

Correlated Itemset Mining in ROC Space: A Constraint Programming Approach

Siegfried Nijssen

Tias Guns

Luc De Raedt

K.U.Leuven, Celestijnenlaan 200A, Leuven, Belgium
{siegfried.nijssen,tias.guns,luc.deraedt}@cs.kuleuven.be

ABSTRACT

Correlated or discriminative pattern mining is concerned with finding the highest scoring patterns w.r.t. a correlation measure (such as information gain). By reinterpreting correlation measures in ROC space and formulating correlated itemset mining as a constraint programming problem, we obtain new theoretical insights with practical benefits. More specifically, we contribute 1) an improved bound for correlated itemset miners, 2) a novel iterative pruning algorithm to exploit the bound, and 3) an adaptation of this algorithm to mine all itemsets on the convex hull in ROC space. The algorithm does not depend on a minimal frequency threshold and is shown to outperform several alternative approaches by orders of magnitude, both in runtime and in memory requirements.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database applications—*Data Mining*; F.4.1 [Mathematical Logic and Formal Languages]: Mathematical Logic—*Logic and Constraint Programming*

General Terms

Algorithms, Theory

Keywords

Itemset Mining, Constraint Programming, ROC Analysis

1. INTRODUCTION

Correlated pattern mining is amongst the most popular data mining tasks. As opposed to frequent itemset mining, correlated pattern mining involves transactions that belong to two different classes and the task is to find those patterns that are correlated with the class attribute, that is, those patterns that are indicators of one of the two classes. In this paper, we formalize this task as that of finding the patterns

that score high w.r.t. a correlation measure, such as χ^2 , information gain, etc., without applying a support threshold.

A large number of publications is concerned with correlated pattern mining [21, 24, 22], but the problem is also known under the names of interesting itemset mining [3], contrast set mining [1], emerging itemset mining [12], subgroup discovery [26, 18, 16] and discriminative itemset mining [8, 9, 27, 13]. Further publications have extended these settings to structured domains, such as graphs, trees and sequences [5, 17, 6, 30]. The popularity of correlated patterns is in part due to their use for classification, where they have been used both as classification rules [30, 5, 9, 27, 13] and as features for building classifiers [6, 11]. The key difference between traditional rule learning and correlated pattern mining approaches is that the former are typically heuristic and the latter provide guarantees concerning completeness and optimality of the computed solutions.

We revisit the problem of correlated pattern mining using the principles of *ROC analysis* and *Constraint Programming* (CP). First, by reformulating correlation measures in PN space (a rescaling of ROC space common in ROC analysis), building on [22, 15], we contribute a theoretical framework that allows a better understanding and comparison of different bounds used for pruning in correlated pattern mining. More specifically, we investigate various n -support bounds that are used for pruning in correlated pattern mining; these are bounds that are based on n different support values. We show that the 2-support bound of [21] is tighter than the 1-support bound proposed by [8], and also show how the 2-support bound can be generalized into a novel 4-support bound, which allows for extra pruning.

Secondly, we formulate the task of correlated itemset mining as a constraint programming problem (cf. [10]). The main argument for this formalisation is that it allows us to incorporate additional constraints in the mining process in an easy and flexible manner, among which closure constraints. This extends earlier results in the application of constraint programming to data mining, by allowing to deal with optimization problems and correlation measures. The resulting approach is incorporated in both an off-the shelf constraint programming solver and a specialized itemset miner. Employing the 4-support bound, both implementations are shown to be highly efficient and orders of magnitude faster than the state-of-the-art systems of [21, 9].

Finally, we study how to find all optimal itemsets independent of the correlation measure. We do this by mining the itemsets on the *convex hull* in ROC space. We relate this approach to an algorithm by Bayardo et al. for mining

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

KDD'09, June 28–July 1, 2009, Paris, France.

Copyright 2009 ACM 978-1-60558-495-9/09/06 ...\$5.00.

interesting itemsets [3]. We show that the hull can be mined effectively and that this set of patterns is surprisingly small.

This paper is organized as follows. First we review earlier work on branch and bound search for correlated patterns (Section 2), and review our approach for formalizing itemset mining in constraint programming (Section 3). Subsequently, we show how to formalize correlated pattern mining in constraint programming, and prove the correctness of our formalization in Section 4. We perform experiments in Section 5 which explain in detail the reasons for the significant improved performance of our method compared to other algorithms for the basic setting. In Section 6 we study how to mine the convex hull of itemsets. We conclude in Section 7.

2. CORRELATED PATTERN MINING

2.1 Problem Setting

While introducing the problem of correlated pattern mining, we assume that the reader is familiar with the standard formulation of frequent itemset mining, and focus on the differences between frequent and correlated itemset mining.

We assume we are given a set of possible items $\mathcal{I} = \{1, 2, \dots, m\}$ and a dataset \mathcal{D} consisting of n examples $\{(X_1, y_1), \dots, (X_n, y_n)\}$, where $y_t \in \{+, -\}$ is a class label and $X_t \subseteq \mathcal{I}$. It will sometimes be convenient to represent the database as a binary matrix, where $\mathcal{D}_{ti} = 1$ if and only if $i \in X_t$ and $\mathcal{D}_{ti} = 0$ otherwise. Furthermore, \mathcal{T} denotes the transaction identifiers $\mathcal{T} = \{1, 2, \dots, n\}$. Given a set of transactions T , T^c denotes the set of transaction identifiers in T having class c , $\{t \mid t \in T, y_t = c\}$. We use $\varphi(I) = \{t \mid t \in T, I \subseteq X_t\}$ to denote the set of transactions in database \mathcal{D} in which itemset I occurs and $\varphi^c(I)$ as shorthand notation for $(\varphi(I))^c$. The vector $\sigma(T) = (|\mathcal{T}^+|, |\mathcal{T}^-|)$ is called the *stamp point* of transaction set T ; similarly, we define the stamp points of an itemset I as $\sigma(I) = \sigma(\varphi(I)) = \sigma(|\varphi^+(I)|, |\varphi^-(I)|)$. So the stamp point of an itemset is a vector (p, n) where p is the frequency of this itemset in \mathcal{T}^+ and n is the frequency of this itemset in \mathcal{T}^- .

In correlated itemset mining, we are given a function which computes a *correlation score* for an itemset, that is, a function $f(\sigma(I))$, or short-handed $f(I)$. One such measure is *information gain*:

$$IG(p, n) = H(|\mathcal{T}^+|, |\mathcal{T}^-|) - \frac{p+n}{|\mathcal{T}|} \cdot H(p, n) - \frac{|\mathcal{T}| - (p+n)}{|\mathcal{T}|} \cdot H(|\mathcal{T}^+| - p, |\mathcal{T}^-| - n),$$

where

$$H(a, b) = -\frac{a}{a+b} \cdot \log_2\left(\frac{a}{a+b}\right) - \frac{b}{a+b} \cdot \log_2\left(\frac{b}{a+b}\right).$$

Other, similar measures are χ^2 and Fisher scores. These scoring functions have two important properties.

