Vietnam J Comput Sci (2015) 2:67–83
DOI 10.1007/s40595-014-0032-7

REGULAR PAPER

An efficient method for mining association rules based on minimum single constraints

Hai Van Duong · Tin Chi Truong

Received: 23 May 2014 / Accepted: 20 October 2014 / Published online: 13 November 2014

Abstract Mining association rules with constraints allow us to concentrate on discovering a useful subset instead of the complete set of association rules. With the aim of satisfying the needs of users and improving the efficiency and effectiveness of mining task, many various constraints and mining algorithms have been proposed. In practice, finding rules regarding specific itemsets is of interest. Thus, this paper considers the problem of mining association rules whose left-hand and right-hand sides contain two given itemsets, respectively. In addition, they also have to satisfy two given maximum support and confidence constraints. Applying previous algorithms to solve this problem may encounter disadvantages, such as the generation of many redundant candidates, time-consuming constraint check and the repeated reading of the database when the constraints are changed. The paper proposes an equivalence relation using the closure of itemset to partition the solution set into disjoint equivalence classes and a new, efficient representation of the rules in each class based on the lattice of closed itemsets and their generators. The paper also develops a new algorithm, called *MAR-MINSC*, to rapidly mine all constrained rules from the lattice instead of mining them directly from the database. Theoretical results are proven to be reliable. Because *MAR-MINSC* does not meet drawbacks above, in extensive experiments on many databases it obtains the outstanding performance in comparison with some of existing algorithms in mining association rules with the constraints mentioned.

H. Van Duong (✉) · T. Chi Truong
Department of Mathematics and Computer Science,
University of Dalat, Dalat, Vietnam
e-mail: haidv@dlu.edu.vn

T. Chi Truong
e-mail: tintc@dlu.edu.vn

Keywords Frequent itemsets · Closed frequent itemsets · Closed itemset lattice · Generators · Association rules · Association rules with constraints · Constraint mining · Equivalence relation · Partition

1 Introduction

For the aim of not only reducing the burden of storage and execution time but also rapidly responding to the demand of users, constraint-based data mining has attracted much interest and attention from researchers. At the beginning, they have designed algorithms to mine data with primitive constraints. A typical example is the one of the frequent itemset discoveries in a transaction database where the primitive constraint is a minimum frequency constraint. Based on frequent itemsets, association rules are mined, where the minimum confidence constraint is other primitive one. More concretely, let $\mathcal{T} = (\mathcal{O}, \mathcal{A}, \mathcal{R})$ be a binary database, where $\mathcal{O}$ is a non-empty set that contains objects (or transactions), $\mathcal{A}$ is a set of attributes (or items) appearing in these objects and $\mathcal{R}$ is a binary relation on $\mathcal{O} \times \mathcal{A}$. The cardinalities of $\mathcal{A}$ and $\mathcal{O}$ are denoted as $m = |\mathcal{A}|$ and $n = |\mathcal{O}|$, respectively (m and n are often very large). Let us denote s_0 as the minimum support threshold and c_0 as minimum confidence threshold, where $s_0, c_0 \in (0; 1]$. The task is to mine frequent itemsets and association rules from $\mathcal{T}$. A basic problem, named (P_1), is that the cardinalities of frequent itemset class $FS(s_0)$ and association rule set $\text{ARS}(s_0, c_0)$ in the worst case are of exponent, i.e., $\text{Max}(\#FS(s_0)) = 2^m - 1 = O(2^m)$ and $\text{Max}(\#\text{ARS}(s_0, c_0)) = 3^m - 2^{m+1} + 1 = O(3^m)$. Therefore, extant algorithms remain riddled with limitations regarding the mining time and the main memory in case the size of $\mathcal{T}$ is quite large. Moreover, for rules that were discovered, it is difficult for users to quickly find the quite small subset of interest if there only have the constraints about support and confidence. To solve this problem (P_1), many more complicated constraints have been introduced into algorithms to only generate association rules related directly to the user's true needs, and to reduce the cost of the mining. Monotonic and anti-monotonic constraints, denoted as $\mathcal{C}_{\rm m}$ and $\mathcal{C}_{\rm am}$ respectively, are considered by Nguyen et al. [25]. They are pushed into an Apriori-like algorithm, named CAP, to reduce the frequent itemsets computation. In [7], the problem is restricted in two constraints that are the consequent and the minimum improvement. Srikant et al. [30] present the problem of mining association rules that include the given items in their two sides. A three-phase algorithm is proposed for mining those rules. First, the constraint is integrated into the Apriori-like candidate generation procedure to find only candidates that contain the selected items. Second, an additional scanning of the database is executed to count the support of the subsets of each mined frequent itemset. Finally, an algorithm based on Apriori principle is applied to generate rules. The concept of convertible constraint is introduced and pushed within the mining process of an FP-growth based algorithm [28]. The authors show that, since frequent itemset mining is based on the concept of prefix-itemsets, it is very easy to integrate convertible constraints into FP-growth-like algorithms. They also state that pushing these constraints into Apriori-like algorithms is not possible. Due to huge input databases, Bonchi et al. [8] propose data reduction techniques and they have been proven to be quite effective in cases of pushing convertible constraints into a level-wise computation. The authors in [21] design the algorithms for discovering association rules with multi-dimension constraints.

