Under consideration for publication in Knowledge and Information Systems

Efficient mining of all margin-closed itemsets with applications in temporal knowledge discovery and classification by compression

Fabian Moerchen[1], Michael Thies[2] and Alfred Ultsch[2]

[1]Siemens Corporate Research, 755 College Road East, Princeton, NJ, 08540, USA
[2]Databionic Research Group, Philipps-University Marburg, Hans-Meerwein-Str, 35032, Marburg, Germany

Keywords: Closed Itemsets, Constrained Itemsets, Condensed Representation, Temporal Data Mining, Compression

Abstract. Margin-closed itemsets have previously been proposed as a subset of the closed itemsets with a minimum margin constraint on the difference in support to supersets. The constraint reduces redundancy in the set of reported patterns favoring longer, more specific patterns. A variety of patterns ranging from rare specific itemsets to frequent general itemsets is reported to support exploratory data analysis and understandable classification models. We present DCI_MARGIN, a new efficient algorithm that mines the complete set of margin-closed itemsets. We modified the DCI_CLOSED algorithm that has low memory requirements and can be parallelized. The margin constraint is checked on-the-fly reusing information already computed by DCI_CLOSED. We thoroughly analyzed the behavior on many datasets and show how other data mining algorithms can benefit from the redundancy reduction.

1. Introduction

Itemset mining has long advanced over the initial concept of market basket analysis Agrawal et al. (1993) and is used to tackle many data mining problems such as frequent pattern mining, association rule generation Hipp et al. (2000), clustering Wang et al. (1999), Beil et al. (2002), Fung et al. (2003), Malik & Kender (2006), classification Liu et al. (1998), Li et al. (2001), Yin & Han (2003), Garriga et al. (2006), van Leeuwen et al. (2006), Wang & Karypis (2006), Nijssen & Fromont (2007), Cheng et al. (2007) and temporal data mining Agrawal & Srikant (1995), Mörchen & Ultsch (2007). The

Received Sep 26, 2009
Revised Jun 05, 2010
Accepted Jun 13, 2010

mining of itemsets is a core step in these methods that often dominates the overall complexity of the problem. The mining of frequent itemsets is a challenging task because the possible number of patterns can be extremely large even for moderately sized datasets complicating a manual analysis or further automated processing steps Xin et al. (2005).

Researchers have proposed many solutions to reduce the number of patterns depending on the context in which the patterns are used, for example, condensed representations Calders et al. (2006), constrained itemsets Pei, Han & Lakshmanan (2001) and combinations thereof Bonchi & Lucchese (2006), De Raedt et al. (2008). For association rule generation closed itemsets Pasquier et al. (1999), Boulicaut & Bykowski (2000) are commonly used to avoid redundant rules Zaki (2004) favoring longer patterns to generate specific rules. For frequency queries non-derivable itemsets Calders & Goethals (2007) provide a compact lossless representation favoring shorter patterns to keep the summary small. Margin-closed itemsets have been previously proposed by the authors for exploratory knowledge discovery tasks in the context of temporal data mining Mörchen (2006), Mörchen & Ultsch (2007) and independently as δ-tolerance itemsets for frequency estimation in Cheng, Ke & Ng (2006). Margin-closed patterns are a specialization of closed itemsets with a constraint to limit the redundancy among reported patterns. An itemset is closed if no superset with the same frequency exists. An itemset is margin-closed if no superset with almost the same frequency exists, where 'almost' is defined by a threshold α on the relative (or absolute) difference of the frequencies. The threshold ensures a frequency *margin* among the reported patterns.

Note, that margin-closed itemsets are *not* an error-tolerant approach and *not* an approximation to closedness. In contrast to pattern summarization, error-tolerant, and approximation only actually observed itemsets with their exact frequencies are reported. In contrast to non-derivable itemsets the goal is not to support frequency queries with a compact summary but to provide long patterns with low redundancy to support exploratory data analysis tasks. Frequent patterns (per class) can help a human analyst understand the structure of a dataset. Less patterns with less redundancy are easier to comprehend. The bias towards longer patterns provides more explanation for each pattern to the user. When used as features in a classification model removing redundancy translates to faster training times and more concise models. In Cheng, Ke & Ng (2006) the authors explore margin-closed itemsets as a condensed representation for frequency estimation. The frequency of non margin-closed itemsets is approximated by the average frequency of the items in the common superset, again motivating the need for favoring longer itemsets.

In this paper we study the problem of *efficiently mining of all frequent margin-closed itemsets from a database of itemset transactions*. The margin-closed itemsets are a subset of all closed itemsets and a superset of all maximal itemsets and the mining of closed and maximal itemsets has been well studied. The naive approach would be to mine all closed itemsets and check the margin constraint for each one. This can be done by comparing the support of a closed itemset to the support of all extensions with one additional item. Obviously this is computationally expensive in particular for low minimum support values that generate large numbers of closed itemsets. Incorporating the pruning from closed to margin-closed itemsets into the mining algorithms can be expected to be more efficient. We propose DCI_MARGIN, a new algorithm based on DCI_CLOSED Lucchese et al. (2006*a*) that efficiently mines all margin-closed itemsets. Several pruning techniques are introduced and the correctness and completeness of the algorithm is shown. Our solution checks the margin constraint on-the-fly reusing information already computed by DCI_CLOSED. Previous work has adapted the FP-Growth Han & Pei (2001) and CHARM Zaki & Hsiao (2002) algorithms for closed itemset mining. The former does not guarantee completeness due to greedy pruning heuristices

Cheng, Ke & Ng (2006) and the latter is less efficient due to required subsumption check Mörchen (2006). We show that DCI_MARGIN can significantly reduce the number of reported patterns if there is redundancy with comparable or faster run time. The discovered redundancy at various minimum margin and support thresholds provides interesting insights into datasets from different domains. Our main contributions are:

- The efficient DCI_MARGIN algorithm that mines the complete set of margin-closed itemsets. Previous work used a variation of CHARM that has much higher memory requirements Mörchen (2006) and an adaption of FP-Growth Cheng, Ke & Ng (2006) that used greedy pruning heuristics leading to incomplete results as demonstrated by our experiments.
- A thorough evaluation of both the pattern class and the algorithm with 60 datasets from various domains. Previous work has concentrated on the special case of temporal data mining Mörchen (2006) or used only few itemset datasets Cheng, Ke & Ng (2006).
- A discussion of different applications of margin-closed itemsets.

In addition we provide several examples of data mining applications using margin-closed patterns.

In the remainder of this paper we motivate and define margin-closed itemsets in Section 2 and describe an efficient algorithm for their discovery in Section 3. Section 4 demonstrates how we can reduce the number of reported itemsets significantly and efficiently. Applications are described in Section 5. The results and related work are discussed in Sections 6-7.

2. Margin-closed itemsets

2.1. Basic Notation

Given a finite set of items $\mathcal{I}$ and a finite set of transactions $I \subseteq \mathcal{I}$, represented by unique identifiers $\mathcal{T}$, a dataset can be described as the relation $\mathcal{D} \subseteq \mathcal{I} \times \mathcal{T}$. The function $g(I) = \{t \in T | \forall i \in I : (i,t) \in \mathcal{D}\}$ returns the transactions in which all items of itemset I are included. The function $f(T) = \{i \in \mathcal{I} | \forall t \in T : (i,t) \in \mathcal{D}\}$ returns all items that are present in all transactions of T. The composite function $c = f \circ g$ is a closure operator (e.g., Lucchese et al. (2006*a*)). Let the support of an itemset I be the fraction of transactions in which the itemset occurs: $supp(I) = \frac{|g(I)|}{|\mathcal{T}|}$.

Definition 2.1. An itemset I is called frequent w.r.t. a minimum support threshold $1 \geq \theta \geq 0$, if $supp(I) \geq \theta$. Let the set of all frequent itemsets be `FI`.

Definition 2.2. An itemset $I \in$ `FI` is maximal if and only if $\forall I \subset I' \Rightarrow supp(I') < \theta$, i.e., if there is no frequent superset. Let the set of all maximal frequent itemsets be `MFI`.

Definition 2.3. An itemset $I \in$ `FI` is (frequent-)closed, if and only if $\forall I' \subseteq \mathcal{I} : I \subset I' \Rightarrow supp(I') \neq supp(I)$, i.e., if there is no superset with the same support. Let the set of all closed frequent itemsets be `CFI`.

The closure operator c partitions the lattice of the power sets of $\mathcal{I}$ into equivalence classes regarding the support. The unique suprema regarding the subset relation of those classes are the closed itemsets.

Itemset	BD	$ABCD$	ACD	C	(total)
Transactions	1	9	61	29	100

Table 1. Example data: four itemsets with the number of transactions composed of exactly these items.

Definition 2.4. An itemset $G \subseteq \mathcal{I}$ with $G = C \cup \{i\}$ [1] and $C \in \texttt{CFI}$ and $i \in \mathcal{I} \setminus C$ is called a *generator*.

Note, that this definition of a generator does not require minimality. A generator represents a seed itemset obtained by extending a closed itemsets with a new item, thus entering a new equivalence class. A generator can be used to obtain the closed itemset representing the class by adding all items that do not decrease the support.

2.2. Margin-closedness

Definition 2.5. An itemset $I \in \texttt{FI}$ is margin-closed w.r.t. a threshold $\alpha \in [0, 1]$ if and only if $\forall I' \in \texttt{FI} : I \subset I' \Rightarrow \frac{supp(I')}{supp(I)} < 1 - \alpha$, i.e., if there is no superset with *almost* the same support. Let the set of all margin-closed frequent itemsets w.r.t. to a threshold α be $\texttt{CFI}^\alpha$.

Example 2.6. Consider the example database in Table 1. There are four itemsets composed of the four items A, B, C, and D. For each itemset the dataset contains the indicated number of transactions with exactly these itemsets. Figure 1(a) shows the lattice of all frequent itemsets for $\theta = 0.09$ with itemsets connected by the subset relation. The empty set at the bottom is present in all 100 transactions, the set of all items at the top has an absolute support of 9. The rectangles indicate closed itemsets. No items can be added to these sets without decreasing the support. Figure 1(b) shows only the closed itemsets and the subset relations annotated with $\frac{supp(I')}{supp(I)}$ for $I \subset I'$. If we set $\alpha = 0.1$, i.e., we require a relative support margin of at least 10%, the itemsets $\emptyset$, D and BD would not be considered margin-closed and removed from Figure 1(b). For example, the superset ACD of D has 98.6% of the support of D, so D is considered redundant. In contrast, if the minimum support was set to $\theta = 0.1$ the itemset BD would be margin-closed because the superset $ABCD$ would not be frequent anymore, making BD a maximal itemset.

