Automated support specification for efficient mining of interesting association rules

Wen-Yang Lin

Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan. E-mail: wylin@nuk.edu.tw

Ming-Cheng Tseng

Institute of Information Engineering, I-Shou University, Kaohsiung 840, Taiwan. E-mail: clark.tseng@msa.hinet.net

Correspondence to: Wen-Yang Lin, Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung 811, Taiwan. E-mail: wylin@nuk.edu.tw

Abstract.

In recent years, the weakness of the canonical support-confidence framework for associations mining has been widely studied. One of the difficulties in applying association rules mining is the setting of support constraint. A high-support constraint avoids the combinatorial explosion in discovering frequent itemsets, but at the expense of missing interesting patterns of low support. Instead of seeking the way for setting the appropriate support constraint, all current approaches leave the users to be in charge of the support setting, which, however, puts the users in a dilemma. This paper is an effort to answer this long-standing open question. According to the notion of confidence and lift measures, we propose an automatic support specification for efficiently mining high-confidence and positive lift associations without consulting the users. Experimental results show that the proposed method not only is good at discovering high-confidence and positive lift associations, but also is effective in reducing the spurious frequent itemsets.

Keywords: Data mining; Decision support systems; Association rules; Support specification

1. Introduction

Mining association rules from a large database of business data, such as transaction records, has been a hot topic within the area of data mining. This problem is motivated by applications known as market basket analysis in finding relationships between items purchased by customers, that is, what kinds of products tend to be purchased together [1]. An association rule is an expression of the form $A \Rightarrow B$, where A, B are sets of items. Such a rule reveals that transactions in the database containing items in A tend to contain items in B, and the conditional probability, measured as the fraction of transactions containing A also containing B, i.e., $P(B|A) = P(A \cup B) / P(A)$, is called the *confidence* (*conf*) of the rule. The *support* (*sup*) of the rule is the fraction of the transactions that contain all items both in A and B, i.e., $sup(A \Rightarrow B) = P(A \cup B)$. For an association rule to hold, the support and the confidence of the rule should satisfy a user-specified minimum support called *minsup* and minimum confidence called *minconf*, respectively.

One of the difficulties in applying association rules mining is the setting of support constraint. A high-support constraint avoids the combinatorial explosion in discovering frequent itemsets, but at the expense of missing interesting patterns of low support. However, most rules with high support are obvious and well-known, and it is the rules with low support that provide interesting new insight, such as deviations or exceptions.

Instead of seeking the way for setting the appropriate support constraint, all current approaches leave the users to be in charge of the support setting, which, however, puts the users in a dilemma: how to specify the most appropriate support constraint, either uniform or non-uniform, to discover interesting patterns without suffering from combinatorial explosion and missing some low-support but perceptive rules. The best one can do is either setting the support at the lowest value ever specified or performing a consecutive sequence of mining processes with various constraints to extract the right patterns.

Our intent is to seek as could as possible the rules of high confidence without the need for user specified support constraint. To this end, we proposed an automatic support specification without consulting the users. The idea is taken from the notion of *lift* measure [2] (also called *interest* [3]) and confidence measure. Experimental results show that this specification is good at discovering low-support, but high-confidence and positive lift associations, and is effective in reducing the spurious frequent itemsets.

The remaining of this paper is organized as follows. The problem of support-confidence framework for associations mining and related work are presented in Section 2. In Section 3, we provide a modified association framework and explain the support specification and mining process for this model. An evaluation of the

proposed specification on IBM synthetic data and real census data is described in Section 4. Finally, our conclusions are stated in Section 5.

2. Problem specification and related work

In the past few years, there has been work on challenging the canonical support-confidence framework for associations mining. These efforts can be categorized into two paradigms: extending the constant support constraint and/or seeking substitutes for confidence measure.

The uniform support constraint was first argued in [4] while generalizing the association model into mutiple-level associations on account of item hierarchy. In [4], Han and Fu extended the uniform support constraint to a form of level-by-level, decreasing assignment. That is, items at the same level receive the same minimum support, and higher level items have larger support constraint. This level-wise support specification accounts for their progressive mining approach: An Apriori-like algorithm is performed progressively from the top level to the bottom, and stops at the very level when no frequent itemset is generated.

Another form of association rules mining with non-uniform minimum supports was proposed by Liu et al [5]. Their method allows the users to specify different minimum supports to different items, and the support constraint of an itemset is defined as the lowest minimum item support among the items in the itemset. The motivation is that the supports of items are non-uniform by nature, and high profit items (e.g., TV) usually occur less frequently than low value items (e.g., toothpaste). The multi-supported model was then extended to generalized associations with taxonomy information in [6]. The problem remains tangling.

Wang et al. [7] proposed a bin-oriented, non-uniform support constraint: Items are grouped into disjoint sets, called *bins*, and items within the same bin are regarded as non-distinguished with respect to the specification of minimum support. In particular, each support constraint specifies a set of bins B_1, B_2, …, B_s of the form $SC_i(B_1, B_2, \ldots, B_s) \geq \theta_i$, where $s \geq 0$ and θ_i is a minimum support. The end-user still needs to determine the appropriate bins and θ_i prior to the mining.

In general, the support of an itemset decreases when its length increases. The uniform support constraint may thus hinder the discovery of frequent long itemsets. In some applications, however, it may be interesting to discover associations between long itemsets. To solve this problem, Seno and Karypis [8] used a support constraint that decreases with the length of the itemset, which helps to find long itemsets without generating lots of spurious short itemsets.

There is also work on mining high confidence associations without support constraints [9]. The proposed method, however, is restricted to discover all top rules (*conf* = 100%) with the consequence being given.

As pointed out by many researchers [6][10][11], the primary deficiency of confidence-based associations is the poor predictive ability, i.e., the confidence measure is unable to capture the real implication. For illustration, consider the example transaction database in Table 1. For a minimum support of 30% and minimum confidence of 60%, the following association rule is discovered:

$$\text{Scanner} \Rightarrow \text{Printer} \ (sup = 44.4\%, conf = 66.7\%).$$

One may conclude that this rule is interesting because of its high support and high confidence. However, note that the support of Printer is 77.8%. This means that a customer who is known to purchase Scanner is less likely to buy Printer (by 11%) than a customer about whom we have no information. The rule is misleading as it does not conform to what the direct association means: When people buy Scanner, they are also likely to buy Printer. Instead, buying Scanner and purchasing Printer are negatively associated.