By combining the power of the condensed representation (closed itemsets and generators) of frequent itemsets with the properties of $\mathcal{C}_{\rm m}$ and $\mathcal{C}_{\rm am}$ constraints, in [2,3,16,17], we consider some different item constraints and propose efficient algorithms to mine-constrained frequent itemsets. In detail, the work in [2] is to mine all frequent itemsets contained in a specific itemset. An algorithm, called *MINE_FS_CONS*, has been proposed to do this task. In [3], the efficient algorithms *MFS-CC* and *MFS-IC* for mining frequent itemsets with the dualistic constraints are presented. They are built based on the explicit structure of frequent itemset class. The class is split into two sub-classes. Each sub-class is found by applying the efficient representation of itemsets to the suitable generators. And in [16,17], we consider the problem of mining frequent itemsets that (i) include a given subset and (ii) contain no items of another specific subset, or only satisfy the condition (i). Mining frequent itemsets that satisfy both (i) and (ii) is quite complicated because there is a tradeoff among these constraints. However, with a suitable approach, the papers propose efficient algorithms, named *MFS-Contain-IC* and *MFS_DoubleCons*, for discovering frequent itemsets with the constraints mentioned.

It is noted that, our results above only relate directly to frequent itemsets. We, in this paper, are interested in extending the result presented in [16] to association rule mining with many different constraints. The approach based on frequent closed itemset and their generators is still used but the problem is much more complicated. Firstly, let us state our problem as in sub-section below.

1.1 Problem statement

Before stating the problem of our study, we present some common concepts and related notations. Given $\mathcal{T} = (\mathcal{O}, \mathcal{A}, \mathcal{R})$, a set $X \subseteq \mathcal{A}$ is called an itemset. The support of an itemset X, denoted by supp(X), is the ratio of the number of transactions containing X and N, the number of transactions in $\mathcal{T}$. Let s_0, s_1 be the minimum and maximum support thresholds, respectively, where $0 < 1/n \leq s_0 \leq s_1 \leq 1$ and $n = |\mathcal{O}|$. A non-empty itemset A is called *frequent* iff[1] $s_0 \leq \text{supp}(A) \leq s_1$ (if s_1 is equal to 1, then the traditional frequent itemset concept is obtained). For any frequent itemset S', we take a non-empty, proper subset L' from S' ($\varnothing \neq L' \subset S'$) and $R' \equiv S' \backslash L'$. Then, $r : L' \rightarrow R'$ is a rule created by L', R' (or by L', S') and its support and confidence are determined by $\text{supp}(r) \equiv \text{supp}(S')$ and $\text{conf}(r) \equiv \text{supp}(S')/\text{supp}(L')$, respectively. The minimum and maximum confidence thresholds are denoted by c_0 and c_1, respectively, where $0 < c_0 \leq c_1 \leq 1$. The rule r is called an association rule in the traditional manner iff $c_0 \leq \text{conf}(r)$ and $s_0 \leq \text{supp}(r)$ and the set of all association rules is denoted by $\text{ARS}(s_0, c_0) \equiv \{r : L' \rightarrow R' \mid \varnothing \neq L', R' \subseteq_{\mathcal{A}}, L' \cap R' = \varnothing, S' \equiv L' + R', s_0 \leq \text{supp}(r), c_0 \leq \text{conf}(r)\}$