Corollary 2.7. $\texttt{CFI}^\alpha \subseteq \texttt{CFI}$, i.e., margin-closed frequent itemsets are a subset of the closed frequent itemsets.

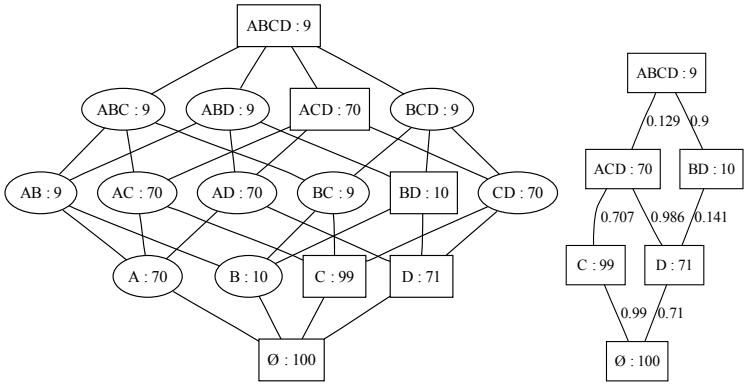
Proof: Let $I \in \texttt{CFI}^\alpha$, then $\forall I' \in \texttt{FI}$ with $I \subset I'$: $\frac{supp(I')}{supp(I)} < 1 - \alpha \Rightarrow supp(I') < (1 - \alpha)supp(I) \leq supp(I) \Rightarrow supp(I') < supp(I) \Rightarrow I \in \texttt{CFI}$

For $\alpha > 0$ the margin-closedness condition is stricter than the closedness condition. We ensure a margin of support between a reported itemset and any frequent superset. For $\alpha = 0$ the set of margin-closed itemsets is equal to the set of all closed itemsets.

Corollary 2.8. $\texttt{MFI} \subseteq \texttt{CFI}^\alpha$, i.e., maximal frequent itemsets are a subset of the margin-closed itemsets.

Proof: Let $I \in \texttt{MFI} \Rightarrow \nexists I' \in \texttt{FI}$ with $I \subset I' \Rightarrow \nexists I' \in \texttt{FI}$ with $\frac{supp(I')}{supp(I)} < 1 - \alpha$ and $I \subset I' \Rightarrow I \in \texttt{CFI}^\alpha$

[1] For the sake of brevity we write $C \cup i$ for $C \cup \{i\}$.



(a) (b)

Fig. 1. Itemset lattices for data from Table 1 and $\theta = 0.09$: (a) Frequent itemsets annotated with absolute support. (b) Closed itemsets and edges annotated with $\frac{supp(I')}{supp(I)}$ for $I \subset I'$.

The margin-closedness condition is surely met if there are no frequent supersets at all, as is the case for maximal itemsets. For $\alpha = 1$ the set of margin-closed itemsets is equal to the set of maximal frequent itemsets, since the margin extends all the way to a support of 0. The equality might also hold for some values $\alpha < 1$ depending on the minimum support θ and the particular item frequencies in the data. If the relative minimum support threshold $\frac{\theta}{supp(\emptyset)}$ exceeds the margin threshold $1 - \alpha$, the margin-condition can only be met by maximal frequent itemsets due to the following inequality for a frequent itemset I: $\theta \leq supp(I) = \frac{supp(I)}{supp(\emptyset)} < (1 - \alpha) * supp(\emptyset)$

The threshold α of margin-closedness prunes itemsets with very similar support to a superset. Our reasoning behind this is that larger itemsets are more specific descriptions of patterns but that patterns that occur in almost the same transactions are redundant. The number of reported patterns is thus reduced without having to raise the minimum support threshold or retreating to maximal frequent itemsets keeping a variety of frequent general and rare specific patterns.

We want to emphasize that the goal is not to approximate the frequency of patterns that are not reported, though this might be possible based on margin closed itemsets Cheng, Ke & Ng (2006), but rather to understand the structure of transaction datasets. Approaches that aim at frequency estimation usually favor shorter patterns to achieve higher compression ratios.

3. Mining margin-closed itemsets

In this section we describe our proposed DCI_MARGIN algorithm that modifies DCI_CLOSED to only report margin-closed itemsets given a threshold α. We first describe the post-processing Algorithm 1 that can be combined with any algorithm for closed itemsets mining. It demonstrates the basic principle used to determine the margin-closedness of a closed itemset. Then we integrate the margin-check into the

variant of DCI_CLOSED algorithm for dense datasets and add several pruning steps to obtain the final Algorithm 2.

Algorithm 1 tests the margin-condition for each closed frequent itemsets $C \in \texttt{CFI}$ using the TESTMARGIN function (Line 3). For all items $j \in \mathcal{I} \setminus C$ the support of the superset $C \cup j$ is calculated. If any superset is frequent and violates the margin condition (Line 8) the closed itemset is ignored, otherwise it can be reported as a margin-closed itemset (Line 12). Due to the monotonicity of support we only need to check supersets with one additional item.

Algorithm 2 is based on the DCI_CLOSED Lucchese et al. (2006*a*) algorithm that uses closure climbing and a vertical representation of the database. We first describe the inherited algorithmic steps briefly skipping the lines that deal with margin-closedness. The algorithm starts with C initialized to the bottom closure, i.e., the set of items present in all transactions (possibly the empty set). P initially contains all remaining frequent items and D and M are empty. The loop starting in Line 2 iterates over all items according to the total order $\prec$ (Line 3). Each item is removed from P (Line 4) and added to the current closed itemset C to obtain a closure generator C_i of an equivalence class. Line 6 checks whether the generator and thus the complete equivalence class is frequent and whether an equivalence class is entered that has already been visited by the algorithm (ISDUPLICATE). The duplicate check is performed by keeping track of two disjunct sets of items during the recursion (D and P): those that would generate equivalence classes that were already visited and those that would generate previously unseen equivalence classes. We refer the interested reader to Lucchese et al. (2006*a*) for more explanations and proofs of the duplicate check and pruning technique.

Algorithm 1 POSTMARGIN
(Mining all margin-closed frequent itemsets from all closed frequent itemsets.)

1: **PROCEDURE** POSTMARGIN(CFI)
2: **for all** $C \in \texttt{CFI}$ **do**
3: TESTMARGIN(C);
4: **end for**
5: **END PROCEDURE**
6: **PROCEDURE** TESTMARGIN(C)
7: **for all** $j \in \mathcal{I} \setminus C$ **do**
8: **if** $supp(C \cup j) \geq \theta \wedge \frac{supp(C \cup j)}{supp(C)} \geq (1 - \alpha)$ **then**
9: **return**
10: **end if**
11: **end for**
12: **print** C;
13: **END PROCEDURE**

If the itemset C_i passed the two tests in Line 6, the unique supremum $\hat{C}$ of the equivalence class generated by C_i is found by adding all items that do not decrease the support (Line 13-14). All other items are collected in P_i (Line 16) and passed in the recursive function call in Line 20, as those items that will create generators of new equivalence classes when added to $\hat{C}$. All items i that were used to generate a new equivalence class are recorded in D, the so-called pre-set Lucchese et al. (2006*a*), to support the duplicate check (Line 24). All closed itemsets could be reported after line 18. All margin-closed itemsets could be reported by calling the TESTMARGIN function of Algorithm 1. A better performance can be achieved by integrating the pruning strategies described below.

Algorithm 2 DCIMARGIN
(Mining all margin-closed frequent itemsets.)

FUNCTION DCIMARGIN($C,\ D,\ P,\ M$)
while $P \neq \emptyset$ **do**
  $i \leftarrow min_{\prec}(P)$;
  $P \leftarrow P \setminus i$;
  $C_i \leftarrow C \cup i$;
  if $(supp(C_i) \geq \theta) \wedge (\neg IsDuplicate(C_i,\ D))$ **then**
    if $\frac{supp(C_i)}{supp(C)} \geq 1 - \alpha$ **then**
      $M \leftarrow M \cup i$;
    end if
    $\hat{C} \leftarrow C_i$;
    $P_i \leftarrow \emptyset$;
    for all $j \in P$ **do**
      if $g(C_i) \subseteq g(j)$ **then**
        $\hat{C} \leftarrow \hat{C} \cup j$;
      else
        $P_i \leftarrow P_i \cup j$;
      end if
    end for
    $\hat{M} \leftarrow \emptyset$;
    DCIMARGIN($\hat{C},\ D,\ P_i,\ \hat{M}$);
    if $\hat{M} = \emptyset$ **then**
      TESTMARGIN($\hat{C}, D$);
    end if
    $D \leftarrow D \cup i$;
  end if
end while
END PROCEDURE
FUNCTION ISDUPLICATE($C_i,\ D$)
for all $j \in D$ **do**
  if $g(C_i) \subseteq g(j)$ **then**
    return true;
  end if
end for
return false;
END FUNCTION
PROCEDURE TESTMARGIN($\hat{C},\ D$)
for all $j \in D$ **do**
  if $supp(\hat{C} \cup j) \geq \theta \wedge \frac{supp(\hat{C} \cup j)}{supp(\hat{C})} \geq (1 - \alpha)$ **then**
    return ;
  end if
end for
print $\hat{C}$;
END PROCEDURE

Delay pruning If we test the margin condition of closed itemsets immediately upon their discovery after Line 18 we have to generate the transaction list of $\hat{C} \cup j$ for all $j \in \mathcal{I} \setminus \hat{C}$. Some of these itemsets are also used in the recursive call in Line 20 to obtain generators of C (instantiated with $\hat{C}$) in Line 5. By delaying the margin test until after the recursion (Line 22) we can utilize these results. We keep track of all items that generate supersets of C which violate the margin condition in the set M (Lines 7-9). When returning from the recursive call in Line 20 these items are available in $\hat{M}$. If any generator violated the margin-condition (Line 21) we do not need to call TESTMARGIN. Not all items in P_i passed to the recursive call are tested in Line 7 because of the conditions in Line 6. A generator excluded by the first condition is infrequent and can therefore not violate the margin condition. A generator excluded by the second condition is a duplicate, i.e., the corresponding transaction list is a subset of the transaction list of a previously tested generator. If the previous test did not violate the margin condition, this generator does not need to be tested because it has a smaller or equal support. The margin tests is thus performed for all necessary items in P_i during the recursion. If $\hat{M}$ is empty after the recursion we still need to check the margin-condition for all items in D that generate equivalence classes derivable from the current closed itemset.

