Deducing Bounds on the Frequency of Itemsets

Toon Calders*

University of Antwerp
Universiteitsplein 1, B-2610 Wilrijk, Belgium
calders@uia.ua.ac.be

Abstract. Mining *Frequent Itemsets* is the core operation of many data mining algorithms. This operation however, is very data intensive and sometimes produces a prohibitively large output. In this paper we give a complete set of rules for deducing tight bounds on the frequency of an itemset if the frequencies of all its subsets are known. These rules allow for reducing data access and providing a more compact output. Based on the derived bounds $[l, u]$ of a candidate itemset C, we can decide not to access the database to count its frequency if l is larger than the support threshold (C will certainly be frequent), or if u is smaller than the threshold (C will certainly fail the frequency test). In this way, the number of runs through the database and the number of sets to count can be reduced significantly. We can also use the rules to reduce the size of an adequate representation of the collection of frequent sets; all itemset frequencies that can be deduced do not need to be stored explicitly. To assess the usability in practice, we implemented the deduction rules and we present experiments on a real-life dataset.

1 Introduction

Mining frequent itemsets forms a core operation in many data mining problems. Since their introduction [1], many algorithms have been proposed to find frequent itemsets, especially in the context of association rule mining [1, 2, 11].

The *frequent itemset problem* is stated as follows. Assume we have a finite set of items I. A *transaction* is a subset of I, together with a unique identifier. A *transaction database* is a finite set of transactions. A subset of I is called an itemset. We say that an itemset K is s-frequent in a transaction database T if the fraction of the transactions in T that contain all items of K is at least s. The frequent itemset problem is, given a support threshold s and a transaction database T, find all s-frequent itemsets.

All algorithms for mining frequent itemsets rely heavily on the following *monotonicity principle* [15]:

> Let $K \subseteq L$ be two itemsets. In every transaction database T, the frequency of L will be at most as high as the frequency of K.

* Research Assistant of the Fund for Scientific Research - Flanders (FWO-Vlaanderen).

This simple rule of *deduction* has successfully been used many times. Because of the success of this simple rule, much attention went into efficient counting schemes for the generated candidates. The best example is the well-known Apriori-algorithm. Apriori goes through the itemset-lattice level by level; in the i-th loop, itemsets of cardinality i are counted. Because of the monotonicity principle, all itemsets in loop i that have at least one subset that failed the support-test can be *pruned*; we know a priori that they will be infrequent. Thus, based on the frequency of one subset that is below the frequency threshold, we can *deduce*, using the monotonicity rule, that also the frequency of the candidate will be below the threshold. In this paper we present some additional deduction rules that calculate lower and upper bounds on the frequency of a candidate, and as such continues work done in [7]. Based on the frequencies of all its subsets, the rules compute bounds $[l, u]$ on the frequency of an itemset. The rules calculate the best possible bounds; i.e., if bounds $[l, u]$ are calculated, then both l and u are possible as frequency of the itemset, and thus, the interval cannot be made more tight.

Based on these bounds we can avoid counting too many candidates, or decide to count more candidates in one run over the database. Also, as the experiments will show, the intervals decrease in width very fast. Therefore, it is possible to generate a *summary* of the frequent itemsets that only contains the non-derivable sets. The collection of non-derivable sets forms a concise representation [14] of the frequent sets. Other concise representations in the literature include *free sets* [4], *closed sets* [16, 3, 17], and *disjunction-free sets* [5].

There are also some analogs with interactive association rule mining. In [8, 9], the authors develop a framework that allows to reuse results of previous data mining queries. For example, parts of the answer to the query asking for the supports of all itemsets containing a certain item A can be reused to answer the query that asks for the frequency of all itemsets that do not contain B. The deduction rules introduced here can be used orthogonally to this approach. Based on previous results, bounds on the frequency of new, not yet counted itemsets can be calculated.

Another application of deduction rules is developed in [10]. Based on the observation that highly frequent items tend to blow up the output of a data mining query by an exponential factor, the authors develop a technique to leave out these highly frequent items, and to reintroduce them after the mining phase by using a deduction rule, the *multiplicative* rule. A special form of this multiplicative rule will also appear in our framework.

In Section 2 we give an example showing that the monotonicity rule is not complete for deduction of frequencies. This example also gives a sketch of the general approach. In Section 3 we will give some important definitions. In Section 4, the complete set of deduction rules is given. In Section 5 we present the results of experiments with the deduction rules and Section 6 concludes the paper.