DEFINITION 1. A scoring function is zero diagonal convex (ZDC) if it has the following two properties:

- the function is convex, i.e., for every pair of stamp points $\sigma \neq \sigma'$ it holds that
$$\forall 0 \leq \alpha \leq 1 : f(\alpha\sigma + (1-\alpha)\sigma') \leq \alpha f(\sigma) + (1-\alpha)f(\sigma').$$
- the function reaches its minimum in all stamp points on the diagonal in PN-space, i.e.,
$$\forall 0 \leq \alpha \leq 1 : f(\alpha|\mathcal{T}^+|, \alpha|\mathcal{T}^-|) = 0.$$

THEOREM 1. Fisher score, information gain, gini index, and chi-square are ZDC correlation measures.

Definitions, as well as independent, alternative proofs of this theorem, can be found in [8, 20, 21]. A plot of χ^2 is given in Figure 1, and illustrates these two properties. A characteristic of these correlation measures is that they are symmetric: both $f(|\mathcal{T}^+|, 0)$ and $f(0, |\mathcal{T}^-|)$ are local maxima. Asymmetric scoring functions, that is, functions in which only one of $f(|\mathcal{T}^+|, 0)$ and $f(0, |\mathcal{T}^-|)$ is a local maximum, are also common; examples include Laplace value, confidence and conviction; furthermore, the growth rate used in emerging patterns is also asymmetric [12]. We focus our attention in this paper on the harder, symmetric case, even though our method can be extended to the asymmetric case.

The *top-k correlated itemset mining problem* can now be defined as:

- Given** a database \mathcal{D} over a set of items \mathcal{I} , an integer k , and a correlation measure f ;
Find the k highest ranked itemsets $I \subseteq \mathcal{I}$ according to their $f(I)$ values.

In this paper we focus mostly on this *optimization* version of the correlated itemset mining problem, for the case that $k = 1$. An alternative problem is to find *all* itemsets $I \subseteq \mathcal{I}$ for which $f(I) \geq \theta$, given a threshold θ ; our technique can easily be adapted towards these other cases.

Notice that none of these formulations involves a minimum frequency threshold. Some authors have defined discriminative patterns as those patterns that are frequent in one class and infrequent in another, given two support thresholds [19]; in this work we do not consider approaches that require the explicit specification of frequency thresholds.

It is often useful to restrict one's attention to *closed* itemsets [25, 29], which are itemsets whose supersets have a different frequency. Given a set of transactions T , the largest itemset in common between these transactions is $\psi(T) = \bigcap_{t \in T} X_t$. An itemset I is *closed* iff $I = \psi(\varphi(I))$. Closed itemsets form a condensed representation of the set of all frequent itemsets: if there is an itemset with correlation score θ , there also is a closed itemset with this score. Usually closed itemsets can be found faster. Taking closedness into account we obtain the problem of *closed top-k correlated itemset mining*.

2.2 Bounds

Most correlated pattern miners employ a branch and bound algorithm to avoid having to enumerate all possible itemsets. They realize this by using bounds on correlation. In our approach, the following result will be essential.

THEOREM 2. Let f be a ZDC correlation measure and $0 \leq p_1 \leq p_2$ and $0 \leq n_1 \leq n_2$. Then

$$\max_{(\sigma_1, \sigma_2) \in [p_1, p_2] \times [n_1, n_2]} f(\sigma_1, \sigma_2) = \max\{f(p_1, n_2), f(p_2, n_1)\}$$

The bound states that to find the highest possible score in a rectangle of stamp points, it suffices to check two corners of the rectangle. A proof can be found in the appendix. The bound generalizes earlier bounds, e.g., those of [21, 8] which cover cases where $p_1 = n_1 = 0$.

We will now first review the bounds that were used in earlier methods and then show how our bound improves on them.

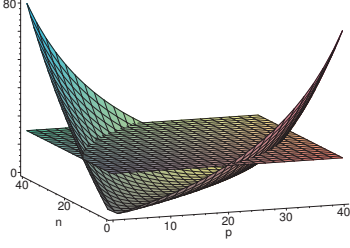


Figure 1: A plot of the χ^2 scoring function, and a threshold on χ^2 .

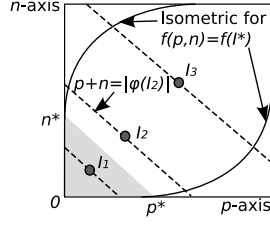


Figure 2: The 1-support bound in PN-space.

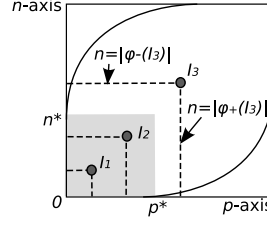


Figure 3: The 2-support bound in PN-space.

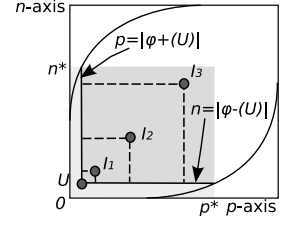


Figure 4: The 4-support bound in PN-space.

In discriminative itemset mining [8, 9, 27, 13], the search is performed depth-first, using similar data structures as traditional frequent itemset miners (such as FP-Trees in the DDPMine algorithm [9]). The aim is to prune those itemsets I for which it can be shown that no $J \supseteq I$ can have $f(J) \geq f(I^*)$, where I^* is the best itemset found so far. The main limitation of discriminative itemset miners is that during the search, only the total support values of itemsets are determined. The supports on the classes separately are not taken into account. Hence, given an itemset I , we need to derive a bound on $f(J)$ for all $J \supseteq I$ from the value $|\varphi(I)|$ only. Given this information, the stamp points (p, n) of the itemsets J which can possibly be reached satisfy

$$0 \leq p \leq |\mathcal{T}^+|, \quad 0 \leq n \leq |\mathcal{T}^-|, \quad p + n \leq |\varphi(I)|,$$

where $p = |\varphi^+(J)|$ and $n = |\varphi^-(J)|$. This leads to the following result, which can be used for pruning itemsets.

THEOREM 3 (1-SUPPORT PRUNING). *Given an itemset I and a ZDC correlation measure f , if*

$$\max\{f(\min\{|\varphi(I)|, |\mathcal{T}^+|\}, 0), f(0, \min\{|\varphi(I)|, |\mathcal{T}^-|\})\} < f(I^*)$$

then $\forall J \supseteq I : f(J) < f(I^)$.*

In Figure 2 we visualize this in *PN space* [15]. Each point in PN space corresponds to a stamp point. An *isometric* in PN space is a line that connects stamp points with the same score according to a correlation function. For a given optimal itemset I^* with score $f(I^*)$, we have plotted the isometric $f(p, n) = f(I^*)$; stamp points that improve the score $f(I^*)$ are found in the upper-left and lower-right corner (see Figure 1). Furthermore, we have drawn the isometrics $p + n = |\varphi(I_k)|$ for three given itemsets I_1, I_2, I_3 . Itemset I_1 can be pruned using the 1-support bound but itemsets I_2 and I_3 cannot be pruned, because given only the total support information $|\varphi(I_k)|$ ($k = 1, 2$), we cannot conclude whether the score of an itemset $J \supseteq I_k$ can exceed $f(I^*)$ or not. The area of itemsets that can be pruned is indicated in gray.