Preset pruning After calling DCI_MARGIN recursively we only need to consider items from the current D for the margin-test in Line 22. Since $\hat{M}$ is empty we already know that none of the items in P_i generate violating supersets. The items in $P \setminus P_i$ are already included in $\hat{C}$, the current closed itemset under study. This leaves any items that were removed from P in Line 4 but not added to D in Line 24. We will now explain why these items can be omitted from the margin-test as well. We are only concerned with the case where none of the items in D has already violated the margin-condition, i.e., $\forall j \in D\ supp(C \cup i \cup j) < (1-\alpha) supp(C \cup i)$. We can further ignore items that produce infrequent generators and are filtered by the first condition in Line 6 for this computed closure. These would also be ignored in the margin-test in Line 38. This leaves items that produce a frequent generator but have been excluded by the duplicate check. Those can be omitted since those generators are covered by items from D.

Support order pruning Recall that any total order $\prec$ of the items can be used. DCI_CLOSED works with any fixed order of the items, sorting the items by decreasing support was used in Lucchese et al. (2006*a*). When used with DCI_MARGIN it increases the probability that the intersection of transaction lists for the support calculation of $\hat{C} \cup j$ in Line 38 results in a violation of the margin-condition avoiding further checks. This is especially helpful in dense datasets. Note that this optimization will improve the post-processing algorithm as well.

We refer to Lucchese et al. (2006*a*) for the proof that all returned itemsets are closed. We only need to show that the additional pruning steps do not remove any margin-closed itemsets from the results.

To prove the correctness and completeness of DCI_MARGIN we need to check whether the delayed pruning decisions for items in P_i for the subsequent call of DCI_MARGIN are correct. Since $P_i \cup \hat{C} \cup D = \mathcal{I}$ is not guaranteed, we also need to show that all items pruned by DCI_CLOSED will not affect the margin-decision for subsequent calls. We omitted the proof that there is no need to test infrequent generators for the sake of brevity.

Lemma 3.1. Given a closed itemset $C \in$ `CFI` and items $i, j \in \mathcal{I}$ with $i, j \notin C$, if $g(C \cup i) \subseteq g(j) \Rightarrow \frac{supp(C \cup i)}{supp(C)} \leq \frac{supp(C \cup j)}{supp(C)}$.

Proof: $g(C \cup i) \subseteq g(j) \Rightarrow g(C \cup i) = g(C \cup i \cup j) \Rightarrow g(C \cup i) \subseteq g(C \cup j)$ since $g(C \cup i \cup j) \subseteq g(C \cup j) \Rightarrow supp(C \cup i) \leq supp(C \cup j) \Rightarrow \frac{supp(C \cup i)}{supp(C)} \leq \frac{supp(C \cup j)}{supp(C)}$.

The Lemma states, that given a generator $C \cup i$ and an additional item j with $g(C \cup i) \subseteq g(j)$, we can assume that either the generator $C \cup j$ violates the margin-condition or if not neither will $C \cup i$. This allows us to skip the check in line 8 for generators which return true for the duplicate-check as long as we use the generator $C \cup j$ for testing the margin-closedness of C. Note that this only covers items in P_i.

Lemma 3.2. Given an infrequent generator $C \cup i$ with $supp(C \cup i) < \theta$, where C is a closed itemset and $i \in \mathcal{I}$ with $i \notin C$ and another closed itemset C' with $C \subset C'$ which is frequent, i.e., $supp(C') \geq \theta$, then $supp(C' \cup i) < \theta$.

Proof: $supp(C \cup i) < \theta \Rightarrow supp(C \cup C' \cup i) < \theta \Rightarrow supp(C' \cup i) < \theta$ since $C \cup C' = C'$.

This allows us to omit all generators $C \cup i$ for TESTMARGIN which were derived from a closed itemset C if $C \cup i$ is not frequent. Therefore we do not need to check the margin-condition and can ignore item i for all closed itemsets that are generated by adding items to C. In particular, we can avoid adding item i to D for subsequent tests in Line 24 if the generator $C \cup i$ is not frequent (Line 6).

Lemma 3.3. Given closed itemsets $C, C' \in \texttt{CFI}$ with $C \subset C'$ and items $i, h \in \mathcal{I}$ with $i, h \notin C'$ then: $g(C \cup i) \subseteq g(h) \Rightarrow \frac{supp(C' \cup i)}{supp(C')} \leq \frac{supp(C' \cup h)}{supp(C')}$

Proof: $g(C \cup i) \subseteq g(h) \Rightarrow g(C' \cup i) \subseteq g(h)$ since $g(C' \cup i) \subseteq g(C \cup i) \Rightarrow$[2] $\frac{supp(C' \cup i)}{supp(C')} \leq \frac{supp(C' \cup h)}{supp(C')}$

The Lemma states that pruning item i will not change the result for subsequent margin-checks, since the missing margin-calculation is still covered by item h. This allows us to avoid adding i to D in line 24 if the ISDUPLICATE check returns true and therefore using the pruning technique from DCI_CLOSED will not lead to incorrect results.

Altogether, Lemma 3.1 allows us to check only those items in P_i that pass the duplicate check if we check all items in D. Exploiting Lemma 3.2, we need to check only those items for a closed itemset which did not produce an infrequent generator for a closed subset. Finally, Lemma 3.3 states that the pruning technique used in DCI_CLOSED will not exclude items necessary for subsequent calculations.

For the proof of correctness and completeness of DCI_MARGIN we assume that exactly the set of all closed itemsets is presented to the modifications after line 18 due to completeness and correctness of DCI_CLOSED. We will therefore restrict the proof to itemsets in CFI.

Corollary 3.4. DCI_MARGIN is correct: only margin-closed itemsets are reported.

Proof: (by contradiction)

Let be $C \in \texttt{CFI} \setminus \texttt{CFI}^{\alpha}$. Suppose there exists an item i, s.t. $C \cup i \in \texttt{FI}$ and $\frac{supp(C \cup i)}{supp(C)} \geq 1 - \alpha$ and C is reported as margin-closed. This leaves two possibilities:

(a) If $i \in P_i \cup D$, we can conclude that $i \in P_i$, since for all $j \in D$ the margin-condition is tested in Line 38. Furthermore there must be an item $h \in D$, s.t. $g(C \cup i) \subseteq g(h)$, otherwise C would be checked in Line 8 and $\hat{M}$ would not be empty. This is a

[2] using Lemma 3.1

contradiction to Lemma 3.1, since C would be checked in TESTMARGIN for item h and therefore not be reported.

(b) $i \in \mathcal{I} \setminus (P_i \cup D)$ Since the algorithm is initiated with the bottom closure as closed itemset and the remaining items as P, i was pruned before by either (i) producing an infrequent generator or (ii) the duplicate-check returned true for the produced generator.

(i) Since the generator for which item i was pruned was infrequent, so is $C \cup i$ (according to Lemma 3.2). This is a contradiction to our assumptions.

(ii) i was not included in D for a closed itemset $C' \subset C$ due to the duplicate detection, which means there exists an item $h \in D$, s.t. $g(C' \cup i) \subseteq g(h)$. Then the closed but not margin-closed itemset would not be reported due to the check in TESTMARGIN in Line 38, since $C \cup i$ will violate the margin-condition according to Lemma 3.3.

Corollary 3.5. DCI_MARGIN is complete: all margin-closed itemsets are reported.

Proof: Let be $C \in \texttt{CFI}^{\alpha}$. Then we know that $\forall i \in \mathcal{I} \setminus C$ with $C \cup i \in \texttt{FI}$: $\frac{supp(C \cup i)}{supp(C)} < 1 - \alpha$. Since $P_i \cup D \subset \mathcal{I}$, we can deduce that (a) $\forall j \in P_i$ with $C \cup j \in \texttt{FI}$ the generator $C \cup j$ is tested and $\hat{M}$ will be empty in line 21 and (b) $\forall j \in D$ with $C \cup j \in \texttt{FI}$ the test in line 38 will lead to write out C. Given that all closed itemsets are traversed we can conclude that all margin-closed itemsets are reported.

3.1. Example

For our example we choose a minimal support $\theta = 0.1$ and $\alpha = 0.1$ as the margin value. To avoid confusion we use lower case letters to represent the items.

Starting with the example of Figure 1(a) and following a lexicographic order, the algorithm is initiated with the empty set as bottom closure. The first item to add is a. It is removed from the set of items P which will be processed in subsequent branches of the recursion. Since it is frequent and no item is in the list D, it passes the duplicate check in line 6. In line 7 the margin condition is not violated regarding the previously found closed itemset $\emptyset$. In the loop in line 12 only c and d can be added to the working set $\hat{C}$ since an addition of b would decrease the support. The algorithm has reached the closed itemset acd invoking a recursive function call of DCI_MARGIN. Within this function call only b could be added, but since $abcd$ is infrequent no further processing is necessary after line 6. Since D is empty and no frequent closed superset in this branch exists, we can print out acd as a margin closed itemset by calling TESTMARGIN. Returning to the subsequent branch of the recursion, a is added to the duplicate list D.

Starting again with the empty set, b is added to the working set C_i. Since it is frequent and not a subset of a (line 6), d can be added to climb to the closure bd. In line 7, the margin condition for the bottom closure $\emptyset$ regarding b is checked. After another call of DCI_MARGIN the newly created working set $C_i = bcd$ does not pass the frequency check. b is added to the duplicate list D in line 24 when returning to the next branch.

Again, calling DCI_MARGIN with the empty set, c is processed and found to be a closed itemset. Since the relative difference between $supp(\emptyset)$ and $supp(c)$ violates the margin condition, c is added to M indicating that the bottom closure is not margin-closed. The algorithm continues with processing the only item left d. Since $g(cd)$ is a subset of the first element in D, namely a, cd is correctly identified as an itemset of a previously visited equivalence class and therefore not added to D before pursuing the last branch. For c the TESTMARGIN procedure is called and it is found to be margin-closed, because none of the items in $D = \{a, b\}$ would result in violating the margin condition.

Finally, d is added in the last branch. Since no other item is left to add in line 12, the

Name	Items	Transactions
Accidents	468	340183
BMS-WebView-1	59602	149639
BMS-WebView-2	77512	358278
BMS-POS	515597	3367020
Chess	75	3196
Connect	129	67557
Kosarak	41270	990002
Mushroom	119	8124
Pumsb	2113	49046
Pumsb$*$	2088	49046
Retail	16470	88162
T10I4D100K	870	100000
T40I10D100K	942	100000

Table 2. FIMI datasets.