Table 1. A transaction database (D)

TID	Items Purchased
11	PC, Printer, PDA
12	Printer, Notebook
13	Printer, Scanner
14	PC, Printer, Notebook
15	PC, Scanner
16	Printer, Scanner
17	PC, Scanner
18	PC, Printer, Scanner, PDA
19	PC, Printer, Scanner

To remedy the above deficiency, two alternative measures have been proposed. They are *lift* [2] (also known as *interest* [3]) and *conviction* [3]. For an association rule $A \Rightarrow B$, the lift is defined as

$$lift\,(A \Rightarrow B) = \frac{P(A \cup B)}{P(A)P(B)} = \frac{sup(A \cup B)}{sup(A)sup(B)} = \frac{conf\,(A \Rightarrow B)}{sup(B)}.$$

Lift measures the deviation of the rule from independence. The farther the value is from 1, the higher the dependence will be. Lift values above 1 indicate positive dependence, while those below 1 express negative dependence. The conviction is

$$conv\,(A \Rightarrow B) = \frac{P(A)P(\neg B)}{P(A \cup \neg B)} = \frac{1 - sup(B)}{1 - con(A \Rightarrow B)},$$

which measures the implication strength of the rule from statistical independence. The conviction value of a rule is between 0 and ∞. A larger than 1 value indicates that it is greater than the expected presence.

Conviction appears to be preferable to lift in capturing the natural semantics of directed associations because it is directed, i.e., $conv(A \Rightarrow B) \neq conv(B \Rightarrow A)$, whereas lift is not. Furthermore, conviction has better discrimination power. To see this, let us consider a rule $A \Rightarrow B$. Assume $sup(A) = 10\%$, $sup(B) = 90\%$ and $sup(A \cup B) = 10\%$. We get $lift(A \Rightarrow B) = 0.1/(0.1 \times 0.9) = 1.11$ which is only slightly above 1. But this rule has the highest possible conviction value of ∞, which conforms to its 100% confidence.

There is also work on investigating alternatives to the association model for attribute set mining [10][11][12]. Brin et al. [11] first proposed the *correlation* framework, aiming at mining strongly correlated attribute sets. They adopted the well-known *chi-squared* test from classical statistics to measure the correlation. However, this measure, though statistically precise, is prohibitively expensive in constructing the contingency table for each itemset. Another criterion for measuring correlation called *collective strength* (*cs*) was thus proposed in [10], which is defined as follows:

$$cs(A) = \frac{1 - v(A)}{1 - E[v(A)]} \times \frac{E[v(A)]}{v(A)},$$

where $v(A)$ denotes the violation rate of an itemset A, i.e., the fraction of transactions which contain a proper non-null subset of A, and $E[v(A)]$ the expected number of violations of itemset A. However, we like to point out one important aspect: Although in theory the correlation framework would discover strongly correlated items without the support constraint, in practice the support threshold is still of essence. Without a support threshold, the computation cost will be prohibitively expensive and many unqualified itemsets would be generated [10][11]. As such, users still confront the problem of appropriate support specification.

In [12], instead of searching for high confidence associations, they focus on identifying similar itemsets (column pairs) without any support threshold. To this end, they introduced a new measure, called *similarity*, whose symmetric property enables the elimination of support constraint. Although the idea of removing support requirement is somewhat similar inspirit to ours, their approach is not feasible for applications that adhere to the traditional *asymmetric* confidence measure.

3. Interesting association rules and support specification

3.1. Interesting association rules

As concluded from the previous discussion, the primary problems of support-confidence framework are poor predictive ability and uniform support constraint. Although substantial work has provided various methods to alleviate these problems, such as adding the lift or conviction measure, and using non-uniform support constraint, there is still no guideline for users in the support specification. Either the user has to, at the cost of computational efficiency, set the support constraint low enough so as not to lose any interesting rules or risk missing new insight patterns.

Our view for solving this problem is to discharge the end-users from specifying the support constraint. Besides, in light of the previous researches, we adopt the lift measure and non-uniform support constraint to eliminate the deficiency of support-confidence framework. That is, we confine ourselves to seek association rules with high-confidence and positive lift, without the need of user-specified support constraint.

As in [1][13], we consider a given transaction database $D = \{t_1, t_2, \ldots, t_n\}$, where each transaction t_i, $1 \le i \le n$, is a set of items taken from a fixed universe set I of items, i.e., $t_i \subset I$. We refine the canonical association rule model as follows:

Definition 1. Let $ms(a)$ denote the minimum support of an item a, $a \in I$. An itemset $A = \{a_1, a_2, \ldots, a_k\}$, where $a_i \in I$, is frequent if the support of A is larger than the lowest value of minimum support of items in A, i.e., $sup(A) \ge \min_{a_i \in A} sup(a_i)$.

Definition 2. An association rule $A \Rightarrow B$ is strong if

$$sup(A \Rightarrow B) \ge \min_{a_i \in A \cup B} ms(a_i)$$

and $conf(A \Rightarrow B) \ge minconf$.

Definition 3. An association rule $A \Rightarrow B$ is interesting if it is strong and $lift(A \Rightarrow B) > 1$.

Note that the lift measure can be replaced by conviction; indeed, $lift(A \Rightarrow B) > 1$ if and only if $conv(A \Rightarrow B) > 1$. That is, the specification derived from conviction is the same as that from lift. We use lift instead of conviction because it is a more general term in business applications [2].

3.2. The confidence-lift-based support specification

However, as long as support is still the primary determining factor in the initial itemset generation, either the user has to set the initial support parameter low enough so as to not lose interesting rules in the output or risk losing

some important rules. The basic idea of our approach is to "push" the confidence and lift measure into the support constraint to prune the spurious frequent itemsets that fail in generating interesting associations as early as possible. First, let us show how to specify the constraint to reduce the frequent itemsets that fail in generating strong associations.

Lemma 1. Let $A \cup B$ be a frequent itemset, $A \cap B = \varnothing$, and without loss of generality, $a \in A$ and a be the one with the smallest support over all items in $A \cup B$, i.e., $sup(a) = \min_{a_i \in A \cup B} sup(a_i)$. The association rule $A \Rightarrow B$ is strong if $ms(a) = sup(a) \times minconf$.