The importance of this bound is that it relates minimum correlation to minimum support, and hence makes it possible to apply traditional itemset miners to correlated itemset mining. As can be seen in PN-space, given an itemset I^* , there is a stamp point $(p^*, 0)$ in which $f(p^*, 0) = f(I^*)$ and a stamp point $(0, n^*)$ in which $f(0, n^*) = f(I^*)$. We can reformulate the pruning condition as follows:

$$\max\{f(\min\{|\varphi(I)|, |\mathcal{T}^+|\}, 0), f(0, \min\{|\varphi(I)|, |\mathcal{T}^-|\})\} < f(I^*) \\ \iff |\varphi(I)| < \min\{p^*, n^*\}. \quad (1)$$

As a consequence a correlated itemset miner that uses 1-support pruning can never be more efficient than a frequent itemset miner run with support threshold $\min\{p^*, n^*\}$.

The restriction that only $|\varphi(I)|$ is available during the search is not present in the pruning strategy proposed in [21] and used in [21, 24, 22, 6, 30, 5, 17, 27]. This method assumes that both $|\varphi^+(I)|$ and $|\varphi^-(I)|$ are available during the search, and, hence, it can be determined that an itemset $J \supseteq I$ can only have stamp points (p, n) with

$$0 \leq p \leq |\varphi^+(I)|, \quad 0 \leq n \leq |\varphi^-(I)|.$$

This yields the following pruning strategy.

THEOREM 4 (2-SUPPORT PRUNING). *Given an itemset I and a ZDC correlation measure f , if*

$$\max\{f(|\varphi^+(I)|, 0), f(0, |\varphi^-(I)|)\} < f(I^*)$$

then $\forall J \supseteq I : f(J) < f(I^)$.*

This bound is visualized in Figure 3. In our example we can now also prune itemset I_2 , as the additional support information allows us to determine that the best stamp point we can reach from I_2 is either $(0, |\varphi^-(I_2)|)$ or $(|\varphi^+(I_2)|, 0)$, and these scores are lower than $f(I^*)$. Also in general the 2-support bound is more powerful than the 1-support bound.

Both 1-pruning and 2-pruning assume that the best reachable itemset $J \supseteq I$ covers zero examples in one of the two classes. We will now drop this assumption. Assume that we know for all $J \supseteq I$ enumerable in the depth first search below I , there is a set $U \subseteq \varphi(J)$ of *unavoidable* transactions (thus the unavoidable transactions are those that are covered by all refinements of I). Then the additional support information $|U^+|$ and $|U^-|$ can be used to obtain an improved 4-support pruning condition:

$$\max\{f(|\varphi^+(I)|, |U^-|), f(|U^+|, |\varphi^-(I)|)\} < f(I^*).$$

This is illustrated in Figure 4, where we are computing a bound for itemsets J with $U \subseteq \varphi(J)$. We have indicated the stamp point $\sigma(U)$. Observe that I_3 is now also pruned, as any itemset $J \supseteq I_3$ is assumed to cover at least $|U^+|$ positive and $|U^-|$ negative examples, and both $f(|\varphi^+(I)|, |U^-|) < f(I^*)$ and $f(|U^+|, |\varphi^-(I)|) < f(I^*)$. The area of stamp points pruned, as indicated by the gray box in Figure 4, is largest for the 4-support bound. A practical complication is however the need to compute the set U . The simplest solution would be to choose $U = \emptyset$, which reduces the 4-pruning bound to the 2-support bound. Ideally, U should be as large as possible, without affecting the optimality of the search. To solve this problem, we will employ the principles of constraint programming that we introduce in the next section.

Algorithm 1 Constraint-Search($Dom(V), C$)

```
1:  $Dom(V) := \text{propagate-till-fix-point}(Dom(V))$ 
2: if  $\exists v \in V : dom(v) = \emptyset$  then
3:   return
4: end if
5: if  $\exists x \in \mathcal{V} : |dom(x)| > 1$  then
6:   select such an  $x$  using some heuristic
7:   for all  $d \in dom(x)$  do
8:     Constraint-Search( $Dom(V), C \cup \{x = d\}$ )
9:   end for
10: else
11:   Output solution
12: end if
```

3. CONSTRAINT PROGRAMMING

Constraint programming (CP) is a flexible programming paradigm in which problems are specified declaratively using constraints on variables. More formally, a problem is specified by a set of variables V , for each variable v an associated domain $dom(v)$, and a set of constraints C on possible assignments of values to these variables. In this paper, we shall only employ binary domains, that is, for all variables $dom(v) = \{0, 1\}$. The problem then is to find an assignment of values to all the variables in V that satisfies all constraints in C . Solutions are found using general purpose solvers that combine two main ideas. First, solvers employ *constraint propagation* pervasively. This means that the domains of variables are iteratively reduced by exploiting the constraints amongst the variables. By shrinking the domains of the variables, one effectively reduces the number of possible variable assignments to consider. To compute the reduced domains, constraint programming techniques employ *propagators*. Secondly, when it is no longer possible to reduce the domains (and hence a fix-point of the propagation is reached), the solver branches, which means that it selects a variable v (according to some heuristic) and recursively calls the solver while assigning a value to this variable. This process is continued until a domain becomes empty and the solver backtracks, or a solution is found. The resulting algorithm is summarized in Algorithm 1, where $Dom(V)$ represents the set of all domains of the variables V , and C the set of constraints.

In [10] we studied how to formalize constrained itemset mining in CP systems. We will illustrate this for the problem of *frequent itemset mining*. In its CP formulation we search in parallel for an itemset I and a transaction set T . We use as variables $V = \{I_1, \dots, I_m\} \cup \{T_1, \dots, T_n\}$; I_i represents item i ; in a final solution, $I_i = 1$ corresponds to $i \in I$; T_t represents transaction t ; $T_t = 1$ corresponds to $t \in T$. Moreover, two constraints need to be satisfied:

$$T = \varphi(I) \quad (2)$$

$$|T| \geq \theta \quad (3)$$

The combination of these constraints expresses the traditional support constraint that $|\varphi(I)| \geq \theta$ must hold. The problem of frequent itemset mining can be reformulated in terms of constraint programming as follows; cf. [10]. Given are a database \mathcal{D} and a threshold θ . The goal is to find

$I_i, T_t \in \{0, 1\}$ such that

$$\forall t \in \mathcal{T} : T_t = 1 \leftrightarrow \sum_{i \in \mathcal{I}} I_i (1 - \mathcal{D}_{ti}) = 0 \quad (4)$$

$$\forall i \in \mathcal{I} : I_i = 1 \rightarrow \sum_{t \in \mathcal{T}} T_t \mathcal{D}_{ti} \geq \theta \quad (5)$$

The first constraint is called the *coverage* constraint, the second the *support* constraint. The coverage constraint states that a transaction $T_t = 1$ if and only if t is covered by I , that is, it belongs to $\varphi(I)$. The support constraint basically states that if an item $I_i = 1$ then the support of i in T should be larger than θ . To see why these CP constraints correspond to itemset mining constraints (2) and (3), consider that $\sum_{i \in \mathcal{I}} I_i (1 - \mathcal{D}_{ti}) = 0$ iff $\forall i \in \mathcal{I} : I_i = 0 \vee \mathcal{D}_{ti} = 1$ iff $I \subseteq X_t$ iff $t \in \varphi(I)$. Also, knowing that $T = \varphi(I)$, we can deduce for $I \neq \emptyset$ that $|T| \geq \theta$ iff $\forall i \in I : |\varphi(\{i\}) \cap T| \geq \theta$ iff $\forall i \in \mathcal{I} : I_i = 1 \rightarrow \sum_{t \in \mathcal{T}} T_t \mathcal{D}_{ti} \geq \theta$.