TESTMARGIN procedure is called to test the margin condition for the closed itemset d. Adding the first item a in D to the closed itemset d passes the frequency check but violates the margin condition in line 38. d is correctly identified as not margin-closed.

The algorithm correctly determines acd, bd and c as margin-closed. $abcd$ is infrequent and neither $\emptyset$ or d are margin-closed due to c and cd, respectively.

4. Experiments

We performed experiments to analyze the class of margin-closed itemsets in detail and to compare the proposed DCI_MARGIN algorithm with a naive extension of DCI_CLOSED. We do not compare experimentally with our own previous work Mörchen (2006) that used a variation of CHARM because this algorithm needs to keep all closed itemsets in memory for the subsumption check. This is a severe disadvantage in particular for low minimum support values. We performed a comparison with FP-Growth variant Cheng, Ke & Ng (2006) to demonstrate that the greedy heuristics lead to incorrect results that can vary greatly from the exact result generated by our algorithm.

The implementation was done in Java using bitmap data structures and the correctness was checked using brute force algorithms on small datasets. The experiments were run on a 64-bit dual core Intel Xeon with 2.66GHz and 8GB of main memory.

4.1. Data sets

We used datasets from three repositories. The FIMIGoethals & Zaki (2003) datasets listed in Table 2 include large transaction datasets derived from traffic data, census data, weblogs Kohavi et al. (2000) and retail data. The last two datasets are synthetically generated to simulate market basket data. The datasets from the UCI Machine Learning Repository Asuncion & Newman (2007) listed in Table 3 represent classification problems from a wide variety of domains. We used the itemset representations of the datasets taken from the LUCS repository Coenen (2003). The text datasets listed in Table 4 are shipped with the Cluto clustering toolkit Zhao & Karypis (2002) and were converted to itemsets using a binary representation of words in documents discarding the term frequencies.

Name	Items	Transactions
adult	97	48842
anneal	73	898
auto	137	205
breast	20	699
chessKRvK	58	28056
congres	34	435
connect4	129	67557
cylBands	124	540
dematology	49	366
ecoli	34	1389
flare	39	48842
glass	48	214
heart	52	303
hepatitis	56	155
horseColic	85	368
ionosphere	157	351
iris	19	150
led7	24	3200
letRecog	106	20000
mushroom	90	8124
nursery	32	12960
pageBlocks	46	5473
penDigits	89	10992
pima	38	768
soybean-large	118	683
ticTacToe	29	958
waveform	101	5000
wine	68	178
zoo	42	101

Table 3. UCI datasets.

Name	Items	Transactions
cacmcisi	41681	4663
classic	41681	7094
cranmed	41681	2431
fbis	2000	2462
hitech	126373	2301
k1a	21839	2340
k1b	21839	2340
la1	31472	3204
la12	31472	6279
la2	31472	3075
mm	126373	2521
new3	83487	9557
ohscal	11465	11161
re0	2886	1503
re1	3758	1656
reviews	7454	4069
sports	8261	8580
wap	8460	1559

Table 4. Text datasets.

4.2. Numerosity reduction

A parameter study was performed on all datasets to investigate the redundancy reduction under various minimum support values and minimum margin between 0.005 and 1.0 depending on the dataset. Obviously redundancy can only be removed if there is any. In the interest of space we only show typical examples of datasets where our approach is beneficial and examples for dataset where only few redundancy exists among the closed itemsets.

Figure 2 shows the number of reported itemsets on a log scale vs. the minimum support threshold for some FIMI datasets. The solid line corresponds to all closed itemsets found by DCI_CLOSED (or equivalently $\alpha = 0$). The dashed lines represent DCI_MARGIN with different minimum margin thresholds. The lowest dashed line represents all maximal itemsets ($\alpha = 1.0$).

For all datasets the margin condition can reduce the number of reported itemsets significantly without necessarily reporting only maximal itemsets. In many cases we observe a smooth transition of the curves showing a decreasing number of patterns with increasing α.

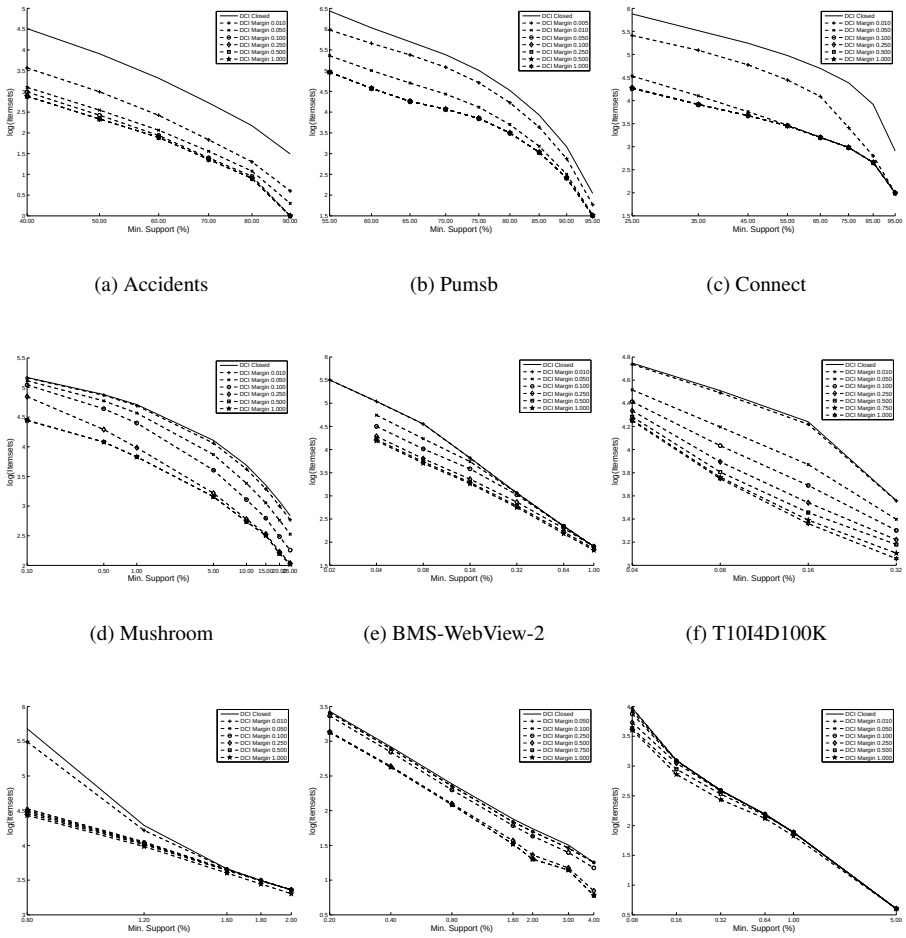
For Accidents and Pumsb very small margins of 0.01 and 0.005, respectively, already largely reduce the number of itemsets indicating a large number of closed itemsets with very similar frequencies in these datasets. Since these datasets have relatively high minimum support levels, the number of margin-closed itemsets approaches the number of maximal itemsets for values of $\alpha = 0.05$ or larger. For Accidents margins of $0.01 - 0.05$ seem adequate because smaller values do not decrease the number of reported patterns and larger values are equivalent to the maximal itemsets. For Pumsb smaller thresholds up to 0.01 represent a good compromise.

For Mushroom, BMS-WebView-2, and T10I4D100K the margins of 0.01 and 0.05 lead to a significant reduction. Larger margins continue to reduce the number of patterns in a relatively smooth transition.

T40I10D100k, Retail, and BMS-WebView-1 show a different behavior. For the Retail data margins up to 0.25 reduce the number of reported itemsets but by far not as clearly as for the other datasets. There does not seem to be a lot of redundancy which could be caused by the large number of distinct items. Margins between 0.1 and 0.5 would be useful to reduce the size of the result. For the T40I10D100k data the number of margin-closed itemsets is very close to the number of maximal itemsets for $\alpha \geq 0.05$ and small minimum supports. For larger minimum support thresholds the number of closed and maximal itemsets are much more similar with margin-closed itemsets providing a better transition.

Figure 3 shows the number of reported itemsets for some UCI datasets. Adult is an example for cases where increasing the margin threshold leads to a gradual reduction of patterns. Note that the interplay of support and margin can lead to non-monotone curves, but in general the larger the minimum support and the larger the minimum margin, the less patterns are reported. Hepatitis is an example where the redundancy removal is very effective for low minimum supports but vanishes for larger values. Again, this is due to the large absolute values of the minimum support of 60%-70% that does not leave a lot of room for any margin of support. For ChessKRvK using a margin constraint does not change the result significantly because no redundancy could be detected.

Figure 4 shows the number of reported itemsets for some text datasets. The first two datasets are examples for document collections where the margin constraint can successfully remove redundancy in the word combination patterns. The last dataset is an example for very concise sets of patterns where even large margins do not reduce the number of patterns.



(a) Accidents (b) Pumsb (c) Connect

(d) Mushroom (e) BMS-WebView-2 (f) T10I4D100K

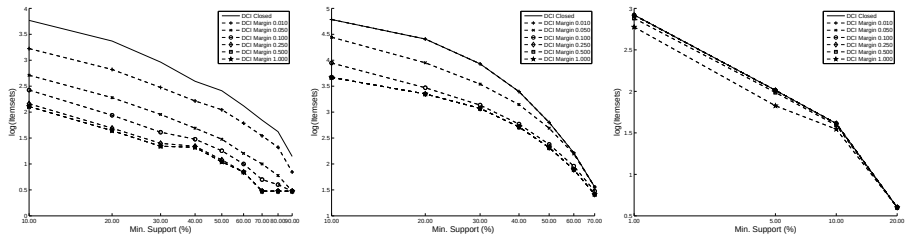
(g) T40I10D100K (h) Retail (i) BMS-WebView-1

Fig. 2. Number of (margin-)closed itemsets for different minimum support and minimum margin in FIMI datasets.

4.3. Computational complexity

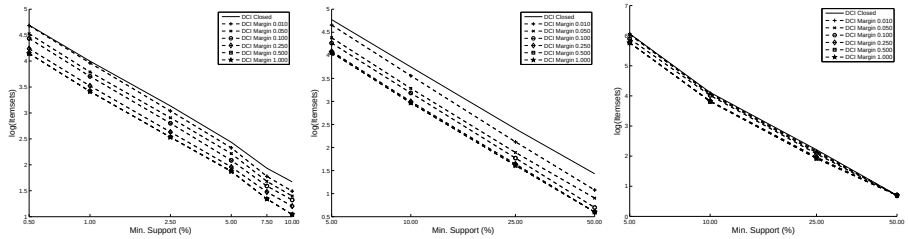
We compare DCI_MARGIN with a naive version of mining margin-closed patterns that adds a post-processing check to DCI_CLOSED to evaluate the efficiency of our pruning steps. The post processing extends each closed pattern by all items not in the pattern and checks the margin condition. This post-processing takes advantage of the support order pruning, but not delay or preset pruning.