Proof. According to Definition 1, $sup(A \cup B) \geq ms(a)$. Furthermore, since $a \in A$, we have

$sup(A) \leq sup(a)$. Now if $ms(a) = sup(a) \times minconf$, it follows that

$$\frac{sup(A \cup B)}{sup(A)} \geq \frac{ms(a)}{sup(A)} \geq \frac{ms(a)}{sup(a)} = minconf. \blacksquare$$

Note that any item would become the smallest supported member of some frequent itemset. Thus, the minimum support of an item a, for $a \in I$, according to Lemma 1, can be specified as follows:

$$ms(a) = sup(a) \times minconf. \quad (1)$$

Note that Lemma 1 does not imply that the rule $B \Rightarrow A$ is strong. This is because the confidence measure is not symmetric over the antecedence and consequence. Therefore, Eq. 1 does not guarantee that all rules generated from the frequent itemsets are strong.

Example 1. Consider Table 1. Let *minconf* = 50% and the minimum supports of items be set according to Eq. 1. Then we have $ms(\text{PC}) = 6/9 \times 1/2 = 33\%$, $ms(\text{Printer}) = 7/9 \times 1/2 = 39\%$, and $ms(\text{Notebook}) = 2/9 \times 1/2 = 11\%$. It can be verified that {PC, Printer, Notebook} is a frequent itemset. Consider the following two rules generated from this itemset

$r1$: Printer $\Rightarrow$ PC, Notebook (*sup* = 11%, *conf* = 14%),

$r2$: PC, Notebook $\Rightarrow$ Printer (*sup* = 11%, *conf* = 100%).

Clearly, rule $r1$ is not strong while $r2$ is.

Now that we have known how to specify the support constraint to obtain strong associations, our next step is to consider how to specify the constraint to generate interesting associations.

Consider any association rule derived from a frequent 2-itemset, say r: $a \Rightarrow b$. Assume that r is strong. Rule r is interesting if

$$\frac{sup(\{a,b\})}{sup(a)sup(b)} > 1.$$

Since $\{a, b\}$ is frequent, it is true that

$$sup(\{a, b\}) \geq \min\{ms(a), ms(b)\}.$$

Without loss of generality, let $\min\{ms(a), ms(b)\} = ms(a)$. To make r an interesting association rule, the minimum support of a should be set to at least of the value $sup(a) \times sup(b)$, i.e., $ms(a) > sup(a) \times sup(b)$. Since our intent is to make all generated rules interesting to the users, the minimum support for any item $a_i \in I$ can be set as

$$ms(a_i) = sup(a_i) \times \max_{a_j \in I - \{a_i\}} sup(a_j). \quad (2)$$

Although Eq. 2 is derived from rules consisting of only two items, the following lemma shows that this setting suffices for all strong association rules.

Lemma 2. Let I be a set of items and the minimum support of each item is specified according to Eq. 2. Then any strong association rule $A \Rightarrow B$, for $A, B \subset I$ and $A \cap B = \varnothing$, is interesting, i.e.,

$$\frac{sup(A \cup B)}{sup(A) \times sup(B)} > 1.$$

Proof. Since $A \Rightarrow B$ is strong, $sup(A \cup B) \geq \min_{a_i \in A \cup B} ms(a_i)$. Specifically, let $a = a_i$. It follows that

$$sup(A \cup B) \geq ms(a_i) = sup(a_i) \times \max_{a_j \in I - \{a_i\}} sup(a_j).$$

Since $A \cap B = \varnothing$, a_i belongs to either A or B. Without loss of generality, let $a_i \in A$. It is easy to show that

$$sup(A) \leq sup(a_i) \text{ and } sup(B) \leq \max_{a_j \in I - \{a_i\}} sup(a_j).$$

The lemma then follows. ■

Note that the support constraint specified in Eq. 2 only provides a sufficient condition for obtaining interesting association rules from frequent itemsets. That is, it is an upper bound in guaranteeing that all frequent itemsets will not generate associations having negative lift ($lift < 1$). There may exist some itemsets that are infrequent with respect to this constraint but can generate positive lift associations.

Example 2. Let us consider Table 1 again. The minimum supports of items PC, Printer, and Notebook, according to Eq. 2, will be set as $ms(\text{PC}) = 6/9 \times 7/9 = 52\%$, $ms(\text{Printer}) = 7/9 \times 6/9 = 52\%$, and $ms(\text{Notebook}) =$

$2/9 \times 7/9 = 17\%$, respectively. Clearly, the itemset {PC, Printer, Notebook} is not frequent for sup({PC, Printer, Notebook}) = 11% < ms(Notebook) and so rule $r1$ will not be generated. But it can be verified that $lift(r1) > 1$.

Now, if our purpose is to construct all associations without missing any interesting rules, we should seek other form of support specification. The intuition is to set the minimum support of an item a as for accommodating all frequent itemsets that consist of a as the smallest supported item capable of generating at least one nonnegative lift association rule.

Consider the association rule $A \Rightarrow B$ derived from a frequent itemset $A \cup B$. Without loss of generality, let a and b be the smallest supported items in itemsets $A \cup B$ and B, respectively, i.e., $sup(a) = \min_{a_i \in A \cup B} sup(a_i)$, $sup(b) = \min_{a_i \in B} sup(a_i)$, and $sup(a) \le sup(b)$. The following conditions hold

$$sup(A \cup B) \ge ms(a),\ sup(A) \le sup(a), \text{ and } sup(B) \le sup(b).$$

Thus, to make $lift(A \Rightarrow B) \ge 1$, we should specify $ms(a) \ge sup(a) \times sup(b)$. Note that b can be any item in the item set I except a, and $sup(b) \ge sup(a)$. What we need is the smallest qualified item, i.e., $b = \min\{a_i \mid a_i \in I - \{a\}$ and $sup(a_i) \ge sup(a)\}$. Let $I = \{a_1, a_2, ..., a_n\}$ and $sup(a_i) \le sup(a_{i+1})$, $1 \le i \le n - 1$. The minimum item support with respect to nonnegative lift (LS) can be specified as follows:

$$ms(a_i) = \begin{cases} sup(a_i) \times sup(a_{i+1}), & \text{if } 1 \le i \le n-1 \\ sup(a_i), & \text{if } i = n \end{cases} \quad (3)$$