The present study considers the problems that comprise many constraints about support, confidence and sub-items. Such a problem is stated as follows. For additional constraints on two sides of rule, $L_0, R_0 \subseteq \mathcal{A}$, *the goal is to discover all association rules* $r : L' \rightarrow R'$ so that their supports and confidences meet the conditions, $s_0 \leq \text{supp}(r) \leq s_1$, $c_0 \leq \text{conf}(r) \leq c_1$, and their two sides contain the item constraints, $L' \supseteq L_0$, $R' \supseteq R_0$, called *minimum single constraints*. The problem can be described formally as follows.

$$\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1) \equiv \{r : L' \rightarrow R' \in \text{ARS}(s_0, s_1, c_0, c_1) | L' \supseteq L_0, R' \supseteq R_0\} \quad (\text{ARS\_MinSC}),$$

where $\text{ARS}(s_0, s_1, c_0, c_1) \equiv \{r : L' \rightarrow R' \in \text{ARS}(s_0, c_0) | \text{supp}(r) \leq s_1, \text{conf}(r) \leq c_1\}$.

For discussing about the constraints of the problem, it is noted that if $s_1 = c_1 = 1$ and $L_0 = R_0 = \varnothing$, we obtain the problem of mining association rule set $\text{ARS}(s_0, c_0)$ in the traditional meaning. Otherwise, the mined rules may be significant in different application domains such as market-basket

[1] Iff is denoted as if and only if.

analysis, network traffic domain and so on. For instance, the managers or leaders want to increase the turnover of their supermarket based on high valuable items such as gold and iPad. To this aim, a solution is to find an interesting association among two of these items. The proposed problem may help them to answer the question if there is an association or not by setting the constraints $L_0 = \{\text{gold}\}$ and $R_0 = \{\text{iPad}\}$. If there has at least a found rule, it means that the association is existent. Then, it can be used to support for attaining the aim such as showing two of these items on close places which may encourage the sale of the items together and do discount strategies. At the beginning, the confidences of mined rules may be not high because such exceptional rules only have a few their instances. If the mining task received the high value of the *maximum confidence threshold*, it may generate a large number of rules. This makes it easy to miss the low confidence rules but they are of potential significance. Thus, in order to realize and monitor them easily, we should use the small value of *maximum confidence threshold*. After a time, if these rules have higher confidences and become more important, then foreseeing these associations of the items at the early period of the rules may bring about the higher profits for the supermarket.

In the other meaning, using a *maximum confidence threshold* is more general than the fixed value that is always equal to 1. For the maximum support threshold, when the value of s_1 is quite low and that of c_0 is very high, $\text{ARS}(s_0, s_1, c_0, c_1)$ comprises association rules with the high confidences, discovered from low frequent itemsets. This problem is of importance and practical significance. For instance, we want to detect fairly accurate rules from new, abnormal yet significant phenomena despite their low frequency.

Extant algorithms to mine rules with minimum single constraints might encounter problem, named (P_2), such as the generation of many redundant candidate rules and the duplicates of solutions that are then eliminated. The current interest is to find an appropriate approach for mining-constrained association rule set (the rules satisfy minimum single constraints) without (P_2).