The algorithms can be implemented efficiently using bitmaps to store the transactions for each item. The high level operations on the bitmaps include logical AND (when adding an item to an itemset), subset (checking if an item can be added to an itemset without decreasing the support) and cardinality (determining the support of an itemset).



(a) adult (b) hepatitis (c) chessKRvK

Fig. 3. Number of (margin-)closed itemsets for different minimum support and minimum margin in UCI datasets.



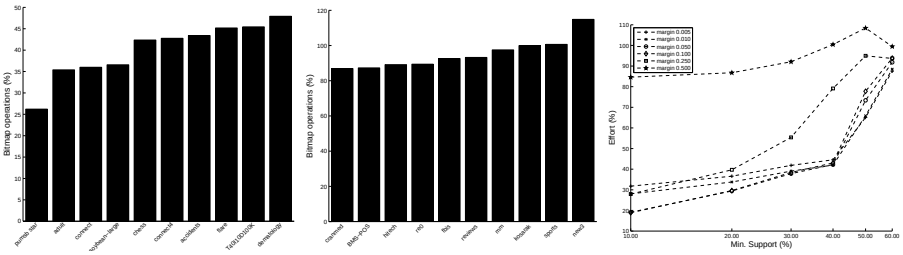
(a) classic (b) hitech (c) new3

Fig. 4. Number of (margin-)closed itemsets for different minimum support and minimum margin in text datasets.

In order to be independent of influences of operating system, programming language, and just-in-time compilers we evaluated the algorithms using counters for the bitmap operations that dominate the computational effort. All bitmaps are accompanied by a pointer to the position of the last set bit. All three high level operations use these pointer to return early. The subset operation is further aborted early as soon as a violating bit is discovered. We counted all low-level 64-bit word operations corresponding to these high level operations.

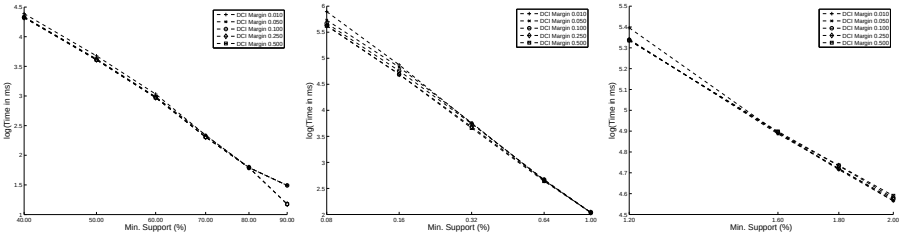
In Figure 5 we show the relative effort required by DCI_MARGIN compared to DCI_CLOSED with post processing. Values below 100% thus indicate that the pruning methods increased the efficiency for the particular choice of parameters and dataset. For each dataset we summed up the effort over all minimum support levels used in Section 4.2 and minimum margins 0.01 through 0.25 as would be typically used in practice. The ten best performances of DCI_MARGIN in Figure 5(a) requires only 25%-50% of the bit vector operations of DCI_CLOSED with post processing. The ten worst performances of DCI_MARGIN show the same or slightly worse performance than the post processing. More bit operations than post processing can be observed if many delay or preset pruning steps are performed unsuccessfully without removing closed itemsets. For the best dataset the effort is shown for each minimum support and minimum margin level in Figure 5(c).

The results indicate a clear benefit of the proposed pruning steps. The observed



(a) Best (b) Worst (c) Pumsb_star

Fig. 5. Relative number of bit operations of DCI_MARGIN compared to naive post processing.



(a) Accidents (b) BMS WebView 2 (c) T40I10D100K

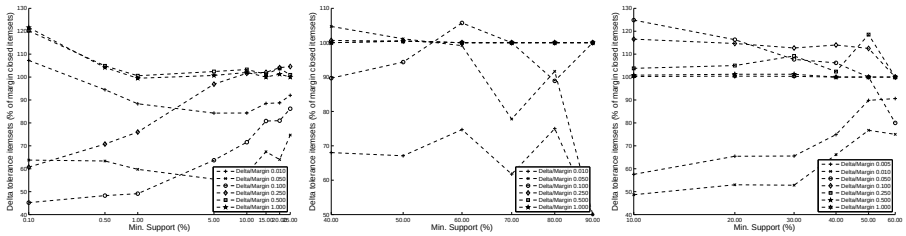
Fig. 6. Runtime of DCI_MARGIN for selected large datasets. For the most complex problem under study the mining took up to about 15min.

gains in effort highly depend on the dataset. In the best cases a significant amount of computation is saved and in the worst cases the performance is similar to post processing.

The space and time complexity of DCI_MARGIN are inherited from DCI_CLOSED Lucchese et al. (2006*a*). The space complexity is bounded by the size of the dataset and independent of the number of patterns found. No explicit computational analysis was given in Lucchese et al. (2006*a*) but the runtimes compared favorably with other efficient algorithms. Furthermore, our algorithm inherits important scalability properties from DCI_CLOSED the possibility to parallelize the search space traversal Lucchese et al. (2007) and mine the data out of core Lucchese et al. (2006*b*). In Figure 6 we show the actual run times for the largest datasets and several minimum support and margin thresholds. For very low minimum supports on the large dataset BMS WebView 2 the mining took about 15 minutes. As for all itemset algorithms the runtime increases exponentially with lower minimum supports. Different values of the margin have relatively small influence on the runtime.

4.4. δ-tolerance itemsets

We obtained the binary version of the algorithm in Cheng, Ke & Ng (2006) and compared the output with the complete set of margin closed itemsets given the same min-



(a) Mushroom (b) Accidents (c) Pumsb∗

Fig. 7. Number of reported δ-tolerance itemsets as percentage of the correct number of δ-tolerance closed (or equivalently margin closed) itemsets.

imum support and threshold (our margin and their δ). The results for three datasets evaluated are shown in Figure 7. In many cases the algorithm is observed to report a significant number of non margin closed itemsets (greater than 100% on the x axis) or to report less than the true number of margin closed itemsets (less than 100%) ranging from under 50% to more than 120%. There is no clear trend in the behaviour of the algorithm with respect to miminum support or margin. While for large minimum support thresholds the absolute difference in reported itemsets is small, for more complex settings the difference can be huge: For the lowest minimum support and margin on Pumsb∗ only 133490 of all 231882 margin closed itemsets are reported. This demonstrates the greedy heuristic can lead to results that vary significantly from the correct result reported by our algorithm.

5. Applications

In this section we demonstrate the usefulness of margin-closed itemsets in two applications. In exploratory analysis of temporal patterns the removal of redundancy generates better interpretable results. For compression based data mining tasks better understandable codebooks with comparable performance are generated.

5.1. Temporal Data Mining

In Mörchen & Ultsch (2007) temporal patterns based on the Time Series Knowledge Representation (TSKR) are mined from symbolic interval time series. The data model for this method consists of possibly overlapping time intervals with a label. Hierarchical patterns are defined as groups of intervals simultaneously active on a subinterval (Chords) on the first level, and a partial order of such groups (Phrases) on the second level. Figure 8a shows a series of observed intervals with labels A, B, and C and intervals of Chord patterns such as ABC beneath. Figure 8b shows a Phrase that represents a partial order of Chords matching both blocks of intervals in Figure 8a. The mining of TSKR patterns can be formulated as a combination of itemset and sequential pattern mining in three phases: 1) Closed itemsets are mined interpreting the interval labels as items and time units as transactions. 2) Sequential patterns are mined interpreting Chords as items and intervals between any Chord start and end point as transactions in a sequence. 3) Closed itemsets are mined interpreting sequential patterns as items and

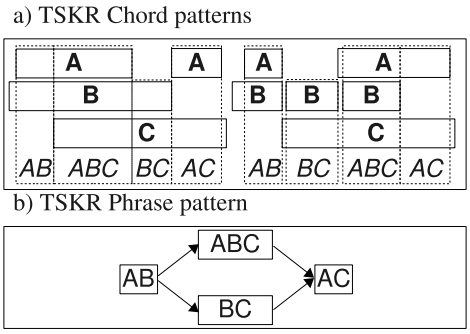


Fig. 8. Simultaneous occurring intervals (Tones) form Chords. Phrases describe a partial order of Chords.

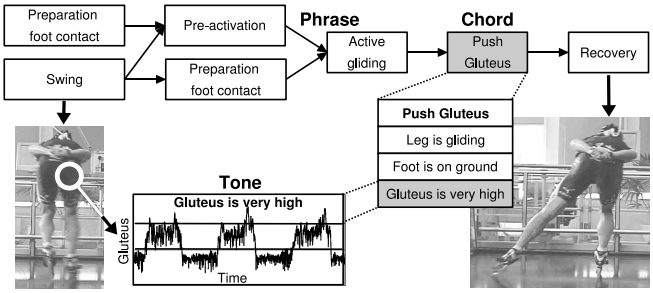


Fig. 9. Detailed Phrase of skating data with additional information on muscle activation.

sequences as transactions. Finally, each closed group of sequential patterns is converted to a partial order graph Garriga (2005).

The support of a Chord is the total number of time units where the intervals are observed simultaneously. Small differences in support are typically meaningless, such as the short leading or trailing parts of observed intervals not described by a Chords in Figure 8a. Margin-closed itemsets can be used to mine a smaller set of Chords with less redundancy. Chords with almost the same duration than more specific Chords are pruned. This leads to better interpretable results and reduces the complexity of the sequential pattern mining algorithm. Groups of simultaneous sequential pattern form a closed partial order Garriga (2005). Again, margin-closedness leads to a reduction of the reported partial orders pruning less specific patterns that are observed in only few additional sequences.

In an application to sports medicine using a minimum margin of 0.1 reduced number of Chords from 60 to 18 and the number of Phrases from 20 to 15. The absolute numbers are small but they significantly eased the burden of the manual analysis by an expert. Having to analyze many very similar patterns can easily result in frustration of the analysts. The expert selected the Phrase in Figure 9 as the most interesting pattern that describes the muscle activation during inline speed skating. The Chord *Push Gluteus* is expanded to show the corresponding items and for the item *Gluteus is very high* the represented value range in the original numerical time series is shown.