Now we have two separate support settings. One is based on the confidence measure and the other is on lift. To prune the spurious frequent itemsets to make most of the generated rules interesting, we combine these two specifications as shown below, which we call the confidence-lift support constraint (*CLS*).

$$ms(a_i) = \begin{cases} sup(a_i) \times \max\{min\,conf,\ sup(a_{i+1})\}, & \text{if } 1 \le i \le n-1 \\ sup(a_i), & \text{if } i = n \end{cases} \quad (4)$$

Example 3. Let $minconf$ = 50% and the sorted set of items in Table 1 be {Notebook, PDA, PC, Scanner, Printer}. The minimum item supports will be ms(Notebook) = $2/9 \times 1/2$ = 11%, ms(PDA) = $2/9 \times 6/9$ =15%, ms(PC) = $6/9 \times 6/9$ = 44%, ms(Scanner) = $6/9 \times 7/9$ = 52%, and ms(Printer) = 7/9 = 78%.

3.3. Methods for mining interesting associations using CLS

With Eq. 4, the traditional associations mining process can be refined as follows. The users specify the minimum confidence (*minconf*) and wait for the results; they no longer have to specify the minimum support. The only focus is on how strong the rules they expect to see and whether these rules are interesting.

The task of associations mining remains the same. First, the set of frequent itemsets are generated and from which the interesting association rules are constructed. For the itemset generation subtask, one might expect to adopt the well-known Apriori algorithm [13]. However, the *downward closure property* on which the Apriori algorithm heavily relies may fail in the case of multiple minimum supports. For example, consider four items *a*, *b*, *c*, and *d* that have minimum supports specified as $ms(a) = 15\%$, $ms(b) = 20\%$, $ms(c) = 4\%$, and $ms(d) = 6\%$. Clearly, a 2-itemset $\{a, b\}$ with 10% support is discarded for $10\% < \min\{ms(a), ms(b)\}$. According to the downward closure, the 3-itemsets $\{a, b, c\}$ and $\{a, b, d\}$ will be pruned even though their supports may be larger than $ms(c)$ and $ms(d)$, respectively. To solve this problem, Liu et al. [5] proposed a concept called *sorted closure property*, which requires that all items within an itemset are sorted in increasing order of their minimum supports. In the following, we detail the theories behind the concept of sorted closure property and how it is useful in itemset pruning. Since no proof is provided in [5], we provide the proofs for completeness.

Lemma 3 (Sorted closure). If a sorted *k*-itemset $\langle a_1, a_2, \ldots, a_k\rangle$, for $k \geq 2$ and $ms(a_1) \leq ms(a_2) \leq \ldots \leq ms(a_k)$, is frequent, then all of its sorted subsets with $k-1$ items are frequent, except for the subset $\langle a_2, a_3, \ldots, a_k\rangle$.

Proof. *A* *k*-itemset $\langle a_1, a_2, \ldots, a_k\rangle$ has *k* subsets with $k-1$ items, which can be divided into two groups according to the inclusion of item a_1 or not, i.e.,

group 1: $\langle a_1, a_2, \ldots, a_{k-1}\rangle$, $\langle a_1, a_2, \ldots, a_{k-2}, a_k\rangle$, …, $\langle a_1, a_3, \ldots, a_k\rangle$,
group 2: $\langle a_2, a_3, \ldots, a_k\rangle$.

Note that all of the itemsets in Group 1 have the same lowest minimum item support as that of $\langle a_1, a_2, \ldots, a_k\rangle$, i.e., $ms(a_1)$, while $\langle a_2, a_3, \ldots, a_k\rangle$ does not, which is $ms(a_2)$. Since $ms(a_2) \geq ms(a_1)$, the lemma follows. ■

Example 4. Consider four items *a*, *b*, *c* and *d* with $ms(a) = 3\%$, $ms(b) = 5\%$, $ms(c) = 8\%$, and $ms(d) = 10\%$, respectively. Let $\langle a, b, c, d\rangle$ be a sorted 4-itemset. Then, of all its 3-subsets, $\langle a, b, c\rangle$, $\langle a, b, d\rangle$, and $\langle a, c, d\rangle$ have the same minimum support $ms(a) = 3\%$, while $\langle b, c, d\rangle$ has $ms(b) = 5\%$. The fact that $\langle a, b, c, d\rangle$ is infrequent implies that $\langle a, b, c\rangle$, $\langle a, b, d\rangle$, $\langle a, c, d\rangle$ rather than $\langle b, c, d\rangle$ are infrequent.

The following lemma derived from the sorted closure provides the intuition for performing an Apriori-like itemset pruning.

Lemma 4. For $k \geq 3$, any *k*-itemset $A = \langle a_1, a_2, \ldots, a_k\rangle$ generated by procedure apriori-gen(L_{k-1}) [13] can be pruned if there exists one $(k-1)$ subset of *A*, say $\langle a_{i_1}, a_{i_2}, \ldots, a_{i_{k-1}}\rangle$, such that $\langle a_{i_1}, a_{i_2}, \ldots, a_{i_{k-1}}\rangle \notin L_{k-1}$, for $a_{i_1} = a_1$ or $ms(a_1) = ms(a_2)$.

Proof. It is straightforward from the contrapositive statement in Lemma 3. ■

Example 5. Consider a candidate 4-itemset $\langle a, b, c, d\rangle$. Note that $\langle a, b, c\rangle$, $\langle a, b, d\rangle$, and $\langle a, c, d\rangle$ have the same minimum support $ms(a)$ as that of $\langle a, b, c, d\rangle$. Therefore, if one of them is not frequent, $\langle a, b, c, d\rangle$ is not frequent

as well. On the other hand, consider the other 3-subset $\langle b, c, d\rangle$. If it is not frequent, then $sup(\langle b, c, d\rangle) < ms(b)$. Under this condition if we further know $ms(a) = ms(b)$, we can be sure that $\langle a, b, c, d\rangle$ is infrequent due to $sup(\langle a, b, c, d\rangle) \le sup(\langle b, c, d\rangle) < ms(b) = ms(a)$.

Figure 1 shows C_k-gen(L_{k-1}), the procedure for generating all candidate k-itemsets with $k \ge 3$, which consists of two steps: First call apriori-gen to produce candidate itemsets and then prune from C_k those itemsets that satisfy Lemma 4.