2 Motivating Example

Apriori does not prune perfectly. Consider the following example[1]:

$$\begin{array}{l} freq(A,R) = freq(B,R) = freq(C,R) = \frac{2}{3} \\ freq(AB,R) = freq(AC,R) = freq(BC,R) = \frac{1}{3} \end{array} \tag{1}$$

Suppose we are running the Apriori-algorithm with the minimal support threshold set to $\frac{1}{3}$. The algorithm will start with counting the frequencies of the singleton-itemsets $L_1 = \{\{A\},\{B\},\{C\}\}$. Since they are all frequent, Apriori will consider in its second loop the candidates $L_2 = \{\{A,B\},\{A,C\},\{B,C\}\}$. Again all candidates are frequent, and thus, Apriori will count $L_3 = \{\{A,B,C\}\}$ in its third loop. However, the following simple observation teaches us that from the frequencies counted for the singletons and the pairs, we can derive that $\{A,B,C\}$ cannot be frequent.

We can encode this situation as a linear programming instance. This representation is also used in [7, 6]. Let a denote the fraction of transactions t in the database having $t.Items = \{A\}$, b is the fraction of transactions having $t.Items = \{B\}$, ..., ab is the fraction having $t.Items = \{A,B\}$, ..., and abc is the fraction of transactions t having $t.Items = \{A,B,C\}$. Variable z is the fraction of transactions having the empty set as set of items. For every relation satisfying the frequencies in (1), the following holds:

$$\begin{cases} a + ab + ac + abc = \frac{2}{3} & (A) \\ b + ab + bc + abc = \frac{2}{3} & (B) \\ c + ac + bc + abc = \frac{2}{3} & (C) \\ ac + abc = \frac{1}{3} & (AC) \\ ab + abc = \frac{1}{3} & (AB) \\ bc + abc = \frac{1}{3} & (BC) \\ a + b + c + ab + ac + bc + abc + z = 1 & \\ a, b, c, ac, bc, ab, z, abc \geq 0 & \end{cases} \tag{2}$$

From this system we derive:

$$\begin{cases} a + ab = \frac{1}{3} \; (A - AC) \\ a + ac = \frac{1}{3} \; (A - AB) \\ b + ab = \frac{1}{3} \; (B - BC) \\ b + ac = \frac{1}{3} \; (B - AB) \\ c + ac = \frac{1}{3} \; (C - BC) \\ c + bc = \frac{1}{3} \; (C - AC) \end{cases} \tag{3}$$

The solution of system (3) is $a = b = c = k, ab = ac = bc = \frac{1}{3} - k$ with k a parameter (6 vars, rank 5, thus 1 parameter). Because $a + b + c + ab + ac + bc + abc + z = 1$, we derive $z + abc \leq 0$. Since $z \geq 0$ and $abc \geq 0$, it holds that $abc = 0$. Therefore, $freq(ABC,R) = 0$, and we know *a priori* that ABC cannot be frequent. Nevertheless, Apriori does not prune ABC. This reasoning shows that pruning still can be improved.

[1] $freq(K,R)$ denotes the frequency of itemset K in the database R. For precise definitions we refer to Section 3

3 Definitions

In this section we define the *frequent itemset problem* and the implication of frequencies. We also introduce the notion of a K-fraction in a transaction database. This notion will allow us to make similar derivations as in the last example, but in a more general setting.

3.1 Frequent Itemset Problem

Let I, the *set of items*, be a finite set.

- A *transaction t over I* is defined as a subset of I, denoted $t.Items$, together with an *identifier*, denoted $t.ID$.
- A finite set of transactions over I is called a *transaction database over I*. We also require that no two different transactions in T have the same ID. Notice that due to the identifiers there can be two different transactions in the database with the same set of items.
- A subset of I is called an *itemset*. We will often denote an itemset K by the list of its elements; i.e., $\{A, B, C\}$ is denoted by ABC.
- The *frequency* of an itemset K over I in a transaction database T over I, denoted $\mathit{freq}(K, T)$, is defined as
$$\mathit{freq}(K, T) := \frac{|\{t \in T \mid K \subseteq t.Items\}|}{|T|} \ .$$
- A *Frequent Itemset Problem* is a triple (I, T, s), with I a finite set of items, T a transaction database over I, and $0 \leq s \leq 1$ a rational number. The solution of the problem (I, T, s) is
$$FreqSet_I(T, s) := \{K \subseteq I \mid \mathit{freq}(K, T) \geq s\} \ .$$

Example 1. Consider the following transaction database T over $\{A, B, C, D\}$.

