE, [4] shows that algorithm $\mathcal{G}$ can be used to search for sets which satisfy $\mathcal{C} = \mathcal{C}_{Close} \wedge \mathcal{C}_{am} \wedge \mathcal{C}_m$ where $\mathcal{C}_{am}$ is anti-monotone (i.e., it can contain $\mathcal{C}_{freq}$) and $\mathcal{C}_m$ is a motone constraint.

Definition 7 (monotone constraint). A monotone constraint $\mathcal{C}_m$ is the negation of an anti-monotone constraint. If $\mathcal{C}_m$ is monotone, $\neg\, \mathcal{C}_m$ is anti-monotone: $\mathcal{C}_m(S)$ is true $\Rightarrow \forall S' \supset S$, $\mathcal{C}_m(S')$ is true.

Example 11. Continuing our running example, $\{A, B, C, D\} \subset S$ and $S \cap \{A, B, C\} \neq \emptyset$ are monotone constraints on S. An interesting case of a monotone constraint, denoted as $\mathcal{C}_{alnp}$, is $|S \cap \texttt{Items}^+| \geq n$. This constraint states that every set has to contain at least n positive attributes.

5 Conclusion

We discussed the possibility to derive association rules with negations using only the information about positive attributes. It gives rise to approximations of rule interestingness measures but has to be considered as a valuable direction of research. Then, we considered how given constraints on the desired rules can be "pushed" efficiently into a set mining algorithm. Using results from [4], we proposed useful constraints for mining association rules

[2] It is possible because CLOSE outputs the closures of the sets in $SAT_{\mathcal{C}_{Close} \wedge \mathcal{C}_{freq}}$

with negations, namely anti-monotone constraints (e.g., frequency constraint or constraint based on closures) and monotone constraints (e.g., ensuring the presence of positive attributes in the mined sets). This is an ongoing research and experimental validations on real datasets have been performed with the `min-ex` system [3] and can be obtained from the authors. This system already implements the use of $\mathcal{C}_{freq}$, $\mathcal{C}_{Close}$ and $\mathcal{C}_{alnp}$ (cf. Ex. 11). We considered constraints on generalized sets and now, this approach has to be extended to rules.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining Association Rules Between Sets of Items in Large Databases. In *Proc. ACM SIGMOD'93*, 207–216, 1993.
2. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo. Fast Discovery of Association Rules. In *Advances in Knowledge Discovery and Data Mining*, 307–328. AAAI Press, 1996.
3. J.-F. Boulicaut and A. Bykowski. Frequent Closures as a Concise Representation for Binary Data Mining. In *Proc. PAKDD'00*, *LNAI* 1805, 62–73, Kyoto, JP, 2000. Springer.
4. J.-F. Boulicaut and B. Jeudy. Using Constraints During Set Mining: Should We Prune or not? In *Proc. BDA'00*, Blois, F, 2000. INRIA. to appear.
5. J.-F. Boulicaut, M. Klemettinen, and H. Mannila. Querying Inductive Databases: A Case Study on the MINE RULE Operator. In *Proc. PKDD'98*, *LNAI* 1510, 194–202, Nantes, F, 1998. Springer.
6. L. Dehaspe and L. D. Raedt. Mining Association Rules in Multiple Relations. In *Proc. ILP'97*, *LNAI* 1297, 125–132. 1997. Springer.
7. F. Giannotti and G. Manco. Querying Inductive Databases via Logic–Based User Defined Aggregates. In *Proc. PKDD'99*, *LNAI* 1704, 125–135, Praha, CZ, 1999. Springer.
8. S. Guillaume, F. Guillet, and J. Philippé. Improving the Discovery of Association Rules with Intensity of Implication (short paper). In *Proc. PKDD'98*, *LNAI* 1510, 318–327, Nantes, F, 1998. Springer.
9. J. Han and Y. Fu. Discovery of Multiple–Level Association Rules from Large Databases. In *Proc. VLDB'95*, 420–431, Zürich, CH, 1995.
10. T. Imielinski and H. Mannila. A Database Perspective on Knowledge Discovery. *Communications of the ACM*, 39(11):58–64, Nov. 1996.
11. M. Klemettinen. *Rule Discovery from Telecommunication Network Alarm Databases*. PhD thesis, Dept. of Comp. Sc., Univ. of Helsinki, Jan. 1999.
12. H. Mannila and H. Toivonen. Multiple Uses of Frequent Sets and Condensed Representations. In *Proc. KDD'96*, 189–194, Portland, USA, 1996.
13. H. Mannila and H. Toivonen. Levelwise Search and Borders of Theories in Knowledge Discovery. *Data Mining and Knowledge Discovery*, 1(3):241–258, 1997.
14. R. Ng, L. V. Lakshmanan, J. Han, and A. Pang. Exploratory Mining and Pruning Optimization of Constrained Association Rules. In *Proc. ACM SIGMOD'98*, 13–24, Seattle, USA, 1998.
15. 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.