In [5], Liu et al. also proposed an algorithm, called MSapriori, for generating frequent itemsets with multiple minimum supports. Since our CLS specification is a form of multiple minimum supports, one might expect to use the MSapriori algorithm to accomplish the first subtask. The specific procedure for generating candidate 2-itemsets in MSapriori, however, is not effective in this case. The reason is that under the CLS specification, all items are frequent 1-itemsets and so the downward closure still holds. The simple apriori-gen procedure indeed is more effective in generating the candidate 2-itemsets. In general, most effective Apriori-like algorithms can be adapted to fit the CLS specification with the following modifications:

procedure C_k -gen(L_{k-1})
  C_k = apriori-gen(L_{k-1}); /* Joins L_{k-1} with L_{k-1} */
  for each itemset $A = \langle a_1, a_2, \ldots, a_k\rangle \in C_k$ **do**
    for each (k-1)-subset $A' = \langle a_{i_1}, a_{i_2}, \ldots, a_{i_{k-1}}\rangle$ of A **do**
      if $a_1 = a_{i_1}$ **or** $ms(a_1) = ms(a_2)$ **then**
        if $A' \notin L_{k-1}$ **then** delete A from C_k;

Fig. 1. *Procedure C_k-gen(L_{k-1})*

1. Scan the database D to obtain the support of each item and set the minimum item supports according to Eq. 4. Set L_1 as the sorted list of items in ascending order of their minimum item supports.
2. Apply apriori-gen in the subsequent candidate generation step to generate candidate 2-itemsets, while invoke C_k-gen for the other candidate k-itemsets, for $k \ge 3$.

A modified Apriori algorithm that adopts the CLS specification, called CLS_Apriori, is described in Figure 2.

Algorithm: CLS_Apriori
Input: Database D, and minimum confidence *minconf*
Output: L, frequent itemsets in D
Method:

```
Compute ms(a) for all a ∈ I; /* compute the minimum support for all items */
L1 = sort(I); /* ascending sort according to ms(a) */
for (k = 2; Lk–1 ≠ ∅; k++) do
   if k = 2 then C2 = apriori-gen(L1);
   else Ck = Ck -gen(Lk-1);
   for each transaction t ∈ D do
      Ct = subset(Ck, t);
      for each candidate A ∈ Ct do
         Increase the count of A;
   end for
   Lk = {A ∈ Ck |sup(A) ≥ ms(A[1])}; /* A[1] denote the first item in A */
end for
return L = ∪k Lk;
```

Fig. 2. *Algorithm CLS_Apriori*

Finally, for the second subtask, all proposed methods for constructing associations from the frequent itemsets can be used. Figure 3 shows a general description for generating association rules from the frequent itemsets.

Algorithm: Assoc_gen
Input: L, the set of frequent itemsets, and minimum confidence, *minconf*
Output: R, the set of association rules
Method:

```
for each frequent itemset l ∈ L do
   for each nonempty subset s of l do
      if sup(l)/sup(s) ≥ minconf then
         output the rule "s ⇒ (l–s)";
```

Fig. 3. *Algorithm Assoc_gen*

Example 6. Let *minconf* = 50%. Figure 4 shows the process of performing CLS_Apriori for the example dataset in Table 1.

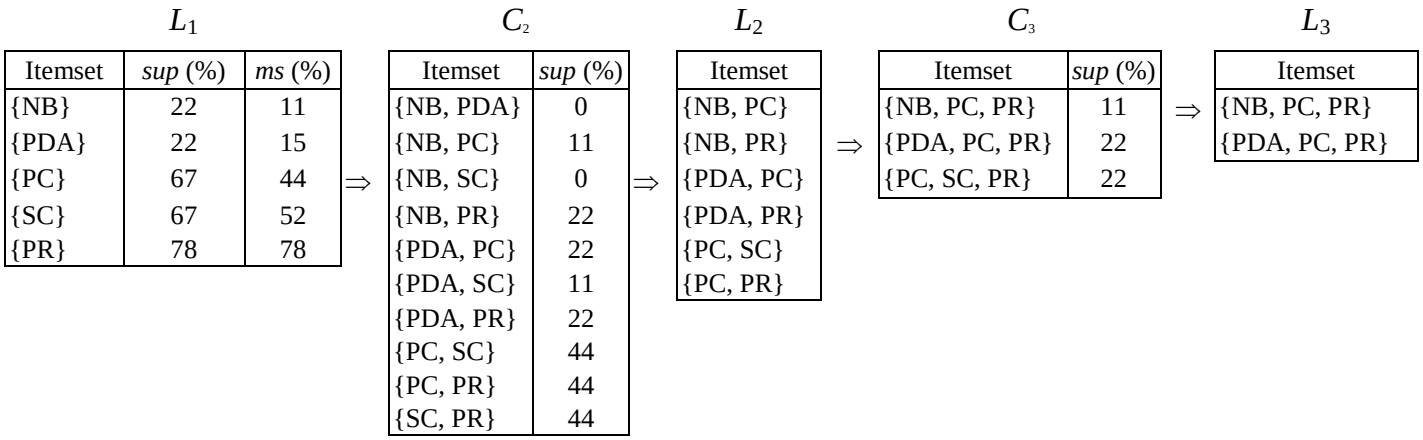


Fig. 4. *An illustration of CLS_Apriori*

4. Experiments

In this section, we evaluate the proposed confidence-lift support specification as opposed to standard uniform specification (US, set to 1%), random support specification (RS, randomly set between 1% and 10%), and the varied item support specification (VIS, $\alpha = 0.6$) [5]. For uniform specification, the well-known Apriori algorithm is used, while for multiple support specifications, i.e., RS and VIS, the MSapriori algorithm is adopted. The evaluation is examined from three aspects: the number of candidate itemsets, the ratio of frequent itemsets that are effective in generating interesting rules, and the execution time. All experiments are performed on an Intel Pentium-II 350 with 192MB RAM, running Windows 98. Both synthetic data and real census data are considered.

4.1. Synthetic data

We use two synthetic data sets generated from IBM synthetic data generator [1]: T5.I2.D100K and T10.I4.D100K. Characteristics of these two data sets are shown in Table 2.