1.2 Paper contribution

The contributions of the paper are as follows. First, we present an approach based on the lattice [26,34,37] of closed itemsets and their generators to efficiently mine association rules satisfying the minimum single constraints and the maximum support and confidence thresholds mentioned above. To this approach, we propose a equivalence relation on constrained rule set based on the closure operator [26]. It helps to partition the set of constrained rules, $\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$, into disjoint equivalence rule classes. Thus, each class is discovered independently and the duplication of the solution may be reduced considerably. Moreover, the partition also helps to decrease the burden of saving the supports and confidences of all rules in the same class and be a reliable theoretical basis for developing parallel algorithms in distributed environments. Second, we point out the necessary and sufficient conditions so that the solution of the problem or a certain rule class is existent. If the conditions are not satisfied, the mining process does not need to uselessly take up time for finding the solution. This makes an important contribution to the efficiency of the approach. Third, a new representation of constrained rules in each class is proposed with many advantages as follows: (1) it helps us to have a clear sight about the structure of constrained rule set; (2) the duplication is completely eliminated; (3) all constrained rules are rapidly extracted without doing any direct check on the constraints, $L' \supseteq L_0$ and $R' \supseteq R_0$. Finally, according to the proposed theoretical results, we design a new, efficient algorithm, named *MAR_MinSC* (*Mining all Association Rules with Minimum Single Constraints*) and related procedures to completely, quickly and distinctly generate all association rules satisfying the given constraints.

1.3 Preliminary concepts and notations

Prior to presenting an appropriate approach to discover the rules with minimum single constraints without (P_2), let us recall some of the following basic concepts about the lattice of closed itemsets and the task of association rule mining.

Given $\mathcal{T} = (\mathcal{O}, \mathcal{A}, \mathcal{R})$, we consider two Galois connection operators $\lambda : 2^{\circ} \rightarrow 2^{\mathcal{A}}$ and $\rho : 2^{\mathcal{A}} \rightarrow 2^{\circ}$ defined as follows: $\forall O, A : \varnothing \neq O \subseteq \mathcal{O}, \varnothing \neq A \subseteq \mathcal{A}, \lambda(O) \equiv \{a \in \mathcal{A} | (o, a) \in \mathcal{R}, \forall o \in O\}, \rho(A) \equiv \{o \in \mathcal{O} | (o, a) \in \mathcal{R}, \forall a \in A\}$ and, as convention, $\lambda(\varnothing) = \mathcal{A}, \rho(\varnothing = \mathcal{O})$. We denote $h(A) \equiv \lambda(\rho(A))$ as the closure of A (h is called the closure operation in $2^{\mathcal{A}}$). An itemset A is called closed itemset iff $h(A) = A$ [26]. We only consider non-trivial items in $\mathcal{A}^F \equiv \{a \in \mathcal{A} : \text{supp}(\{a\}) \geq s_0\}$. Let $\mathcal{CS}$ be the class of all closed itemsets together with their supports. With normal order relation "$\supseteq$" over subsets of $\mathcal{A}$, the lattice of all closed itemsets that is organized by Hass diagram is denoted by $\mathcal{LC} \equiv (\mathcal{CS}, \supseteq)$. Briefly, we use $\text{FS}(s_0, s_1) \equiv \{L' : \varnothing \neq L' \subseteq \mathcal{A}, s_0 \leq \text{supp}(L') \leq s_1\}$ to denote the class of all frequent itemsets and $\text{FCS}(s_0, s_1) \equiv \text{FS}(s_0, s_1) \cap \mathcal{CS}$ to denote the class of all frequent closed itemsets. For any two non-empty itemsets G and A, where $\varnothing \neq G \subseteq A \subseteq \mathcal{A}$, G is called a generator [23] of A iff $h(G) = h(A)$ and $(h(G') \subset h(G), \forall G' : \varnothing \neq G' \subset G)$. The class of all generators of A is denoted by $\mathcal{G}(A)$. Since $\mathcal{G}(A)$ is non-empty and finite [5], $|\mathcal{G}(A)| = k$, all generators of A could be indexed as $\mathcal{G}(A) = \{A_1, A_2, \ldots, A_k\}$. Let $\mathcal{LCG} \equiv \{(S, \text{supp}(S), \mathcal{G}(S)) | (S, \text{supp}(S)) \in \mathcal{LC}\}$ be the lattice $\mathcal{LC}$ of closed itemsets together their generators and $\mathcal{FLCG}(s_0, s_1) \equiv \{(S, \text{supp}(S), \mathcal{G}(S)) \in \mathcal{LCG} | S \in \text{FS}(s_0, s_1)\}$ be the lattice of frequent closed itemsets and their generators.