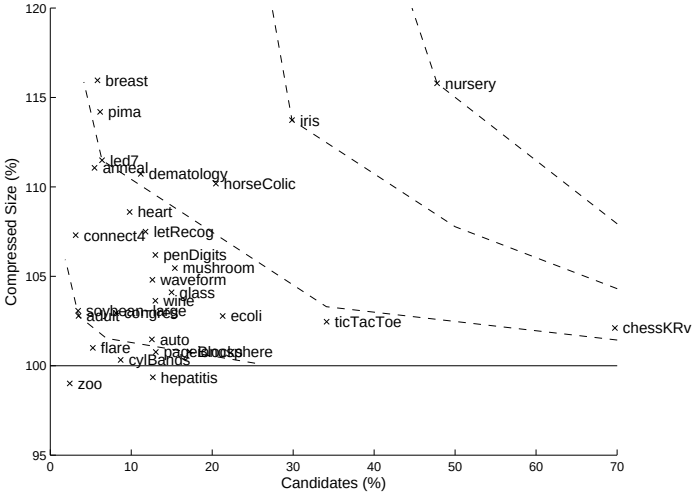


Fig. 10. Relative number of candidates vs. relative size of compressed database when using margin-closed itemsets instead of closed itemsets as candidates for Krimp.

5.2. Mining by compression

Performing data mining tasks using compression as an approximation to Kolmogorov complexity has recently gained popularity, see Faloutsos & Megalooikonomou (2007), Keogh et al. (2007) and references therein. For itemsets the Krimp algorithm has been proposed to find a codebook of itemsets that compresses a transaction database well Siebes (2006). The algorithm has subsequently been used to support classification van Leeuwen et al. (2006), change detection Van Leeuwen & Siebes (2008), and missing value replacement Vreeken & Siebes (2008).

Krimp starts with a trivial code book of single items that is greedily improved considering longer itemsets from a set of candidate patterns. The authors recommend to use closed itemsets rather than frequent itemsets to avoid redundancy. We performed a set of experiments to evaluate the use of margin-closed itemsets as candidates.

We ran the Krimp algorithm with closed and margin-closed itemsets as candidates and compared the sizes of the compressed database. We used the smallest minimum support values and the same minimum margin values as in Section 4.2. Only the UCI datasets were used because compression is particularly relevant for classification and the available version of the Krimp program had problems processing the high dimensional sparse text datasets.

Figure 10 plots the ratio of candidates vs. the ratio in achieved compression. The ratios are calculated comparing the margin-closed itemsets with closed itemsets as the baseline. For example, for the TicTacToe data using only 35% of the closed itemsets as candidates results in a compressed database that is only 2.5% bigger. For the sake of presentation we show only the best result over all evaluated margin values for most datasets. The best result for each dataset was determined using the minimum sum of the candidate ratio and compression ratio, i.e., using the Manhattan distance from $(0, 100)$.

For many datasets a significant reduction in the number of candidates does not hurt the compression. This directly translates into much improved runtime of the Krimp algorithms that uses the candidate itemsets to build a codebook that compresses the data best. A larger number of datasets has less than 10% increase in compression with fewer

than 30% of the closed itemsets. In some cases we can even improve the compression ratio. Most notably, for the Zoo dataset we achieve a slightly better compression using less than 5% of the closed itemsets. The dotted lines passing through the best results for Nursery, Iris, Led7, and Adult show the candidate and compression ratios for multiple margin values applied to these datasets. This shows how different margin values provide a trade off between candidate size and compression.

The compression results show that a lot of redundancy can be removed without compromising the quality of the codebook. The reduced number of candidates speeds up the codebook generation of Krimp and makes the codebook more interpretable. For example, to understand why a certain instance has been labeled by the Krimp-based classifier in a particular way one can extract the codebook vectors that had the largest influence on the decision. We expect that the performance of subsequent data mining tasks will not suffer significantly because comparable compression ratio are obtained.

6. Discussion

Our experimental results show that with comparable run time our algorithm can mine the more compact set of all margin-closed itemsets instead of reporting all closed itemsets. The pruning is performed on-the-fly utilizing the data structures of DCI_CLOSED and saving IO costs otherwise required to report all closed itemsets. The best value of α is application dependent: which difference in support between similar patterns can be considered insignificant enough to report only the longer pattern? In most cases we assume that α would be small but not very small (between 0.01 and 0.2). For very small α one should expect increased run time because less pruning can be performed. If the margin is chosen to be bigger than $1 - \theta$ only maximal itemsets are reported. In some applications it might be more natural to specify an absolute support margin. All our results hold and the algorithm can be used in the same way.

In principle the test for margin-closedness can also be integrated in the FP-Tree Han et al. (2000) algorithms or the version of DCI_CLOSED for sparse datasets. We chose the DCI_CLOSED algorithm for dense datasets as the basis of our work for several reasons. The vertical data format used by DCI algorithms can be exploited with the SIMD architecture of modern processors and even GPUs. In addition, the found patterns do not need to stay in memory and the partition of the search space enables parallelization Lucchese et al. (2007). The vertical representation has further advantages for itemset problems that represent temporal data Mörchen & Ultsch (2007) which is typically dense. Checking additional constraints on the duration of temporal patterns can be easily done using bit vectors but would require tracking the transaction times in projected databases Han et al. (2000).

A breadth-first approach might be more suitable for sparse datasets Yahia et al. (2006). Since the number of closed itemsets is commonly much smaller in this case the margin-condition can be checked after the closed itemset computation.

The concept of a minimum margin could also be used to generalize the definition of minimal generators Li et al. (2006), the minimal elements of an equivalence class induced by the closure operator. A minimal generator is an itemset where no item can be removed without increasing the support. A margin minimal generator would be an itemset where no itemset can be removed without increasing the support significantly (given a threshold parameter).

Our aim was to avoid redundancy of reported patterns to support exploratory analysis and favor longer patterns with more explanatory power. The concept of margin-closedness is in no way limited to itemsets, it can also be applied to sequential patterns

Agrawal & Srikant (1995), partial orders Pei et al. (2006), Mörchen & Ultsch (2007) and graphs Kuramochi & Karypis (2001).

7. Related work

The two closest publications to our approach are δ-tolerance itemsets Cheng, Ke & Ng (2006) and relaxed frequent closed itemsets Song et al. (2007).

The δ-tolerance closed itemsets of Cheng, Ke & Ng (2006) are equivalent in definition to margin-closed itemsets and have been proposed independently. The motivation in Cheng, Ke & Ng (2006) was to provide a condensed itemset representation that provides an approximate frequency estimation for itemsets. This is achieved with approximation formulas that use the support and the support differences (margins) of the itemsets stored in an FP-tree Han et al. (2000). The mining algorithm uses several heuristics that try to avoid but do not guarantee false dismissals. The reported itemsets are thus possibly a subset of all margin-closed itemsets. As demonstrated this does not seem to hurt the frequency estimation and enables fast performance. In contrast, we can guarantee completeness which is important for exploratory analysis. While not designed for frequency estimation, the same techniques as proposed in Cheng, Ke & Ng (2006) are applicable to our approach.

The relaxed frequent closed itemsets of Song et al. (2007) require the user to define a uniform partition of the support range. Subsets whose supersets are in the same support interval are pruned removing redundancy. The motivation is to reduce the number of patterns in memory when mining data streams. The effectiveness and efficiency was demonstrated using synthetic data. The a priori definition of several support thresholds might still generate redundant patterns if the supports of the subset and superset are just below and above one of the support thresholds, respectively. In contrast our pruning is data driven and removes any redundancy according to the single threshold α.

In comparison with the two approaches outlined above we performed much more extensive experiments with a total of 60 datasets from many different domains whereas Cheng, Ke & Ng (2006) and Song et al. (2007) used only three and one FIMI datasets, respectively.

In our previous work we have presented a modified CHARM Zaki & Hsiao (2002) algorithm to mine margin-closed itemsets Mörchen (2006). This approach suffers from scalability problems because it requires all closed (even the non-margin closed) itemsets to be kept in memory for the subsumption check.

We proceed to categorize less directly related approaches below by the purpose they have been designed for to highlight the differences to our approach.

Condensed itemset representations Calders et al. (2006) have been developed to derive the support of all itemsets from a compact summary exactly Bykowski & Rigotti (2001), Kryszkiewicz (2001), Calders & Goethals (2002, 2003), Muhonen & Toivonen (2006), Calders & Goethals (2007), Liu et al. (2007) or approximately Pei et al. (2002), Boulicaut et al. (2003), Cheng, Ke & Ng (2006). Querying the support of an itemset from a data structure is a key step in generating association rules Hipp et al. (2000). The basic idea of non-derivable itemsets Calders & Goethals (2007) and related approaches is to derive the support of a query itemset from the support of subsets stored in the condensed representation Calders & Goethals (2002, 2003). If this is possible exactly or within error bounds, the larger set does not need to be stored in the summary. Pudi & Haritsa (2003) prunes all supersets with approximately the same support as a smaller itemset. Note that these approaches favor short itemsets and prune longer itemsets whereas we prune the shorter subsets with support similar to a longer supersets.

This favors more detailed patterns that are generally more interesting in exploratory analysis.

Condensed representations are a special case of more general constraints on the reported itemsets that are commonly categorized into several classes Srikant et al. (1997), Ng et al. (1998): monotone, anti-monotone, succinct, convertible and tough. The first three were integrated in early constraint based itemset mining algorithms. Convertible constraints were later integrated for depth first algorithms in Pei, Han & Lakshmanan (2001) and for level-wise algorithms in Bonchi & Lucchese (2005), see Bonchi & Lucchese (2007) for more details. Bonchi & Lucchese (2006) describes issues with combining closed itemsets (and other condensed representations) with additional constraints. Our margin constraint does not cut closure equivalence classes but simply merges them avoiding potential problems. Recently, De Raedt et al. (2008) presented an elegant way of mining constrained itemsets, including margin-closed itemsets, with constraint programming.