A constraint of the form

$$I_i = 1 \rightarrow \sum_{t \in \mathcal{T}} T_t \mathcal{D}_{ti} \geq \theta, \quad (6)$$

containing a sum within an implication, is called a *reified summation constraint*. Reified constraints are readily available in many CP systems and in our case allow for pervasive propagation. The propagation for the constraint of equation (6) is performed as follows. Let us define the minimum value of T_t according to its domain as $T_t^{min} = \min_{d \in dom(T_t)} d$ and the maximal value as $T_t^{max} = \max_{d \in D(T_t)} d$. Then the propagator for constraint (6) is the following:

$$\text{if } \sum_{t \in \mathcal{T}} T_t^{max} \mathcal{D}_{ti} < \theta \text{ then } I_i^{max} = 0.$$

This propagator states that when the highest possible value the sum can take is too low, $dom(I_i)$ is modified to not contain the value 1. In a similar way, propagators for other reified summation constraints can be defined; the CP system will execute a propagator whenever one of its variables is changed (even though the order of execution is often system dependent). Using these constraints and propagators in Algorithm 1 results in a search that is similar to many specialized depth-first itemset miners. Indeed, once we set $dom(I_i) = \{1\}$ in line 8 of Algorithm 1, we include item i in the itemset, and recurse for the resulting itemset; in line 1 the propagation ensures that first transactions that do not include item i are set to $T_t^{max} = 0$; then, all items which are no longer frequent in T are set to $I_i^{max} = 0$, and the search recurses over those items that have not been fixed yet.

The key advantage of constraint programming is that it is very general and flexible. In an itemset mining context this manifests itself in that it is easy to incorporate further constraints and also to combine them in a complex way. All that is needed to realize this is a formulation of the constraint and a propagator. The CP system takes care that solutions are found. In [10] it has been shown that the above CP formulation of frequent itemset mining can be extended to cope with anti-monotonic, monotonic, succinct, closed, free, fault-tolerant patterns, etc. One example of such a constraint that will be used later on in this paper is that for obtaining closed sets:

$$\forall i \in \mathcal{I} : I_i = 1 \leftrightarrow \sum_{t \in \mathcal{T}} T_t (1 - \mathcal{D}_{ti}) = 0; \quad (7)$$

This *closedness* constraint is very similar to the coverage constraint and the propagation of this constraint can be shown to emulate the search strategy of LCM [25].

Also for other constraints, other itemset miners are emulated; for instance, for monotonic constraints, CP emulates the pruning strategies of ExAnte and DualMiner [4, 7].

4. CORRELATED ITEMSET MINING IN CP

While previously we have shown how to tackle a wide variety of constraint based itemset mining problems with constraint programming [10], the questions were left open as how to tackle top- k queries or to find correlated patterns. This section shows how the optimization version of correlated itemset mining can be realized using constraint programming. It can be specified as follows. Given is a database \mathcal{D} and a ZDC correlation measure f ; the problem is to find $I_i, T_t \in \{0, 1\}$ and the *maximal value* $\theta \in \mathcal{R}$ such that the following constraints are satisfied.

$$\forall t \in \mathcal{T} : T_t = 1 \iff \sum_{i \in \mathcal{I}} I_i (1 - \mathcal{D}_{ti}) = 0 \quad (8)$$

$$\forall i \in \mathcal{I} : I_i = 1 \rightarrow f\left(\sum_{t \in \mathcal{T}^+} T_t \mathcal{D}_{ti}, \sum_{t \in \mathcal{T}^-} T_t \mathcal{D}_{ti}\right) \geq \theta \quad (9)$$

This formulation is called the *CP version* of the top-1 correlated itemset mining problem. The first constraint is again the coverage constraint, the second one is the *correlation* constraint. It is shown in the appendix that this reified formulation is correct, cf. Theorem 5. As argued before, it is easy to add further constraints to the CP engine in a flexible manner. One constraint that is useful in the context of correlated pattern mining is the closedness constraint, which shall be used in our experiments. Furthermore, using the 4-support bound, the following propagator can be shown to be *sound* and *complete* for the correlation constraint (9), cf. Theorem 6 and its proof in the appendix.

$$\begin{aligned} \text{if } \max \{ & f\left(\sum_{t \in \mathcal{T}^+} T_t^{max} \mathcal{D}_{ti}, \sum_{t \in \mathcal{T}^-} T_t^{min} \mathcal{D}_{ti}\right), \\ & f\left(\sum_{t \in \mathcal{T}^+} T_t^{min} \mathcal{D}_{ti}, \sum_{t \in \mathcal{T}^-} T_t^{max} \mathcal{D}_{ti}\right) \} < \theta \\ \text{then } & I_i^{max} = 0. \end{aligned} \quad (10)$$

To deal with the optimization criterion in CP, the threshold θ is increased during the search whenever a solution is found. Most CP systems provide functionality for this.

An interesting question is now how the propagation of this propagator differs from the search strategies used in traditional correlated itemset miners. It is important to see that the propagator also considers the T_t^{min} variables; if $T_t^{min} = 1$, transaction t is *unavoidable* and included in all transaction sets deeper down the search. To see how T_t^{min} is set we need to consider the propagators for the coverage constraint (8). One propagator is the following:

$$\text{if } \sum_{i \in \mathcal{I}} (1 - \mathcal{D}_{ti}) I_i^{max} = 0 \quad \text{then } \quad T_t^{min} = 1. \quad (11)$$

Following [7], let us denote by $I_{tail} = \{i \mid I_i^{max} = 1\}$ the largest itemset that can still be reached at a specific point in the search space. This propagator states that if for all

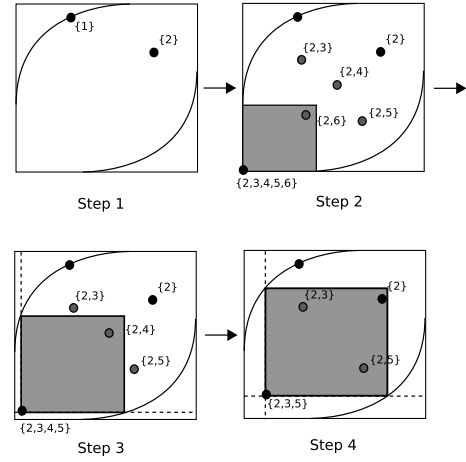


Figure 5: Example of how CP prunes the search.

items $i \in I_{tail}$ we have that $\mathcal{D}_{ti} = 1$, i.e. $I_{tail} \subseteq X_t$, then we know that $T_t^{min} = 1$, that is, transaction t is unavoidable.