$T =$

TID	items
1	A, B, C
2	A, B, D
3	A, D
4	A, D

The frequency of the itemset $\{A, D\}$ in T is $\frac{3}{4}$ because 3 transactions out of 4 contain both A and D; only the transaction with transaction identifier (TID) 1 does not contain D.

3.2 Fraction

In the rest of the paper the following definition will be very important.

- Let T be a transaction database and L be a subset of I. We define the *L-fraction of* T, denoted $t_L(T)$ as
$$\frac{|\{t \in T \mid t.Items = L\}|}{T} \ .$$
Hence, the L-fraction of T is the fraction of transactions having L as set of items. If T is clear from the context, we will write t_L.

These definitions allow us to restate the frequency of an itemset K in terms of the different L-fractions in the transaction database.

Lemma 1. *Let T be a transaction database and K be an itemset over I. Then the following holds.*

$$\mathit{freq}(K,T) = \sum_{K \subseteq L \subseteq I} t_L \enspace .$$

Thus, the frequencies of the itemsets only depend on the different L-fractions t_L, with $L \subseteq I$.

Proof.

$$\begin{aligned}\mathit{freq}(K,T) &= \frac{|\{t \in T \mid K \subseteq t.\mathit{Items}\}|}{|T|} = \frac{\left|\bigcup_{K \subseteq L}\{t \in T \mid t.\mathit{Items} = L\}\right|}{|T|} \\ &= \frac{\sum_{K \subseteq L \subseteq I} |\{t \in T \mid t.\mathit{Items} = L\}|}{|T|} = \sum_{K \subseteq L \subseteq I} \frac{|\{t \in T \mid t.\mathit{Items} = L\}|}{|T|} \\ &= \sum_{K \subseteq L \subseteq I} t_L\end{aligned}$$

Example 2. In the transaction database T in Example 1, the AB-fraction t_{AB} is 0, $t_{ABC} = \frac{1}{4}$, $t_{ABD} = \frac{1}{4}$, $t_{ABCD} = 0$. $\mathit{freq}(AB,T) = t_{AB} + t_{ABC} + t_{ABD} + t_{ABCD} = \frac{2}{4}$.

In the remainder of the paper, the following lemma will be important since it will allow us to restate frequency problems as linear inequality problems. The existence of a transaction database fulfilling certain conditions will be equivalent to the existence of a solution of a linear system of inequalities.

Lemma 2. *Let I be a set of items, and for each $L \subseteq I$, let v_L be a rational number. There exists a transaction database T with $\forall L \subseteq I : t_L(T) = v_L$ iff $\{x_L = v_L \mid L \subseteq I\}$ is a solution of the following system of inequalities:*

$$\begin{cases} x_L \geq 0 & \forall L \subseteq I \\ \sum_{L \subseteq I} x_L = 1 \end{cases}$$

Proof. Only if $v_L \geq 0$ cannot be violated since $t_L(T)$ must always be positive. By definition, $\sum_{L \subseteq I} t_L(T) = \sum_{L \subseteq I} \frac{|\{t \in T \mid t.\mathit{Items} = L\}|}{T} = \frac{|T|}{|T|} = 1$.

If Suppose $\{x_L = v_L \mid \forall L \subseteq I\}$ is a solution of the system. Let n be the least common multiple of the nominators of the different v_L's. Construct the transaction database T as follows: for each L add $n.v_L$ transactions t with $t.\mathit{Items} = L$. It is easy to see this construction provides us with a transaction database with the desired fractions.