From now on, we shall assume that the following conditions are satisfied, $0 < s_0 \leq s_1 \leq 1, 0 < c_0 \leq c_1 \leq 1, L_0, R_0 \subseteq \mathcal{A}$. (H_0).

Paper organization The rest of this paper is organized as follows. In Sect. 2, we present some approaches to the problem (ARS_MinSC) and the related works. Section 3 shows a partition and a unique representation of constrained association rule set based on closed itemsets and their generators. An efficient algorithm *MAR_MinSC* to generate all association rules with minimum single constraints is also proposed in this section. Experimental results are discussed in Sect. 4. Finally, conclusions and future works are presented in Sect. 5.

2 Approaches to the problem and related works

2.1 Approaches

Post-processing approaches To find association rule set with minimum single constraints $\mathrm{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$, the approaches often perform two phases: (1) association rule set $\mathrm{ARS}(s_0, c_0)$ without the constraints is discovered; (2) the procedures for checking and selecting rules $r : L' \to R'$ that satisfy the constraint $(\mathcal{C}) \equiv \mathrm{supp}(r) \leq s_1, \mathrm{conf}(r) \leq c_1$ and $L' \supseteq L_0, R' \supseteq R_0\}$ are executed. In the phase (1), the rule set, $\mathrm{ARS}(s_0, c_0)$, is able to be mined based on the following simple two methods. One is that it is found by definition, i.e., the class of frequent itemsets $\mathrm{FS}(s_0)$ with the threshold s_0 needs to be mined by a well-known algorithm, such as Apriori [1,23] or Declat [37]. Then, for $\forall S' \in \mathrm{FS}(s_0)$, all rules $r : L' \to R' \in \mathrm{ARS}(s_0, c_0)$, where $\varnothing \neq L' \subset S'$, $R' \equiv S' \setminus L'$ are discovered by an algorithm based on the Apriori principle, such as Gen-Rules [26]. The time for finding $\mathrm{ARS}(s_0, c_0)$ is often quite long because of the reasons as follows: (i) the phase of finding frequent itemsets may generate too many candidates and/or scan the database many times; (ii) the association rule extracting phase often produces many candidates and takes time a lot to calculate the confidences (since the supports of the left-hand sides of the rules may be undetermined). Let us call this post-processing algorithm *PP-MAR-MinSC-1* (*Post Processing-Mining Association Rule with Minimum Single Constraints-1*). The other is to find $\mathrm{ARS}(s_0, c_0)$ based on the lattice $\mathcal{FLCG}$ of frequent closed itemsets and the partition of $\mathrm{ARS}(s_0, c_0)$ as presented in cotemoh4. Instead of exploiting all frequent itemsets, we only need to extract frequent closed itemsets and partition $\mathrm{ARS}(s_0, c_0)$ into equivalence classes. The rules in each class have the same support and confidence that are calculated only once (see in Sect. 3.1.1 for more details). We name *PP-MAR-MinSC-2* for the algorithm of the second method. *PP-MAR-MinSC-2* seems to be more efficient than *PP-MAR-MinSC-1* because it is more suitable in cases support and confidence thresholds are often changed.