CP systems propagate constraints whenever changes in a domain occur. This leads to two types of propagation that are uncommon in existing correlated itemset miners. First, if the search branches over an item i and excludes it by setting $I_i^{max} = 0$, the data is scanned again, as the removal of this item might make certain transactions unavoidable, which may improve the bounds for remaining items (using the 4-support bound). This contrasts with most itemset miners that only scan the data when an item is added to an itemset. Second, propagation is repeated as long as changes in domains occur, as illustrated in the following example, which is visualized in Figure 5.

In this example we assume that the best itemset found so far is the itemset $\{1\}$. This itemset reaches a certain score θ ; the curved lines indicate the isometrics for this score. The itemset which we are refining is the itemset $\{2\}$, the possible items which we can add to this itemset are 3, 4, 5 and 6. Hence, we evaluate the support of itemsets $\{2, 3\}$, $\{2, 4\}$, $\{2, 5\}$, $\{2, 6\}$ and $\{2, 3, 4, 5, 6\}$ (step 2). Itemset $\{2, 3, 4, 5, 6\}$ is I_{tail} in the initial situation and is the largest itemset that can still be reached; it determines the set of unavoidable transactions by propagating the coverage constraint using propagator (11). Assume that we find that the support of $\{2, 3, 4, 5, 6\}$ is zero in both classes. The gray box indicates the stamp points of itemsets that will be pruned. In our example, $\{2, 6\}$ falls within this box. Consequently we know that no itemset containing $\{2, 6\}$ will achieve sufficient correlation. Hence we remove 6 from consideration. This means that the largest itemset we can reach now is itemset $\{2, 3, 4, 5\}$, whose supports in the classes are evaluated (step 3) to determine the number of unavoidable transactions. Even though these supports may be small, the isometrics of many correlation measures are such that the set of prunable itemsets becomes significantly larger. In our example we now find that $\{2, 4\}$ cannot achieve significant correlation. Consequently, we remove item 4 from consideration and we evaluate the support of $\{2, 3, 5\}$ (step 4). As a result, the prunable region is increased, and we find that no pattern can reach the desired correlation score; we prune the search for all itemsets below $\{2\}$.

5. EXPERIMENTS

In this section, we report on experiments for top-1 correlated itemset mining. We compare several variations of our approach and other state-of-the-art algorithms to get insight in the differences in performance between the approaches.

Experimental setup.

We use data from the UCI repository¹. To deal with missing values we preprocessed each dataset by first eliminating all attributes having more than 10% of missing values and then removing all instances (transactions) for which the remaining attributes still had missing values. Numerical attributes were discretized in a number of binary splits, setting thresholds for these splits using equal-frequency binning. Nominal attributes are represented using one item for every value. For multi-class problems we combined all smallest classes together in one class. Experiments were run on PCs with Intel Core 2 Duo E6600 processors and 4GB of RAM, running Ubuntu Linux. The code of our implementation and the datasets used are available on our website².

To make an empirically fair comparison of the three different bounds, we implemented them all in a Constraint Programming system. The system we use is Gecode³, which is an open and efficient CP solver. Because of its generality, it was sufficient to plug in a propagator for each of the bounds, and model the problem as defined in Section 4. The resulting system is called *cimcp*. The correlation measure used is *information gain*, although any ZDC correlation measure can be supplied. As variable-selection heuristic the *most-constrained* heuristic was used; this is comparable with an item ordering by increasing frequency. In our earlier work [10] we noticed that the Gecode system incurs extra overhead on very sparse data. In a specialized itemset mining implementation, based on the same principles, this overhead would not occur. To show this, and to show that any principle discovered in CP can be straightforwardly added to existing itemset miners, we implemented the specialized correlated itemset miner *corrmine*. It implements the same propagation as the CP system, including iterative *k*-support pruning and closedness pruning; it is based on ECLAT [28], a memory-efficient depth-first miner. Also this algorithm is provided on our website. In all experiments we only mine for closed itemsets. We found that this is always more efficient.

We will compare our approach with the state-of-the-art *ddpmine* algorithm, presented in [9]. *ddpmine* is based on an implementation of FPGrowth by Zhu and Grahne [29], which stores closed itemsets in memory during the search to prevent duplicate itemsets from being found; it uses *1*-support pruning. In our experiments the algorithm often ran out of memory; this will be indicated by a “-” in our result tables. In our experiments we answer the following questions.

Q1: How many discretization bins should be used?

In Figure 6 we compare the runtime of *ddpmine*, *cimcp* and *corrmine* on the same dataset, but discretized with a different number of bins. Using more bins, one can expect the correlation score of the correlated itemsets to increase;

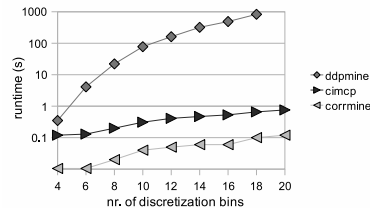


Figure 6: Runtime for the anneal dataset, having 812 transactions. The number of items ranges from 53 to 147.

Name	Dense	Trans	Item	4-sup	2-sup	1-sup
anneal	0.45	812	93	0.22	24.09	72.71
australian-cr	0.41	653	125	0.30	0.63	17.52
breast-wisc	0.50	683	120	0.28	13.66	228.08
diabetes	0.50	768	112	2.45	128.04	>
german-cr	0.34	1000	112	2.39	66.79	>
heart-clevel	0.47	296	95	0.19	2.15	29.58
hypothyroid	0.49	3247	88	0.71	10.91	>
ionosphere	0.50	351	445	1.44	>	>
kr-vs-kp	0.49	3196	73	0.92	46.20	713.35
letter	0.50	20000	224	52.66	>	>
mushroom	0.18	8124	119	14.11	13.48	27.31
pendigits	0.50	7494	216	3.68	>	>
primary-tum	0.48	336	31	0.03	0.13	0.85
segment	0.50	2310	235	1.45	>	>
soybean	0.32	630	50	0.05	0.07	0.38
splice-1	0.21	3190	287	30.41	31.11	35.02
vehicle	0.50	846	252	0.85	>	>
yeast	0.49	1484	89	5.67	781.63	>

Table 1: Statistics of UCI datasets, and runtimes, in seconds, of the CP model for the different bounds.

however, the size of the transaction database will also increase, as will the runtimes. Using 20 bins, the *ddpmine* algorithm is no longer able to find the optimal solution because of memory problems. To keep runtimes at reasonable levels for all algorithms we use 8 bins in further experiments.

Q2: How efficient are the 1,2 and 4-support bounds?