Corollary 1. *Let I be a set of items, $L_1, \ldots, L_n \subseteq I$, and $f_1, \ldots, f_n$ rational numbers. There exists a transaction database T over I such that for all $1 \leq i \leq n$ it holds that $\mathit{freq}(L_i, T) = f_i$ iff the following system of inequalities over variables $\{t_L \mid L \subseteq I\}$ has a solution:*

$$\begin{cases} t_L \geq 0 \qquad \forall L \subseteq I \\ \sum_{L \subseteq I} t_L = 1 \\ \sum_{I \supseteq t_{L_i}} = f_i \qquad 1 \leq i \leq n \end{cases}$$

Example 3. There exists a transaction database T with $\mathit{freq}(A,T) = \frac{2}{3}$,$\mathit{freq}(B,T) = \frac{2}{3}$, and $\mathit{freq}(AB,T) = 0$ iff the following system of inequalities has a solution:

$$\begin{cases} t_{\{\}} \geq 0, t_A \geq 0, t_B \geq 0, t_{AB} \geq 0 \\ t_{\{\}} + t_A + t_B + t_{AB} = 1 \\ t_A + t_{AB} = \frac{2}{3} \\ t_B + t_{AB} = \frac{2}{3} \\ t_{AB} = 0 \end{cases}$$

From $t_A + t_{AB} = \frac{2}{3}$, $t_B + t_{AB} = \frac{2}{3}$, and $t_{AB} = 0$, we derive that $t_A = t_B = \frac{2}{3}$. Therefore, because $t_{\{\}} \geq 0$, it follows that $t_{\{\}} + t_A + t_B + t_{AB} > 1$, and the system has no solution. Hence, we conclude that there does not exist a transaction database with $\mathit{freq}(A,T) = \frac{2}{3}$, $\mathit{freq}(B,T) = \frac{2}{3}$, and $\mathit{freq}(AB,T) = 0$.

3.3 Implication

We now define implication of frequencies. Let I be a set of items.

- A *Frequency Expression* over I is an expression $freq(K) = f_K$, with K an itemset over I and $0 \leq f_K \leq 1$ a rational number.
- A transaction database T over I is said to *satisfy* a frequency expression $freq(K) = f_K$ iff $\mathit{freq}(K,T) = f_K$.
- A transaction database is said to *satisfy a set of frequency expressions S* iff it satisfies every expression in S.
- Let K be an itemset over I, and $0 \leq l \leq u \leq 1$. A set of frequency expressions S *implies bounds $[l, u]$ for K*, denoted $S \models freq(K) \in [l, u]$, iff in every transaction database satisfying S, it holds that $l \leq \mathit{freq}(K,T) \leq u$.
- Let K be an itemset over I, and $0 \leq l \leq u \leq 1$. A set of frequency expressions S implies *tight* bounds $[l, u]$ for K, denoted $S \models_{\text{tight}} freq(K) \in [l, u]$, iff $S \models freq(K) \in [l, u]$ and for all rational numbers l', u' such that $S \models freq(K) \in [l', u']$ it holds that $l' \leq l$ and $u' \geq u$.

Example 4. Let

$$S := \left\{ \begin{matrix} freq(A) = \frac{2}{3}, & freq(B) = \frac{2}{3}, & freq(C) = \frac{2}{3} \\ freq(AB, R) = \frac{1}{3}, & freq(AC, R) = \frac{2}{3}, & freq(BC, R) = \frac{1}{3}. \end{matrix} \right\} .$$

From the monotonicity rule we know that

$$S \models freq(ABC) \in [0, \frac{1}{3}] \ .$$

From Section 2 we know that

$$S \models_{\text{tight}} freq(ABC) \in [0,0] \enspace .$$

In the remainder of the paper, when we study the bounds implied for an itemset K, we will always start with a set of frequency expressions that contains exactly one expression for every subset of K, and no other expressions. For this specific case we will give sound and complete rules for deduction bounds of an itemset. This special case is interesting because in many algorithms, like in Apriori, we have the frequencies of all subsets at our disposal. If there are more expressions, or some expressions are missing, the rules will no longer compute a complete answer.

4 Deduction Rules

In this section we describe sound and complete rules for deducing tight bounds on the frequency of a set K if the frequencies of all its subsets are given. Because we do not consider itemsets that are not subset of K, we can assume that all items in the database are elements of K. This assumption allows us to reduce the complexity of the problem. Since "projecting away" the other items in a transaction database does not change frequencies of subsets of K, we can assume without loss of generality that $K = I$. This property is expressed by the next lemma, but first we define the projection of a transaction database on an itemset.