A related line of work is motivated by the fact, that transaction data is often noisy. The strict definition of support, requiring all items of an itemset to be present in a transaction, is relaxed Pei, Tung & Han (2001), Yang et al. (2001), Seppänen & Mannila (2004), Afrati et al. (2004), Yan et al. (2005), Liu et al. (2005), Cheng, Yu & Han (2006), Uno & Arimura (2007), Calders et al. (2007), Cheng et al. (2008). A recent comparison analyzed the efficiency and effectiveness of approximate itemset mining Gupta et al. (2008). These approaches can reveal important structures in noisy data that might otherwise get lost in a huge amount of fragmented patterns. One needs to be aware though that they report approximate support values and possibly list itemsets that are not observed as such in the collection at all Afrati et al. (2004) or with much smaller support. This might be misleading in exploratory applications. In our application of itemset mining to temporal data mining Mörchen & Ultsch (2007) we filtered out noise in preprocessing steps using the temporal structure of the data and found it beneficial to list exact patterns with exact support. This corresponds to the Gricean maxim of quality Grice (1989) that states that only well supported facts and no false descriptions should be reported and has been recommended as a guideline for pattern discovery for data exploration Sripada et al. (2003). Finally, we want to note that margin-closed itemsets might be used instead of closed-itemsets as seeds to the AC-Close algorithm for approximate itemset mining Cheng, Yu & Han (2006) improving its efficiency that was criticized in Gupta et al. (2008).

Other approaches try to reduce the number of patterns after they are mined Afrati et al. (2004), Xin et al. (2005), Mielikäinen (2005). By this time a lot of computational resources have been spent on mining and storing the results. Our algorithm integrates the mining with the pruning on-the-fly and never stores or further processes the superfluous patterns.

For post processing techniques such as Bringmann & Zimmermann (2009) or Siebes (2006) that use closed itemsets as their input and remove redundancy in the pattern set, margin-closed itemsets can be used as an alternative input reducing their runtime without scarificing performance. This was demonstrated for Siebes (2006) in Section 5.2. In Bringmann & Zimmermann (2009) a small subset of patterns is selected that preserves much of the transaction partition collectively induced by presence and absence of a set of patterns. Patterns are selected according to a user defined ordering such as by size or by support.

In Geerts et al. (2004) the number of reported closed itemsets is reduced to the top-k patterns optimizing the coverage of the database with the transactions and items of the patterns. This is likely to remove redundancy in the output but our constraint is more explicit. It would be interesting to investigate the top-k least redundant pattern mining

problem. In Boley & Grosskreutz (2009) the number of frequent (but not closed) itemsets given a minimum support is estimated using random walks on the itemset lattice.

In addition to the margin constraint, statistical measures for interestingness, significance, and surprise Malik & Kender (2006), Tatti (2007), Gallo et al. (2007), Webb (2007), Tatti (2008) could be used to rank or further reduce the number of reported margin-closed itemsets.

In summary, margin-closedness is a stricter constraint than closedness that leads to a lossy, concise, exact itemset representation designed for exploratory and explanatory data mining tasks.

8. Summary

Margin-closed itemsets provide a compromise between closed and maximal itemsets designed for exploratory data analysis favoring longer itemsets that provide the users with more specific information and reporting exact information. We have presented DCI_MARGIN, a new efficient algorithm that mines *all* margin-closed itemsets on-the-fly and proved its correctness and completeness. Compared to closed itemset mining the algorithm can largely reduce the number of reported itemsets depending on the redundancy structure of the dataset under study. The algorithm achieves this with small computational overhead and was experimentally shown to have comparable or better speed than DCI_CLOSED. We show the usefulness of the patterns in two applications: exploratory mining for temporal patterns Mörchen & Ultsch (2007) and finding compressing datasets Siebes (2006) that are useful for classification, change detection, or missing value replacement.

9. Acknowledgments

We thank Matthijs van Leeuwen and James Cheng for sharing their software and Philipp Hussels for helping to run it. We acknowledge Blue Martini Software for contributing the KDD Cup 2000 data.

References

Afrati, F., Gionis, A. & Mannila, H. (2004), Approximating a collection of frequent sets, *in* 'Proc. 10th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining', ACM Press, pp. 12–19.

Agrawal, R., Imielinski, T. & Swami, A. N. (1993), Mining association rules between sets of items in large databases, *in* 'Proc. ACM SIGMOD Intl. Conf. on Management of Data', ACM Press, pp. 207–216.

Agrawal, R. & Srikant, R. (1995), Mining sequential patterns, *in* P. S. Yu & A. S. P. Chen, eds, 'Proc. 11th Intl. Conf. on Data Engineering', pp. 3–14.

Asuncion, A. & Newman, D. (2007), 'UCI machine learning repository'.
URL: *http://www.ics.uci.edu/∼mlearn/MLRepository.html*

Beil, F., Ester, M. & Xu, X. (2002), Frequent term-based text clustering, *in* 'Proc. 8th Intl. Conf. on Knowledge Discovery and Data Mining', pp. 436–442.

Boley, M. & Grosskreutz, H. (2009), 'Approximating the number of frequent sets in dense data', *Knowledge and Information Systems* **21**(1), 65–89.

Bonchi, F. & Lucchese, C. (2005), Pushing tougher constraints in frequent pattern mining, *in* 'Proc. Pacific-Asia Conf. on Knowledge Discovery and Data Mining', pp. 114–124.

Bonchi, F. & Lucchese, C. (2006), 'On condensed representations of constrained frequent patterns', *Knowledge and Information Systems* **9**(2), 180–201.

Bonchi, F. & Lucchese, C. (2007), 'Extending the state-of-the-art of constraint-based pattern discovery', *Data Mining and Knowledge Discovery* **60**(2), 377–399.

Boulicaut, J.-F. & Bykowski, A. (2000), Frequent closures as a concise representation for binary data mining, *in* 'Proc. Pacific-Asia Conf. on Knowledge Discovery and Data Mining', pp. 62–73.

Boulicaut, J.-F., Bykowski, A. & Rigotti, C. (2003), 'Free-sets: A condensed representation of boolean data for the approximation of frequency queries', *Data Mining and Knowledge Discovery* **7**(1), 5–22.

Bringmann, B. & Zimmermann, A. (2009), 'One in a million: picking the right patterns', *Knowledge and Information Systems* **18**(1), 61–81.

Bykowski, A. & Rigotti, C. (2001), A condensed representation to find frequent patterns, *in* 'Proc. 20th ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems', ACM Press, pp. 267–273.

Calders, T. & Goethals, B. (2002), Mining all non-derivable frequent itemsets, *in* 'Proc. 6th European Conf. on Principles of Data Mining and Knowledge Discovery', Springer, pp. 74–85.

Calders, T. & Goethals, B. (2003), Minimal k-free representations of frequent sets, *in* 'Proc. 7th European Conf. on Principles and Practice of Knowledge Discovery in Databases', Springer, pp. 71–82.

Calders, T. & Goethals, B. (2007), 'Non-derivable itemset mining', *Data Mining and Knowledge Discovery* **14**(1), 171–206.

Calders, T., Goethals, B. & Mampaey, M. (2007), Mining itemsets in the presence of missing values, *in* 'Proc. Intl. Symp. on Applied Computing', ACM, pp. 404–408.

Calders, T., Rigotti, C. & Boulicaut, J.-F. (2006), A survey on condensed representations for frequent sets, *in* 'Constraint-Based Mining and Inductive Databases', pp. 64–80.

Cheng, H., Yan, X., Han, J. & Hsu, C. (2007), Discriminative frequent pattern analysis for effective classification, *in* 'Proc. IEEE Intl. Conf. on Data Engineering', pp. 716–725.

Cheng, H., Yu, P. S. & Han, J. (2006), AC-Close: Efficiently mining approximate closed itemsets by core pattern recovery, *in* 'Proc. IEEE Intl. Conf. on Data Mining', IEEE, pp. 839–844.

Cheng, H., Yu, P. S. & Han, J. (2008), Approximate frequent itemset mining in the presence of random noise, *in* 'Soft Computing for Knowledge Discovery and Data Mining', Springer, pp. 363–389.

Cheng, J., Ke, Y. & Ng, W. (2006), δ-tolerance closed frequent itemsets., *in* 'Proc. 6th IEEE Intl. Conf. on Data Mining', IEEE Press, pp. 139–148.

Coenen, F. (2003), 'The LUCS-KDD discretised/normalised ARM and CARM data library'.
URL: *http://www.csc.liv.ac.uk/∼frans/KDD/Software/LUCS_KDD_DN/*

De Raedt, L., Guns, T. & Nijssen, S. (2008), Constraint programming for itemset mining, *in* 'Proc. 14th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining', ACM, pp. 204–212.

Faloutsos, C. & Megalooikonomou, V. (2007), 'On data mining, compression, and kolmogorov complexity', *Data Mining Knowledge Discovery* **15**(1), 3–20.

Fung, B., Wang, K. & Ester, M. (2003), Hierarchical document clustering using frequent itemsets, *in* 'Proc SIAM Intl. Conf. on Data Mining'.

Gallo, A., De Bie, T. & Cristianini, N. (2007), Mini: Mining informative non-redundant itemsets, *in* 'Proc. Europ. Symp. on Principles of Data Mining and Knowledge Discovery', pp. 438–445.

Garriga, G. (2005), Summarizing sequential data with closed partial orders, *in* 'Proc. 5th SIAM Intl. Conf. on Data Mining', SIAM, pp. 380–391.

Garriga, G., Kralj, P. & Lavrac, N. (2006), Closed sets for labeled data, *in* 'Proc. European Conf. on Principles and Practice of Knowledge Discovery in Databases', pp. 163–174.

Geerts, F., Goethals, B. & Mielikäinen, T. (2004), Tiling databases, *in* 'Proc. Discovery Science', pp. 278–289.

Goethals, B. & Zaki, M. (2003), FIMI '03, frequent itemset mining implementations, *in* 'Proc. ICDM 2003 Workshop on Frequent Itemset Mining Implementations'.

Grice, H. (1989), *Studies in the Way of Words*, Harvard University Press.

Gupta, R., Fang, G., Field, B., Steinbach, M. & Kumar, V. (2008), Quantitative evaluation of approximate frequent pattern mining algorithms, *in* 'Proc. 14th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining', ACM, pp. 301–309.

Han, J. & Pei, J. (2001), Pattern growth methods for sequential pattern mining: Principles and extensions, *in* 'Workshop on Temporal Data Mining, 7th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining', ACM Press.

Han, J., Pei, J. & Yin, Y. (2000), Mining frequent patterns without candidate generation, *in* 'Proc. ACM SIGMOD Intl. Conf. on Management of Data', ACM Press, pp. 1–12.

Hipp, J., Güntzer, U. & Nakhaeizadeh, G. (2000), 'Algorithms for association rule mining - a general survey and comparison.', *SIGKDD Explorations* **2**(1), 58–64.

Keogh, E., Lonardi, S., Ratanamahatana, C., Wei, L., Lee, S. & Handley, J. (2007), 'Compression-based data mining of sequential data', *Data Mining Knowledge Discovery* **14**(1), 99–129.