Table 1 shows the difference in runtime when using the different bounds. The density of a dataset, listed in column 2, is the fraction of 1’s in the binary representation of the transaction database. “>” indicates that no solution was found within 900 seconds. Using the *1*-support bound, we can only find the optima for 10 out of the 18 datasets. Using the *2*-support bound, the same optima are found significantly faster, and for even more datasets. The *4*-support bound finds these even faster, and for all datasets. For the mushroom dataset, the *4*-support bound takes slightly more time than the *2*-support bound, but the next experiment will show that this is due to the overhead incurred in CP systems on data of low density. In general it is clear that using the *4*-support bound results in the most pruning power, reducing the search-space enough to make exhaustive search feasible, and even fast.

Q3: Are 4-support bound miners faster than the state-of-the-art?

Table 2 shows the result of comparing the *corrmine*, *cimcp* and *ddpmine* implementations on the different datasets. We also compare the runtimes with LCM, one of the fastest, most memory efficient closed itemset mining implementation from the FIMI challenge [25]. We ran LCM with the

¹<http://archive.ics.uci.edu/ml/>

²<http://www.cs.kuleuven.be/~dtai/CP4IM/>

³<http://gecode.org>

Name	corrmine	cimcp	ddpmine	lcm
anneal	0.02	0.22	22.46	7.92
australian-credit	0.01	0.30	3.40	1.22
breast-wisconsin	0.03	0.28	96.75	27.49
diabetes	0.36	2.45	–	697.12
german-credit	0.07	2.39	–	30.84
heart-cleveland	0.03	0.19	9.49	2.87
hypothyroid	0.02	0.71	–	>
ionosphere	0.24	1.44	–	>
kr-vs-kp	0.02	0.92	125.60	25.62
letter	0.65	52.66	–	>
mushroom	0.03	14.11	0.09	0.03
pendigits	0.18	3.68	–	>
primary-tumor	0.01	0.03	0.26	0.08
segment	0.06	1.45	–	>
soybean	0.01	0.05	0.05	0.02
splice-1	0.05	30.41	1.86	0.02
vehicle	0.07	0.85	–	>
yeast	0.80	5.67	–	185.28
avg. when found:	0.15	6.55	28.88+	81.54+

Table 2: Runtimes, in seconds, of 3 correlated and one frequent itemset miner.

minimum support threshold of equation (1) (see Section 2), obtained by first running our system to find the optimal itemset. It therefore represents the minimal amount of time a 2-phase algorithm would need, which would first enumerate all frequent closed itemsets, and in a second step would select the most correlated one.

corrmine, our specialized implementation using 4-support pruning, is by far the most efficient on all datasets. The difference in runtime between *corrmine* and *cimcp* illustrates the overhead of using the general CP system. *ddpmine* does not find a solution for many of the datasets (indicated by “–”). This is a limitation of the pruning of the 1-support bound, as our implementation of the 1-support bound in Table 1 is not able to find the solution in a reasonable amount of time either. For the mushroom and splice datasets, *ddpmine* is faster than *cimcp*, but the results of our specialized implementation show convincingly that this is due to the overhead incurred in the Gecode system on sparse data. The runtimes of LCM confirm that a 2-phase algorithm can never be faster than a specialized 1-step algorithm like *corrmine*. In general, using the 4-support bound results in more effective algorithms, with some extra efficiency to be gained by creating a specialized implementation.

6. MINING THE CONVEX HULL

In the problem setting of correlated itemset mining we assumed we were given a ZDC correlation measure f . The correlation measure can be seen as a parameter of the method. In exploratory data mining parameter-free methods are often preferred. We can obtain a parameter-free setting by informally formulating the following problem setting: “can we find all itemsets for which there exists a ZDC correlation measure under which the itemset is optimal?” In ROC analysis it is well-known that the problem of finding such itemsets (rules, classifiers) can be solved by determining which ones are on the *convex hull* in ROC space [14]. Exploiting the aforementioned relation between correlated itemset mining and ROC analysis, we can now apply this result to correlated itemsets.

Due to lack of space, we here skip a formal definition of a

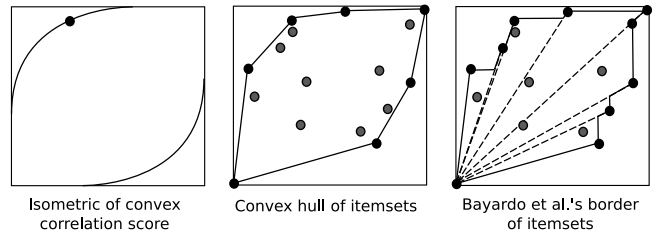


Figure 7: Comparison of an isometric, a convex hull and the border of Bayardo et al.: an isometric is a continuous curve; the convex-hull is piece-wise linear; Bayardo’s border also contains points not on the convex-hull.

convex hull; intuitively, given a set of points in PN -space, the convex hull consists of all stamp points on a rubber band that surrounds all stamp points as tightly as possible (including the $(0, 0)$ and $(|T^+|, |T^-|)$ points). For symmetric measures, the hull has two sides, one for each of the two classes. An example is given in Figure 7. We are interested in finding *all* (closed) itemsets on the convex-hull.

To find itemsets on the convex hull we need to implement a new propagator. The main idea behind this propagator is that the convex hull is a curve in PN -space, similar to an isometric for a correlation score; this curve is piece-wise linear between the stamp points on this curve (see Figure 7). Hence, instead of modifying the isometric during the search, by updating the correlation threshold, we will update the piece-wise linear curve when a new solution is found. When adding a new stamp point to the hull, previous hull points that are no longer on the convex hull are removed.

To find all itemsets on the hull, we replace propagator (10) with the following propagator:

$$\begin{aligned}
 \text{if } \text{in_hull} \left(\sum_{t \in T^+} T_t^{\max} \mathcal{D}_{ti}, \sum_{t \in T^-} T_t^{\min} \mathcal{D}_{ti}, \right. \\
 \left. \sum_{t \in T^+} T_t^{\min} \mathcal{D}_{ti}, \sum_{t \in T^-} T_t^{\max} \mathcal{D}_{ti} \right) \\
 \text{then } I_i^{\max} = 0. \tag{12}
 \end{aligned}$$

Here, $\text{in_hull}(p_2, n_1, p_1, n_2)$ checks whether we can still reach a stamp point outside the current convex hull. This test is performed as follows. We split the convex-hull in two sets of stamp points: one set of stamp points \mathcal{S}^- above the diagonal, i.e. stamp points (p, n) in which $n/|T^-| > p/|T^+|$, and one set of remaining stamp points below the diagonal, \mathcal{S}^+ . Then we check the following:

- let $(p_{\downarrow}^+, n_{\downarrow}^+) \in \mathcal{S}^+$ be the largest stamp point for which $p_{\downarrow}^+ < p_2$ and $(p_{\uparrow}^+, n_{\uparrow}^+) \in \mathcal{S}^+$ be the smallest stamp point for which $p_{\uparrow}^+ > p_2$; we propagate if $(p_2 - p_{\downarrow}^+) / (p_{\uparrow}^+ - p_{\downarrow}^+) < (n_1 - n_{\downarrow}^+) / (n_{\uparrow}^+ - n_{\downarrow}^+)$;
- let $(p_{\downarrow}^-, n_{\downarrow}^-) \in \mathcal{S}^+$ be the largest stamp point for which $n_{\downarrow}^- < n_2$ and $(p_{\uparrow}^-, n_{\uparrow}^-) \in \mathcal{S}^-$ be the smallest stamp point for which $n_{\uparrow}^- > n_2$; we propagate if $(n_2 - p_{\downarrow}^-) / (n_{\uparrow}^- - n_{\downarrow}^-) < (p_1 - p_{\uparrow}^-) / (p_{\uparrow}^- - p_{\downarrow}^-)$.