First, we compare the ratio of effective frequent itemsets, denoted as $|Fe|/|F|$. As shown in Figure 5, almost all frequent itemsets generated by CLS are effective; the ratio reaches 1 when *minconf* ≥ 30%. The effective ratio for the other specifications, however, decreases when the *minconf* increases. This means that the ratio of spurious frequent itemsets increases when the *minconf* becomes higher, and more uninteresting rules will be generated.

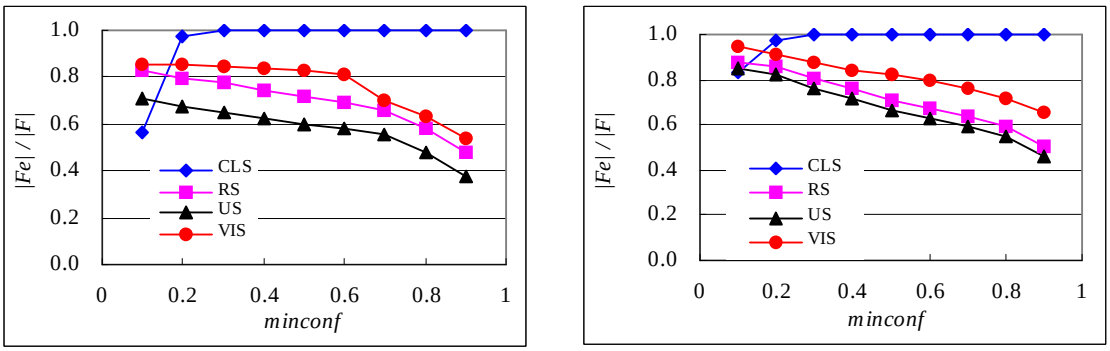
Table 2. *Characteristics of synthetic data sets*

Parameter		T5.I2.100K	T10.I4.100K
$\|D\|$	Number of transactions	85,792	98,272
N	Number of items	94	156
f_{min}	Minimum item frequency	0.27%	0.09%
f_{max}	Maximum item frequency	20.71%	27.97%
f_{avg}	Average item frequency	5.43%	6.40%
f_{stv}	Standard deviation of item frequencies	4.42%	5.50%

We then compare the number of candidates vs different *minconf*s for each specification. The result is depicted in Figure 6. Note that only our CLS specification is affected by *minconf*; all the other minimum support specifications remain the same. For CLS, the value of minimum item support increases when the *minconf* increases, and so the number of candidates decreases. The number of candidates for other specifications remains constant through all different *minconf*s. This phenomenon asserts the primary advantage of CLS over other specifications: it can adapt itself to appropriate item support in response to different confidence thresholds. On the one hand, it explores more hidden but effective frequent itemsets than other ways of specification in finding low-confidence rules. On the other hand, it refrains from generating a large amount of spurious candidates that cannot lead to high-confidence rules.

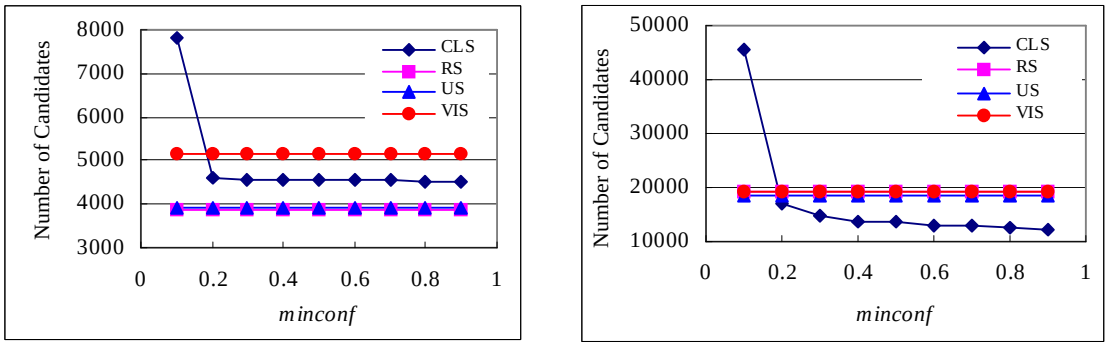
Figure 7 shows the execution time of the Apriori-like algorithms running under different support specifications. Intuitionally, the execution time of an Apriori-like algorithm is proportional to the number of candidates. The results conform approximately to this rule of thumb.

Finally, we compare the average running time spent on discovering each effective frequent itemsets, denoted as $T/|Fe|$, using different support specifications. We adopt this criterion instead of the total cost for finding all effective frequent itemsets because this can alleviate the bias that different minimum support thresholds would lead to different sets of frequent itemsets. As the results shown in Figure 8, our method performs better than the others, especially for higher *minconf*s. Further, the execution time for all specifications except our CLS increases as the *minconf* increases; with our CLS, the cost increases for T5.I2.D100K while decreases dramatically for T10.I4.D100K as $minconf \geq 50\%$.