Definition 1. *Let I be a set of items, $K \subseteq I$.*

- *The* projection of a transaction t over I on K, *denoted $\pi_K t$, is a transaction t' with $t'.ID = t.ID$, and $t'.Items = t.Items \cap K$. Hence, the projection keeps only the items of the transaction that are in K.*
- *The* projection *of a transaction database T over I on K, denoted $\pi_K T$, is defined as $\pi_K T := \{\pi_K t \mid t \in T\}$.*

Lemma 3. *Let I be a set of items, and $L, K \subseteq I$ be itemsets such that $L \subseteq K$. For every transaction database T over I it holds that*

$$freq(L,T) = freq(L, \pi_K T).$$

Proof. Straightforward; if $L \subseteq K$, then $L \subseteq t.Items$ implies $L \subseteq (t.Items \cap K) = (\pi_K t).Items$.

Corollary 2. *Given a set of items I, $K \subseteq I$, and a rational number f_L for each $L \subseteq K$. There exists a transaction database T satisfying $\forall L \subseteq K : freq(L,T) = f_L$ iff the following system of inequalities has a solution.*

$$\begin{cases} t_L \geq 0 \qquad \forall L \subseteq K \\ \sum_{L \subseteq K} t_L = 1 \\ \sum_{L \subseteq M \subseteq K} = f_L \qquad \forall L \subseteq K \end{cases}$$

Let K be a set of items, T a transaction database over K. We assume that all frequencies of the strict subsets of K are known, let f_L denote *freq*(L,T). From Lemma 1, we derive the following equalities.

$$\begin{cases} f_{\{\}} & = t_{\{\}} t_A + t_B + t_C + t_D + + t_{AB} + t_{AC} + \ldots + t_K \\ f_A & = t_A + t_{AB} + t_{AC} + \ldots + t_{ABC} + t_{ABD} + \ldots + t_K \\ f_B & = t_B + t_{AB} + t_{BC} + \ldots + t_{ABC} + t_{ABD} + \ldots + t_K \\ \ldots & \ldots\ldots \\ f_{AB} & = t_{AB} + t_{ABC} + t_{ABD} + \ldots + t_K \\ \ldots & \ldots\ldots \\ f_{K-A} & = t_{(K-A)} + t_K \\ f_K & = t_K \end{cases} \tag{4}$$

This system of equalities contains $2^{|K|}$ equations and $2^{|K|}+1$ variables (t_L for all $L \subseteq K$, and f_K). Thus, the solution of this system will contain one parameter. Let f_K be this parameter. This choice gives the following solution.

$$\begin{cases} t_{\{\}} & = f_{\{\}} - f_A - f_B - f_C - f_D + f_{AB} + \ldots - f_{ABC} - \ldots + (-1)^k f_K \\ t_A & = f_A - f_{AB} - f_{AC} - \ldots + f_{ABC} + \ldots - f_{ABCD} - \ldots + (-1)^{k-1} f_K \\ \ldots & \ldots\ldots \\ t_{AB} & = f_{AB} - f_{ABC} - f_{ABD} - \ldots + f_{ABCD} + f_{ABCE} + \ldots + (-1)^{k-2} f_K \\ \ldots & \ldots\ldots \\ t_{K-A} & = f_{K-A} - f_K \\ t_K & = f_K \end{cases} \tag{5}$$

Thus, for every value of f_K we get a solution of the system in (4). Some of these solutions will present fractions in a transaction database, and some will not (e.g., if one of the t_K's is negative.) The lower bound on the frequency of K based on the frequencies of the subsets will be the smallest value of f_K such that the associated solution of the system in (4) represents fractions in a transaction database. Analogously, the upper bound will be the greatest f_K such that the solution represents fractions in a transaction database. From Lemma 2 we know that the solution represents a valid transaction database if and only if the following conditions are satisfied by the solution.

$$\begin{cases} t_A & \geq\ 0 \\ t_B & \geq\ 0 \\ \ldots & \ldots\ldots \\ t_K & \geq\ 0 \\ t_{\{\}} + t_A + t_B + \ldots + t_K & =\ 1 \end{cases} \tag{6}$$

Therefore, by applying these conditions to the parameterized solution, we get the following conditions and theorem for determining tight bounds on *freq*(K,T). Notice the similarity with the inclusion-exclusion principle [13, p. 181]. Notice also that the condition $t_{\{\}} + t_A + t_B + \ldots + t_K = 1$ is already fulfilled by the solutions of the system, since $f_{\{\}} = 1$.