Kohavi, R., Brodley, C., Frasca, B., Mason, L. & Zheng, Z. (2000), 'KDD-Cup 2000 organizers' report: Peeling the onion', *SIGKDD Explorations* **2**(2), 86–98. http://www.ecn.purdue.edu/KDDCUP.

Kryszkiewicz, M. (2001), Concise representation of frequent patterns based on disjunction-free generators, *in* 'Proc. 1st IEEE Intl. Conf. on Data Mining', IEEE Press, pp. 305–312.

Kuramochi, M. & Karypis, G. (2001), Frequent subgraph discovery, *in* 'Proc. IEEE Intl. Conf. on Data Mining', pp. 313–320.

Li, J., Li, H., Wong, L., Pei, J. & Dong, G. (2006), Minimum description length principle: Generators are preferable to closed patterns, *in* 'Proc. AAAI', pp. 409–414.

Li, W., Han, J. & Pei, J. (2001), CMAR: Accurate and efficient classification based on multiple class-association rules, *in* 'Proc. IEEE Intl. Conf. on Data Mining', pp. 369–376.

Liu, B., Hsu, W. & Ma, Y. (1998), Integrating classification and association rule mining, *in* 'Proc. Intl. Conf. on Knowledge Discovery and Data Mining', pp. 80–86.

Liu, G., Li, J. & Wong, L. (2007), 'A new concise representation of frequent itemsets using generators and a positive border', *Knowledge and Information Systems* .

Liu, J., Paulsen, S., Wang, W., Nobel, A. & Prins, J. (2005), Mining approximate frequent itemsets from noisy data, *in* 'Proc. 5th Intl. Conf. Data Mining', IEEE, pp. 721–724.

Lucchese, C., Orlando, S. & Perego, R. (2006*a*), 'Fast and memory efficient mining of frequent closed itemsets', *IEEE Trans. on Knowledge and Data Engineering* **18**(1), 21–36.

Lucchese, C., Orlando, S. & Perego, R. (2006*b*), Mining frequent closed itemsets out of core, *in* 'Proc. of the 6th SIAM International Conf. on Data Mining (SDM'06)'.

Lucchese, C., Orlando, S. & Perego, R. (2007), Parallel mining of frequent closed patterns: Harnessing modern computer architectures, *in* 'Proc. IEEE Intl. Conf. on Data Mining'.

Malik, H. & Kender, J. (2006), High quality, efficient hierarchical document clustering using closed interesting itemsets, *in* 'Proc IEEE Intl. Conf. on Data Mining', pp. 991–996.

Mielikäinen, T. (2005), Summarization Techniques for Pattern Collections in Data Mining, PhD thesis, University of Helsinki, Finnland.

Mörchen, F. (2006), Algorithms for time series knowledge mining, *in* 'Proc. 12th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining', ACM Press, pp. 668–673.

Mörchen, F. & Ultsch, A. (2007), 'Efficient mining of understandable patterns from multivariate interval time series', *Data Mining and Knowledge Discovery* **15**(2), 181–215.

Muhonen, J. & Toivonen, H. (2006), Closed non-derivable itemsets., *in* 'Proc. Europ. Symp. on Principles of Data Mining and Knowledge Discovery', pp. 601–608.

Ng, R., Lakshmanan, L. V., Han, J. & Pang, A. (1998), Exploratory mining and pruning optimizations of constrained associations rules, *in* 'Proc. ACM SIGMOD Conf. on Management of Data', ACM, pp. 13–24.

Nijssen, S. & Fromont, E. (2007), Mining optimal decision trees from itemset lattices, *in* 'Proc. Intl. Conf. on Knowledge Discovery and Data Mining', ACM, pp. 530–539.

Pasquier, N., Bastide, Y., Taouil, R. & Lakhal, L. (1999), Discovering frequent closed itemsets for association rules, *in* 'Proc. 7th Intl. Conf. on Database Theory', Springer, pp. 398–416.

Pei, J., Dong, G., Zou, W. & Han, J. (2002), On computing condensed frequent pattern bases, *in* 'Proc. 2nd IEEE Intl. Conf. on Data Mining', IEEE Press, pp. 378–385.

Pei, J., Han, J. & Lakshmanan, L. V. S. (2001), Mining frequent itemsets with convertible constraints, *in* 'Proc. IEEE Intl. Conf. on Data Engineering', IEEE, pp. 433–442.

Pei, J., Tung, A. K. & Han, J. (2001), Fault-tolerant frequent pattern mining: Problems and challenges, *in* 'Workshop on Research Issues in Data Mining and Knowledge Discovery, 20th ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems'.

Pei, J., Wang, H., Liu, J., Wang, K., Wang, J. & Yu, P. (2006), 'Discovering frequent closed partial orders from strings', *IEEE Trans. on Knowledge and Data Engineering* **18**(11), 1467–1481.

Pudi, V. & Haritsa, J. (2003), Generalized closed itemsets for association rule mining, *in* 'Proc. 19th Intl. Conf. on Data Engineering', IEEE Press, pp. 714–716.

Seppänen, J. & Mannila, H. (2004), Dense itemsets, *in* 'Proc. 10th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining', ACM Press, pp. 683–688.

Siebes, A. (2006), Item sets that compress, *in* 'Proc. SIAM Conf. on Data Mining', pp. 393–404.

Song, G., Yang, D., Cui, B., Zheng, B., Liu, Y. & Xie, K. (2007), CLAIM: An efficient method for relaxed frequent closed itemsets mining over stream data, *in* 'Proc. 12th Intl. Conf. on Database Systems for Advanced Applications', Springer, pp. 664–675.

Srikant, R., Vu, Q. & Agrawal, R. (1997), Mining association rules with item constraints, *in* 'Proc. Intl. Conf. on Knowledge Discovery and Data Mining', ACM, pp. 67–73.

Sripada, S. G., Reiter, E. & Hunter, J. (2003), Generating English summaries of time series data using the Gricean maxims, *in* 'Proc. 9th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining', ACM Press, pp. 187–196.

Tatti, N. (2007), Maximum entropy based significance of itemsets, *in* 'Proc. IEEE Intl. Conf. on Data Mining', pp. 312–321.

Tatti, N. (2008), 'Maximum entropy based significance of itemsets', *Knowledge and Information Systems* **17**(1), 57–77.

Uno, T. & Arimura, H. (2007), An efficient polynomial delay algorithm for pseudo frequent itemset mining, *in* 'Proc. 10th Intl. Conf. Discovery Science', Springer, pp. 219–230.

Van Leeuwen, M. & Siebes, A. (2008), StreamKrimp: Detecting change in data streams, *in* 'Proc. Europ. Conf. on Machine Learning and Principles and Practices of Knowledge Discovery in Data', pp. 765–774.

van Leeuwen, M., Vreeken, J. & Siebes, A. (2006), Compression picks item sets that matter, *in* 'Proc. European Conf. on Principles and Practice of Knowledge Discovery in Databases', pp. 585–592.

Vreeken, J. & Siebes, A. (2008), Filling in the blanks - Krimp minimisation for missing data, *in* 'Proc. 8th IEEE Intl. Conf. on Data Mining', pp. 1067–1072.

Wang, J. & Karypis, G. (2006), 'On mining instance-centric classification rules', *IEEE Trans. on Knowledge and Data Engineering* **18**(11), 1497–1511.

Wang, K., Xu, C. & Liu, B. (1999), Clustering transactions using large items, *in* 'Conf. on Information and Knowledge Management', pp. 483–490.

Webb, G. I. (2007), 'Discovering significant patterns', *Mach. Learn.* **68**(1), 1–33.

Xin, D., Han, J., Yan, X. & Cheng, H. (2005), Mining compressed frequent-pattern sets, *in* 'Proc. 31st Intl. Conf. on Very Large Data Bases', pp. 709–720.

Yahia, S. B., Hamrouni, T. & Mephu Nguifo, E. (2006), 'Frequent closed itemset based algorithms: A thorough structural and analytical survey', *ACM SIGKDD Explorations* **8**(1), 93–104.

Yan, X., Cheng, H., Han, J. & Xin, D. (2005), Summarizing itemset patterns: a profile-based approach, *in* 'Proc. 11th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining', ACM Press, pp. 314–323.

Yang, C., Fayyad, U. & Bradley, P. (2001), Efficient discovery of error-tolerant frequent itemsets in high dimensions, *in* 'Proc. 7th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining', ACM Press, pp. 194–203.

Yin, X. & Han, J. (2003), CPAR: Classification based on predictive association rules, *in* 'Proc. SIAM Intl. Conf. on Data Mining'.

Zaki, M. (2004), 'Mining non-redundant association rules', *Data Mining and Knowledge Discovery* **9**(3), 223–248.

Zaki, M. & Hsiao, C.-J. (2002), CHARM: An efficient algorithm for closed itemset mining, *in* 'Proc. 2nd SIAM Intl. Conf. on Data Mining', SIAM, pp. 457–473.

Zhao, Y. & Karypis, G. (2002), Evaluation of hierarchical clustering algorithms for document datasets, *in* 'Proc. 11th Conf. of Information and Knowledge Management', pp. 515–524.

Author Biographies

Fabian Moerchen graduated with a Ph.D. in Feb 2006 from the Philipps University of Marburg, Germany after just over 3 years with summa cum laude. In his thesis he proposed a radically different approach to temporal interval patterns that uses itemset and sequential pattern mining paradigms. Since 2006 he has been working at Siemens Corporate Research, a division of Siemens Corporation, leading data mining projects with applications in predictive maintenance, text mining, healthcare, and sustainable energy. He has continued the study of temporal data mining in the context of industrial and scientific problems and has served the community as a reviewer, organizer of workshops, and presenter of tutorials.

Michael Thies received his Master degree in Computer Science from Philipps University of Marburg, Germany. He is currently working as a self-employed software developer and data analyst. His research interests include data mining in general and temporal pattern mining in particular. He further participated in the MusicMiner project that deployed signal processing and data mining methods to groups recorded musical pieces by perceived similarity.

Alfred Ultsch received his Ph.D Degree in Computer Science from ETH Zurich, Switzerland. He holds Master degrees in Computer Science from TU Munich, Germany and Purdue University West Lafayette Indiana, USA. He is currently a Professor in the Philipps University of Marburg, Germany in the field of Databionics. Prof. Ultsch has published in the areas of bio-inspired computing, databionics, knowledge discovery recognition, bioinformatics and financial analysis.

Correspondence and offprint requests to: Fabian Moerchen, Siemens Corporate Research, 755 College Road East, Princeton, NJ, 08540, USA. Email: fabian.moerchen@siemens.com