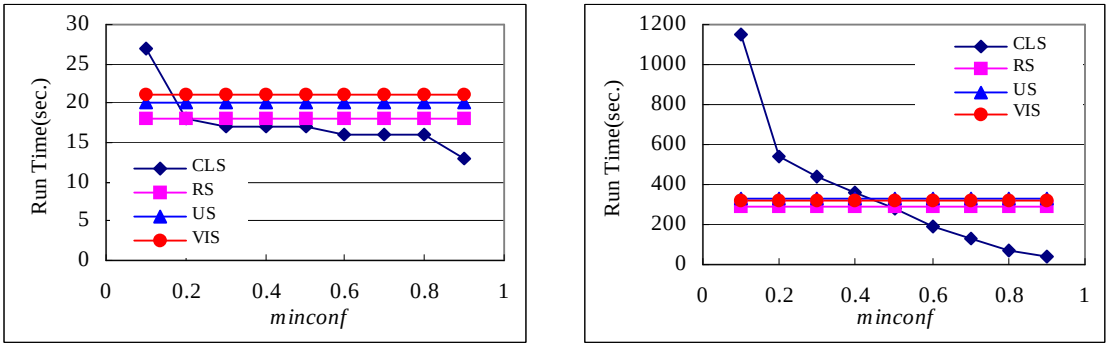
(a) T5.I2.D100K (b) T10.I4.D100K

Fig. 5. *Synthetic data: Comparison of ratios of effective frequent itemsets*



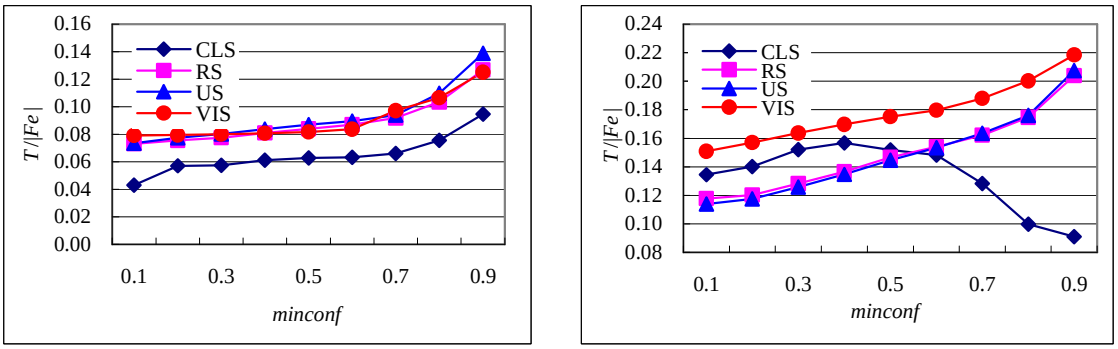
(a) T5.I2.D100K (b) T10.I4.D100K

Fig. 6. *Synthetic data: Comparison of number of candidates*



(a) T5.I2.D100K (b) T10.I4.D100K

Fig. 7. *Synthetic data: Comparison of execution time*



(a) T5.I2.D100K (b) T10.I4.D100K

Fig. 8. *Synthetic data: Comparison of average execution time for effective frequent itemset*

4.2. Census data

We chose two sets of real data from the census of United States 1990 [14]. One is a small database (tiny.dat) with 171 items and 3,649 tuples, another set is a large database (small.dat) with 175 items and 36,305 tuples. Each entry has 11 attributes, each of which is represented as a decimal number. Characteristics of these two data sets are listed in Table 3. The most important difference of these synthetic data is item frequency: The smallest frequency is less than 0.03% while the largest is over 86%, leading to over 10% of standard deviation.

Table 3. *Characteristics of census data sets*

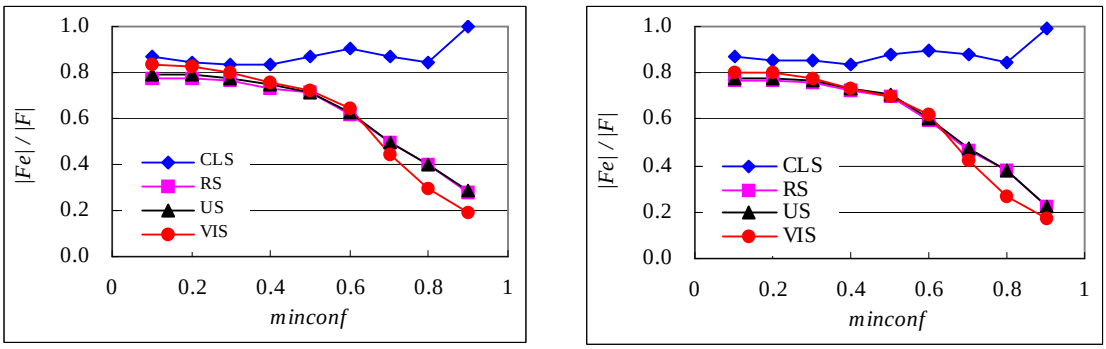
Parameter		Tiny	Small
$\|D\|$	Number of transactions	3,649	36,305
N	Number of items	171	175
f_{min}	Minimum item frequency	0.03%	0.02%
f_{max}	Maximum item frequency	87.28%	86.74%
f_{avg}	Average item frequency	4.09%	4.00%
f_{stv}	Standard deviation of item frequencies	11.07%	10.95%

The comparison of the ratio of effective itemsets is shown in Figure 9. Note that the result for Tiny is analogous to that for Small because these two data sets have similar statistics in item distribution. Comparing this to Figure 5, we observe that: (1) Although CLS exhibits less effective ratio, it maintains at a high value; and (2) the

effective ratio for all other specifications decreases dramatically when the confidence setting increases. This phenomenon is mainly due to large variance in item frequency. The larger the variance is the more difficult is it to determine an appropriate minimum support.

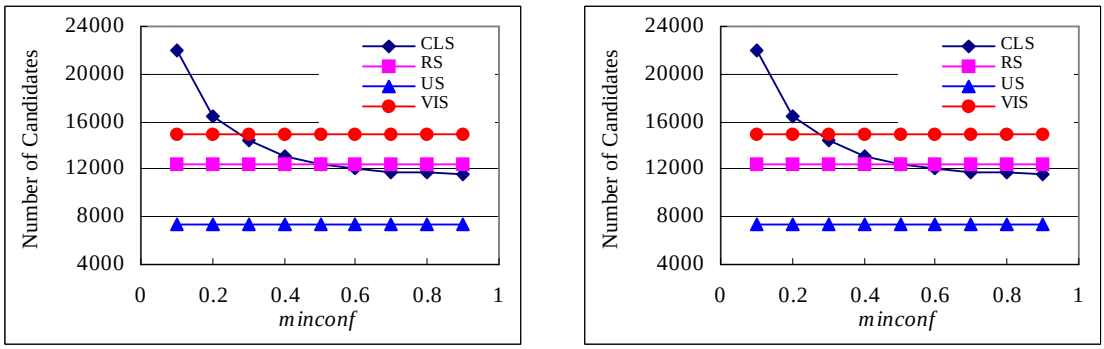
Like the comparison of effective ratio, both census data exhibit the same picture in the number of candidates in Figure 10. The performance evaluation in Figure 11, however, shows different pictures. With CLS, run time reduction for the Small data set is more significant than that for Tiny. Ranks of the other three specifications are different as well.

The results for evaluating $T/|Fe|$ are shown in Figure 12. Both census data exhibit similar performance. Comparing this to Figure 8, although our method outperforms the others in most cases, the gain is less significant.