$$\begin{cases} (-1)^k f_K & \geq 1 - \Big[(-1)^k(f_{K-A} + f_{K-B} + \ldots) + (-1)^{k-1}(f_{K-AB} + \ldots) \\ & \quad + \ldots + f_A + f_B + \ldots\Big] \\ (-1)^{k-1} f_K & \geq -f_A + f_{AB} + f_{AC} + \ldots - f_{ABC} - \ldots + f_{ABCD} \cdots \\ \cdots & \cdots\cdots \\ (-1)^{k-2} f_K & \geq -f_{AB} + f_{ABC} + f_{ABD} + \ldots - f_{ABCD} - f_{ABCE} - \ldots \\ & \quad + f_{ABCDEF} + f_{ABCDEG} + \cdots \\ \cdots & \cdots\cdots \\ -f_K & \geq f_{K-A} \\ f_K & \geq 0 \end{cases} \tag{7}$$

Theorem 1. *Let I be a set of items, $K \subseteq I$. For each $L \subset K$ a rational number f_l has been given. Let $Lower(K)$ be the right-hand sides of the equations in (7) with left-hand side f_K, and $Upper(K)$ be the negation of the right-hand sides of the equations in (7) with left-hand side $-f_K$. Then*

$$\{freq(L) = f_L \mid L \subset K\} \models_{\text{tight}} f_K \in [\max(Lower(K)), \min(Upper(K))] \enspace .$$

$$\begin{cases} f_{ABCD} \leq f_A - f_{AB} - f_{AC} - f_{AD} + f_{ABC} + f_{ABD} + f_{ACD} \\ f_{ABCD} \leq f_B - f_{AB} - f_{BC} - f_{BD} + f_{ABC} + f_{ABD} + f_{BCD} \\ f_{ABCD} \leq f_C - f_{AC} - f_{BC} - f_{CD} + f_{ABC} + f_{ACD} + f_{BCD} \\ f_{ABCD} \leq f_D - f_{AD} - f_{BD} - f_{CD} + f_{ABD} + f_{ACD} + f_{BCD} \\ \\ f_{ABCD} \geq f_{ABC} + f_{ABD} - f_{AB} \\ f_{ABCD} \geq f_{ABC} + f_{ACD} - f_{AC} \\ f_{ABCD} \geq f_{ABD} + f_{ACD} - f_{AD} \\ f_{ABCD} \geq f_{ABC} + f_{BCD} - f_{BC} \\ f_{ABCD} \geq f_{ABD} + f_{BCD} - f_{BD} \\ f_{ABCD} \geq f_{ACD} + f_{BCD} - f_{CD} \\ \\ f_{ABCD} \leq f_{ABC} \\ f_{ABCD} \leq f_{ABD} \\ f_{ABCD} \leq f_{ACD} \\ f_{ABCD} \leq f_{BCD} \\ \\ f_{ABCD} \geq 0 \\ \\ f_{ABCD} \geq f_{ABC} + f_{ABD} + f_{ACD} + f_{BCD} - f_{AB} - f_{AC} - f_{AD} - f_{BC} - f_{BD} - f_{CD} \\ \qquad\quad + f_A + f_B + f_C + f_D - 1 \end{cases} \tag{8}$$

Fig. 1. Tight bounds on *freq*$(ABCD, T)$

Example 5. Consider the following transaction database.

$T =$

TID	items
1	A, B
2	A, C, D
3	A, B, D
4	C, D
5	B, C, D
6	A, D
7	B, D
8	B, C, D
9	B, C, D
10	A, B, C, D

$$\begin{array}{lll} f_A = \frac{1}{2}, & f_B = \frac{7}{10}, & f_C = \frac{3}{5}, \\ f_D = \frac{9}{10}, & f_{AB} = \frac{3}{10}, & f_{AC} = \frac{1}{5}, \\ f_{AD} = \frac{2}{5}, & f_{BC} = \frac{2}{5}, & f_{BD} = \frac{3}{5}, \\ f_{CD} = \frac{3}{5}, & f_{ABC} = \frac{1}{10}, & f_{ABD} = \frac{1}{5}, \\ f_{ACD} = \frac{1}{5}, & f_{BCD} = \frac{2}{5}. & \end{array}$$

Figure 1 gives the rules to determine tight bounds on the frequency of $ABCD$. Based on these deduction rules we derive the following bounds on *freq*$(ABCD, T)$ *without counting in the database*.