In Table 3 we compare the runtime of finding the top-1 most correlated itemset, according to the information gain measure, with the runtime needed to find all itemsets on the

Name	cimcp		size of hull
	time (s)	convex hull time (s)	
anneal	0.22	0.44	17
australian-credit	0.30	1.33	22
breast-wisconsin	0.28	0.83	20
diabetes	2.45	11.9	30
german-credit	2.39	3.93	21
heart-cleveland	0.19	0.37	20
hypothyroid	0.71	3.01	19
ionosphere	1.44	8.69	15
kr-vs-kp	0.92	1.75	17
letter	52.66	405.14	34
mushroom	14.11	32.45	10
pendigits	3.68	45.79	19
primary-tumor	0.03	0.07	16
segment	1.45	8.96	6
soybean	0.05	0.09	9
splice-1	30.41	40.13	10
vehicle	0.85	4.12	22
yeast	5.67	25.51	28
<i>average:</i>	<i>6.55</i>	<i>33.03</i>	<i>18.61</i>

Table 3: Mining the top-1 correlated itemset versus all itemsets on the convex hull in ROC space.

convex hull. Both are implemented in Gecode. Although more time is always required to mine the entire convex hull, we believe most runtimes are within acceptable boundaries. Except for datasets that incur extra overhead in Gecode, mining all itemsets on the hull is even faster than existing approaches to finding the top-1 itemset. The results show that mining all itemsets on the convex hull is a realistic alternative to having to choose a correlation measure up front. Furthermore, the results in Table 3 also show that the number of patterns is reasonably small, on average 19 itemsets on the entire convex hull.

This algorithm for mining itemsets on the convex-hull is related to the algorithm of Bayardo et al. for finding interesting correlated itemsets [2]. Bayardo et al.’s method for finding interesting correlated itemsets is an *indirect* method, in which a border set of itemsets is computed even if the correlation function is fixed; the optimal pattern is always chosen from this border set. Bayardo et al.’s border is similar to the convex hull in ROC space. The main difference is that they define a border of non-dominated itemsets in support-confidence space, which is a transformation of PN-space. Restricting ourselves to one half of PN-space, one itemset I dominates another itemset I' iff it dominates both in support and confidence, i.e. $|\varphi^+(I)| > |\varphi^+(I')|$ and $|\varphi^+(I)|/|\varphi(I)| > |\varphi^+(I')|/|\varphi(I')|$. This border representation is illustrated in PN-space in Figure 7. In this figure we have highlighted several isometrics for confidence. The border is along iso-support and iso-confidence lines; we see that the border of Bayardo et al. includes redundant stamp points compared to the convex-hull representation.

7. CONCLUSIONS

We have revisited the problem of correlated pattern mining starting from an analysis of correlation measures in PN-space and principles of constraint programming. This has allowed us to clarify the relationship between different bounds used for pruning and has also led to the introduction of a new bound that allows for significantly more pruning. The

approach has been empirically evaluated and shown to outperform state-of-the-art methods, either in terms of speed or memory requirements. Finally, we have adapted our technique for finding the set of all itemsets on the convex hull, a problem of relevance to the machine learning and classification literature.

Several challenges still remain. First, it is unclear in how far our results can be extended towards structured domains, as the approach depends on an upper-bound which may not be present in some structured pattern domains. Second, even though our algorithm can be plugged into many existing classification algorithms, an interesting question is whether it can be integrated in some algorithms exploiting ROC spaces [23].

Acknowledgements.

We are grateful to Albrecht Zimmermann for discussions, and to Hong Cheng for providing the implementation of DDPMine. This work was supported by a Postdoc and a project grant from the Research Foundation—Flanders, project “Principles of Patternset Mining”.

8. REFERENCES

- [1] S. D. Bay and M. J. Pazzani. Detecting change in categorical data: Mining contrast sets. In *KDD*, pages 302–306, 1999.
- [2] R. J. Bayardo Jr. and R. Agrawal. Mining the most interesting rules. In *KDD*, pages 145–154, 1999.
- [3] R. J. Bayardo Jr., R. Agrawal, and D. Gunopulos. Constraint-based rule mining in large, dense databases. In *ICDE*, pages 188–197, 1999.
- [4] F. Bonchi and C. Lucchese. Extending the state-of-the-art of constraint-based pattern discovery. *Data Knowl. Eng.*, 60(2):377–399, 2007.
- [5] B. Bringmann and A. Zimmermann. Tree² - decision trees for tree structured data. In *PKDD*, pages 46–58, 2005.
- [6] B. Bringmann, A. Zimmermann, L. De Raedt, and S. Nijssen. Don’t be afraid of simpler patterns. In *PKDD*, pages 55–66, 2006.
- [7] C. Bucila, J. Gehrke, D. Kifer, and W. M. White. DualMiner: A dual-pruning algorithm for itemsets with constraints. *Data Min. Knowl. Discov.*, 7(3):241–272, 2003.
- [8] H. Cheng, X. Yan, J. Han, and C.-W. Hsu. Discriminative frequent pattern analysis for effective classification. In *ICDE*, pages 716–725, 2007.
- [9] H. Cheng, X. Yan, J. Han, and P. S. Yu. Direct discriminative pattern mining for effective classification. In *ICDE*, pages 169–178, 2008.
- [10] L. De Raedt, T. Guns, and S. Nijssen. Constraint programming for itemset mining. In *KDD*, pages 204–212, 2008.
- [11] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis. Frequent substructure-based approaches for classifying chemical compounds. *IEEE Trans. Knowl. Data Eng.*, 17(8):1036–1050, 2005.
- [12] G. Dong and J. Li. Efficient mining of emerging patterns: Discovering trends and differences. In *KDD*, pages 43–52, 1999.
- [13] W. Fan, K. Zhang, H. Cheng, J. Gao, X. Yan, J. Han, P. S. Yu, and O. Verscheure. Direct mining of

discriminative and essential frequent patterns via model-based search tree. In *KDD*, pages 230–238, 2008.