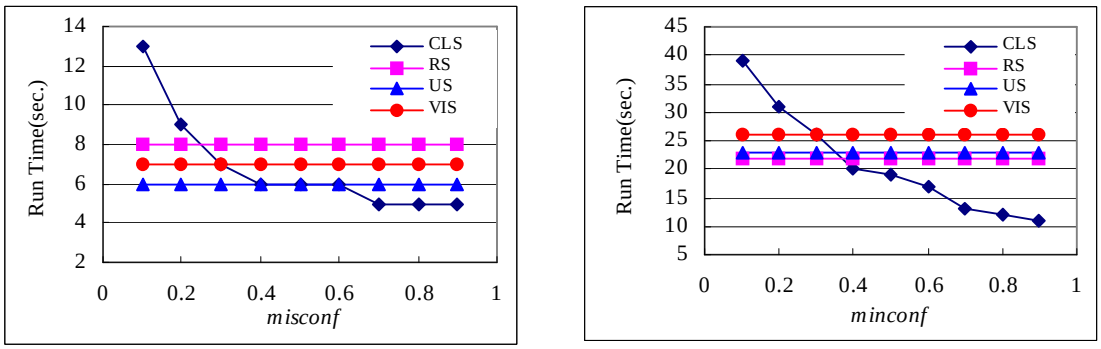
(a) Tiny (b) Small

Fig. 9. *Census data: Comparison of ratios of effective frequent itemsets*



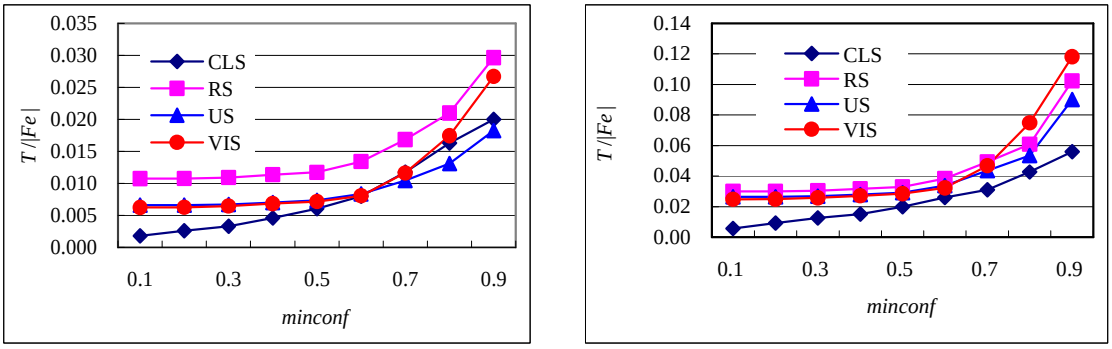
(a) Tiny (b) Small

Fig. 10. *Census data: Comparison of number of candidates*



(a) Tiny (b) Small

Fig. 11. *Census data: Comparison of execution time*



(a) Tiny (b) Small

Fig. 12. *Census data: Comparison of average execution time for effective frequent itemset*

5. Conclusions

We have investigated in this paper the problem of mining interesting association rules without the user-specified support constraint. We proposed a confidence-lift-based support constraint which can be automatically derived from the item support. Empirical evaluation showed that the proposed support specification is good at discovering the high-confidence and positive lift associations, and is effective in reducing the spurious frequent itemsets.

Finally, we should point out that the proposed support specification still cannot find all interesting association rules. In fact, it may miss some interesting rules composed of more than two items because the specification is derived from frequent 2-itemsets. We conjecture that in general there is no way to set automatically minimum

item support without missing any interesting association rules. Currently, we are endeavoring to unveil the facet of this problem.

References

[1] R. Agrawal, T. Imielinski, and A. Swami, Mining association rules between sets of items in large databases. *Proceedings of 1993 ACM-SIGMOD International Conference on Management of Data* (1993) 207-216.

[2] J.A. Berry and G.S. Linoff, *Data Mining Techniques for Marketing, Sales and Customer Support* (John Wiley & Sons, Inc., 1997).

[3] S. Brin, R. Motwani, J.D. Ullman, and S. Tsur, Dynamic itemset counting and implication rules for market-basket data. *Proceedings of 1997 ACM-SIGMOD International Conference on Management of Data* (1997) 207-216.

[4] J. Han and Y. Fu, Discovery of multiple-level association rules from large databases. *Proceedings of the 21st International Conference on Very Large Data Bases* (1995) 420-431.

[5] B. Liu, W. Hsu, and Y. Ma, Mining association rules with multiple minimum supports. *Proceedings of 1999 ACM-SIGKDD International Conference on Knowledge Discovery and Data Mining* (1999) 337-341.

[6] M.C. Tseng and W.Y. Lin, Mining generalized association rules with multiple minimum supports. *Proceedings of International Conference on Data Warehousing and Knowledge Discovery* (2001) 11-20.

[7] K. Wang, Y. He, and J. Han, Mining frequent itemsets using support constraints. *Proceedings of the 26th International Conference on Very Large Data Bases* (2000) 43-52.

[8] M. Seno and G. Karypis, LPMiner: An algorithm for finding frequent itemsets using length-decreasing support constraint. *Proceedings of the 1st IEEE International Conference on Data Mining* (2001).

[9] J. Li and X. Zhang, Efficient mining of high confidence association rules without support thresholds. *Proceedings of the 3rd European Conference on Principles and Practice of Knowledge Discovery in Databases* (1999).

[10] C.C. Aggarwal and P.S. Yu, A new framework for itemset generation. *Proceedings of the 17th ACM Symposium on Principles of Database Systems* (1998) 18-24.

[11] S. Brin, R. Motwani, and C. Silverstein, Beyond market baskets: generalizing association rules to correlations. *Proceedings of 1997 ACM-SIGMOD International Conference on Management of Data* (1997) 265-276.

[12] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J.D. Ullman, and C. Yang, Finding interesting associations without support pruning. *Proceedings of IEEE International Conference on Data Engineering* (2000) 489-499.

[13] R. Agrawal and R. Srikant, Fast algorithms for mining association rules. *Proceedings of the 20th International Conference on Very Large Data Bases* (1994) 487-499.

[14] MPC (Minnesota Population Center), University of Minnesota: IPUMS-USA, Available at: http://www.ipums.umn.edu (Accessed 4th November 2005)