Lower bound: *freq*$(ABCD, T) \geq \frac{1}{10}$ (Rule $f_{ABCD} \geq f_{ABC} + f_{ACD} - f_{AC}$)
Upper bound: *freq*$(ABCD, T) \leq \frac{1}{10}$ (Rule $f_{ABCD} \leq f_{ABC}$)

Therefore, we can conclude, without having to count, that the frequency of $ABCD$ in T is exactly $\frac{1}{10}$. In the experiments we will see that this exactness is not very unusual; even in real-life data, and for small itemsets, we will be able to derive very narrow intervals.

5 Experiments

5.1 Dataset

The dataset we used to perform the experiments is derived from the census-dataset as available in the UCI KDD-repository [12]. This dataset is *in se* a relational table, with 68 numerical attributes. We transformed this dataset into a transaction database in the following way: every (attribute,value)-pair was considered as a different item. Notice that therefore a value a in attribute A denotes another item as the same value a but in another attribute B. Using this convention, every tuple was transformed into a transaction with 68 items. In order to speed-up the experiments, we only used a random sample of 10000 transactions. The dataset contains 396 different items.

5.2 Results

In this section we describe the tests we performed and the results.

Pruning In this test we want to see how much pruning can be performed by using the deduction rules. We mined the transaction database at different support levels, and we record in every pass of the Apriori-algorithm the following measures: (a) the number of candidate itemsets, (b) the number of frequent itemsets, (c) the number of itemsets for which the lower bound is above the support, and (d) the number of itemsets for which the upper bound is below the support.

It is important to remark that in these counts only the itemsets that are not pruned by the monotonicity rule are evaluated with the deduction rules. Thus, the numbers we give represent pruning *additional* to the monotonicity rule. From these tests it is clear that the amount of pruning done by the monotonicity rule can be improved drastically. For example, in all tests, from pass 4 we know almost perfectly, even before we counted the candidates, which candidates will turn out to be frequent.

Support = 90%, all 396 items

	$\lvert Can\rvert$	#Freq	$\#l \geq s$	$\#u < s$
1	396	20		
2	190	159	151	0
3	750	598	592	152
4	1512	1170	1170	342
5	1469	1186	1186	283
		...		

Support = 90%, 100 items

	$\lvert Can\rvert$	#Freq	$\#l \geq s$	$\#u < s$
1	100	20		
2	190	159	151	0
3	750	598	592	152
4	1512	1170	1170	342
5	1469	1186	1186	283
6	710	622	622	88
7	170	165	165	5
8	16	16	16	0
9	1	1	1	0

Support = 10%, all 396 items

	$\lvert Can\rvert$	#Freq	$\#l \geq s$	$\#u < s$
1	396	133		
2	8778	5444	3085	0
3	131258	121875	117089	2089
4	1853220	1809695	1802860	35491
		...		

Support = 10%, 20 items

	$\lvert Can\rvert$	#Freq	$\#l \geq s$	$\#u < s$
1	20	16		
2	120	101	72	0
3	355	348	347	2
4	759	754	752	5
5	1091	1050	1050	41
6	985	974	974	11
7	623	621	621	2
8	278	278	278	0
9	82	82	82	0
10	14	14	14	0
11	1	1	1	0

Interval width We test the mean width of the intervals we derive. If an interval is a point-interval; i.e. the lower bound equals the upper bound, there is no need to count the frequency, nor is there a need to store the set in a concise representation of the frequent itemsets. Therefore, we pay special attention to this type of itemsets. We mined for frequent itemsets at different support levels and we report for each loop of the Apriori-algorithm the following measures: (a) the mean interval width, (b) the number of candidate itemsets for which $l = u$, and (c) the number of candidate itemsets with interval width at most 0.1%, and 0.05%.

From the tests we see that the width decreases very fast. After pass 4, in all our tests, we know *exactly* the frequencies of all sets that follow. Of course, when we increase the number of transactions, the number of frequencies we know exactly will decrease, but the width of the intervals will remain the same.

Support = 90%, all 396 items

	$\lvert Can\rvert$	Width	$l = u$	0.05%	0.1%
1	396				
2	190	2.16%	0	0	19
3	750	0.029%	313	625	697
4	1512	$\approx 0\%$	1494	1512	1512
5	1469	0%	1469	1469	1469
			...		