- [14] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.
- [15] J. Fürnkranz and P. A. Flach. ROC ‘n’ rule learning – towards a better understanding of covering algorithms. *Machine Learning*, 58(1):39–77, 2005.
- [16] H. Grosskreutz, S. Rüping, and S. Wrobel. Tight optimistic estimates for fast subgroup discovery. In *ECML/PKDD (1)*, pages 440–456, 2008.
- [17] M. Hirao, H. Hoshino, A. Shinohara, M. Takeda, and S. Arikawa. A practical algorithm to find the best subsequence patterns. *Theor. Comput. Sci.*, 292(2):465–479, 2003.
- [18] B. Kavsek, N. Lavrac, and V. Jovanoski. APRIORI-SD: Adapting association rule learning to subgroup discovery. In *IDA*, pages 230–241, 2003.
- [19] S. Kramer, L. De Raedt, and C. Helma. Molecular feature mining in HIV data. In *KDD*, pages 136–143, 2001.
- [20] Y. Morimoto, T. Fukuda, H. Matsuzawa, T. Tokuyama, and K. Yoda. Algorithms for mining association rules for binary segmentations of huge categorical databases. In *VLDB*, pages 380–391, 1998.
- [21] S. Morishita and J. Sese. Traversing itemset lattice with statistical metric pruning. In *PODS*, pages 226–236, 2000.
- [22] S. Nijssen and J. N. Kok. Multi-class correlated pattern mining. In *KDD*, pages 165–187, 2005.
- [23] R. C. Prati and P. A. Flach. ROCCER: An algorithm for rule learning based on ROC analysis. In *IJCAI*, pages 823–828, 2005.
- [24] J. Sese and S. Morishita. Answering the most correlated n association rules efficiently. In *PKDD*, pages 410–422, 2002.
- [25] T. Uno, M. Kiyomi, and H. Arimura. Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In *FIMI*, 2004.
- [26] S. Wrobel. An algorithm for multi-relational discovery of subgroups. In *PKDD*, pages 78–87, 1997.
- [27] X. Yan, H. Cheng, J. Han, and P. S. Yu. Mining significant graph patterns by leap search. In *SIGMOD Conference*, pages 433–444, 2008.
- [28] M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. New algorithms for fast discovery of association rules. In *KDD*, pages 283–286, 1997.
- [29] J. Zhu and G. Grahne. Reducing the main memory consumptions of FPmax* and FPclose. In *FIMI*, 2004.
- [30] A. Zimmermann and B. Bringmann. CTC - correlating tree patterns for classification. In *ICDM*, pages 833–836, 2005.

APPENDIX

THEOREM 2. *Let f be a ZDC correlation measure and $0 \leq p_1 \leq p_2$ and $0 \leq n_1 \leq n_2$. Then*

$$\max_{(\sigma_1, \sigma_2) \in [p_1, p_2] \times [n_1, n_2]} f(\sigma_1, \sigma_2) = \max\{f(p_1, n_2), f(p_2, n_1)\}.$$

PROOF. The proof is similar to that of [21]. First, we observe that the function is *convex*. Hence, we know that the maximum in a space $[p_1, p_2] \times [n_1, n_2]$ is reached in one

of the points (p_1, n_1) , (p_1, n_2) , (p_2, n_1) and (p_2, n_2) . Next, we need to show that we can ignore the corners (p_1, n_1) and (p_2, n_2) . For this we observe that the minimum is reached on the diagonal. We can distinguish several situations.

If $n_1/|T^-| < p_1/|T^+|$, the point (p_1, n_1) is ‘below’ the diagonal. We know for the stamp point $(\frac{|T^+|}{|T^-|}n_1, n_1)$ on the diagonal that $f(\frac{|T^+|}{|T^-|}n_1, n_1) = 0$. Due to the convexity we know then that $f(\frac{|T^+|}{|T^-|}n_1, n_1) = 0 \leq f(p_1, n_1) \leq f(p_2, n_1)$.

Similarly, we can show that if (p_1, n_1) is above the diagonal that $f(p_1, n_1) \leq f(p_1, n_2)$; that $f(p_2, n_2) \leq f(p_2, n_1)$ if (p_2, n_2) is below the diagonal; and that $f(p_2, n_2) \leq f(p_1, n_2)$ if (p_2, n_2) is above the diagonal. \square

THEOREM 5. *The CP version of top-1 correlated itemset mining is equivalent to the top-1 correlated itemset mining problem.*

PROOF. We show that the itemset $I = \{i \mid I_i = 1\}$ satisfying the CP version of the problem is a solution to the original top-1 correlated itemset mining problem. We already showed that the two versions of the coverage constraint (4) and (2) are equivalent. Hence, we only need to show that the correlation constraint (9) ensures that $f(|\varphi^+(I)|, |\varphi^-(I)|) \geq \theta$. To see this, observe that we can rewrite $\varphi(I)$ as follows:

$$\varphi(I) = \{t \in \mathcal{T} \mid \forall i \in I : \mathcal{D}_{ti} = 1\} = \bigcap_{i \in I} \varphi(\{i\})$$

Using this observation, it follows that:

$$\begin{aligned} & f(|\varphi^+(I)|, |\varphi^-(I)|) \geq \theta \\ \iff & f\left(\left|\bigcap_{j \in I} \varphi^+(\{j\})\right|, \left|\bigcap_{j \in I} \varphi^-(\{j\})\right|\right) \geq \theta \\ \iff & \forall i \in I : \\ & f\left(\left|\varphi^+(\{i\}) \cap \varphi^+(\{j\})\right|, \left|\varphi^-(\{i\}) \cap \varphi^-(\{j\})\right|\right) \geq \theta \\ \iff & \forall i \in I : f(|\varphi^+(\{i\}) \cap \varphi^+(I)|, |\varphi^-(\{i\}) \cap \varphi^-(I)|) \geq \theta \\ \iff & \forall i \in \mathcal{I} : I_i = 1 \rightarrow f\left(\sum_{t \in T^+} T_t \mathcal{D}_{ti}, \sum_{t \in T^-} T_t \mathcal{D}_{ti}\right) \geq \theta. \end{aligned}$$

Finally, because θ is required to be maximal in the CP formulation, an itemset reaching at least score θ is a top-1 itemset. \square

THEOREM 6 (PROPAGATOR FOR REIFIED CORRELATION). *The propagator in equation (10) is sound and complete for the correlation constraint (9).*

PROOF. (Soundness) Given the current values for T_t^{max} and T_t^{min} , We know that the stamp points σ that we can still reach are in $[\sum_{t \in T^+} T_t^{min} \mathcal{D}_{ti}, \sum_{t \in T^+} T_t^{max} \mathcal{D}_{ti}] \times [\sum_{t \in T^-} T_t^{min} \mathcal{D}_{ti}, \sum_{t \in T^-} T_t^{max} \mathcal{D}_{ti}]$. Hence, according to Theorem 2, the best score we can still reach is given by the two corners checked in the propagator. If the condition of the propagator is no longer satisfied, we know that the condition $I_i = 1$ can no longer be satisfied, and change the domain.

(Completeness) We need to show that if all variables are fixed, the constraint is not satisfied iff the propagator derives a contradiction. This follows from the fact that if all variables are fixed, the two stamp points checked in the propagator are the same stamp point, i.e., the stamp point occurring in the constraint. \square