Support = 10%, 20 items

	$\lvert Can\rvert$	Width	$l = u$	0.05%	0.1%
1	20				
2	120	7.5%	0	0	0
3	355	0.21%	71	170	201
4	759	$\approx 0\%$	590	746	756
5	1091	$\approx 0\%$	1087	1091	1091
6	985	0%	985	985	985
7	623	0%	623	623	623
8	278	0%	278	278	278
9	82	0%	82	82	82
10	14	0%	14	14	14
11	1	0%	1	1	1

Concise representations In this test we measure how large a concise representation of the set of frequent itemsets would be. Let $[l_K, u_K]$ be the bounds we can derive for an itemset K, based on the frequency of its subsets. As a concise representation of the set of frequent itemsets F we take the following set: $C := \{(K, \mathit{freq}(K,T)) \in (F \times [0,1]) \mid l_K \neq u_K\}$. The following table gives $|C|$ and the number of frequent sets for some tests. In the tests, the concise representation is much smaller than the actual set of frequent itemsets. This concise representation contains all information needed to find frequent itemsets.

Support	$\lvert I\rvert$	#Freq	$\lvert C\rvert$
90%	100	3937	634
10%	20	4239	569
1%	10	255	113

6 Conclusions and Further Work

We presented sound and complete rules for deducing bounds on the frequency of an itemset. These rules have many possible applications, such as improving the pruning in the Apriori-algorithm, making concise representations, and deducing the result of a data mining query based on previous query results. We implemented the rules and we evaluated them against a real-life dataset. Although the results of the experiments are still premature, they show that in at least the census-dataset, many useful deductions can be made. The results even show that in most cases the frequencies of the itemsets up to length 4 determine all other frequencies almost exactly.

Interesting further work includes testing on different datasets, comparing with other consise representations such as free sets and closed sets, and evaluating which rules tend to give the best bounds. An important question to answer here is: are all rules as important? We also need to find an effective way to evaluate the rules. At this moment, a brute force calculation is performed. This calculation takes exponential time in the size of the itemset to be tested.

References

1. R. Agrawal, T. Imilienski, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. ACM SIGMOD*, pages 207–216, 1993.
2. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. VLDB*, pages 487–499, 1994.
3. J.-F. Boulicaut and A. Bykowski. Frequent closures as a concise representation for binary data mining. In *Proc. PaKDD*, pages 62–73, 2000.
4. J.-F. Boulicaut, A. Bykowski, and C.Rigotti. Approximation of frequency queries by means of free-sets. In *Proc. PaKDD*, pages 75–85, 2000.
5. A. Bykowski, and C. Rigotti. A Condensed Representation to find Frequent Patterns. In *Proc. ACM PODS*, 2001.
6. A. Bykowski, J.K. Seppänen, and J. Hollmén. Model-independent bounding of the supports of Boolean formulae in binary data. CIS Research Report, Helsinki University of Technology, September 2001.
7. T. Calders and J. Paredaens. Axiomatization of frequent sets. In *Proc. ICDT*, pages 204–218, 2001.
8. B. Goethals and J. Van den Bussche. A priori versus a posteriori filtering of association rules. In *ACM SIGMOD Workshop DMKD*, 1999.
9. B. Goethals and J. Van den Bussche. On supporting interactive association rule mining. In *Proc. DaWaK*, pages 307–316, 2000.
10. D. Groth and E. Robertson. Discovering frequent itemsets in the presence of highly frequent items. In *In INAP Int'l Conf. Applications of Prolog RBDM workshop*, 2001.
11. J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proc. ACM SIGMOD*, pages 1–12, 2000.
12. S. Hettich and S. D. Bay. *The UCI KDD Archive. [http://kdd.ics.uci.edu]*. Irvine, CA: University of California, Department of Information and Computer Science, 1999.
13. D. Knuth. *Fundamental Algorithms.* Addison-Wesley, Reading, Massachusetts, 1997.
14. H. Mannila and H. Toivonen. Multiple uses of frequent sets and condensed representations. In *Proc. KDD*, 1996.
15. H. Mannila and H. Toivonen. Levelwise search and borders of theories in knowledge discovery. *DMKD*, 1(3):241–258, 1997.
16. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *Proc. ICDT*, pages 398–416, 1999.
17. J. Pei, J. Han, and R. Mao. Closet: An efficient algorithm for mining frequent closed itemsets. In *ACM SIGMOD Workshop DMKD*, 2000.