Post-processing approaches have the advantage of being simple, but they also have several disadvantages. Due to the enormous cardinality of $\mathrm{ARS}(s_0, c_0)$, the algorithms take a long time to search, but then there might be only a few or even no association rules in $\mathrm{ARS}(s_0, c_0)$ which are of $\mathrm{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$ (the cardinality of $\mathrm{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$ is often quite small compared to that of $\mathrm{ARS}(s_0, c_0)$). Moreover, after finding $\mathrm{ARS}(s_0, c_0)$ is completed, post-processing algorithms have to do direct checks on the constraints, $L' \supseteq L_0$, $R' \supseteq R_0$. This might be time-consuming. In addition, when the constraints are changed based on the demands of online users, recalculating $\mathrm{ARS}(s_0, c_0)$ will uselessly take up time. If, at the beginning, we mine and store $\mathrm{ARS}(s_0, c_0)$ with $s_0 = c_0 = 1/|\mathcal{O}|$, then the computational and memory costs will be very high.

Paper approach To avoid the disadvantages of post-processing approaches and to solve the problem (P_2), the paper proposes a new approach based on three key factors as follows. The first is the lattice $\mathcal{LCG}$ of closed itemsets, their generators and supports. Using $\mathcal{LCG}$ has three advantages: (1) the size of $\mathcal{LCG}$ is often very small in comparison with that of $\mathrm{FS}(s_0)$; (2) $\mathcal{LCG}$ is calculated just once by one of the efficient algorithms such as CHARM-L and MinimalGegenators [36,37], Touch [31] or GenClose [5]; (3) from the lattice $\mathcal{LCG}$, we can quickly derive the lattice $\mathcal{FLCG}_{(\mathcal{C})}$ of frequent closed itemsets satisfying the constraint $(\mathcal{C})$ together with the corresponding generators whenever $(\mathcal{C})$ appears or changes. The second is the equivalence relation based on the closure of two sides of rules $(L \equiv h(L') \subseteq S \equiv \mathrm{h}\,(L'+R'))$. The third is the explicitly unique representation of rules in the same equivalence class $\mathrm{AR}(L, S)$ upon the generators and their closures, $(L, \mathcal{G}\,(L))$ and $(S, \mathcal{G}\,(S))$. In each class, this representation helps us to have a clear sight of the rule structure and to completely eliminate the duplication. An important note is that our method does not need to directly check the generated rules on the constraints, $L' \supseteq L_0$, $R' \supseteq R_0$.

2.2 Related works

To solve the problem (P_1) and improve the efficiency of existing mining algorithms, various constraints have been integrated during the mining process to only generate association rules of interest. The algorithms are mainly based on either the Apriori principle [1] or the FP-growth [18] in combination with the properties of $\mathcal{C}_{\mathrm{am}}$ and $\mathcal{C}_{\mathrm{m}}$ constraints. *FP-bonsai* [9] uses both $\mathcal{C}_{\mathrm{am}}$ and $\mathcal{C}_{\mathrm{m}}$ to mine frequent patterns. The advantage of *FP-bonsai* is that it utilizes $\mathcal{C}_{\mathrm{m}}$ to support the process of pruning candidate itemsets and the database upon $\mathcal{C}_{\mathrm{am}}$. It is efficient on dense databases but not on sparse ones. *Fold-Growth* [29,35] is an improvement of FP-tree using a pre-processing tree structure, named SOTrieIT. The first strength of SOTrieIT is its ability to quickly find frequent 1-itemsets and 2-itemsets with a given support threshold. The second one is that it does not have to reconstruct the tree when the support is changed. A primary drawback of the FP-growth based algorithms is to require the large size of main memory for saving the original database and intermediate projected databases. Thus, if the main memory is not enough, the algorithms cannot be used. Another important limitation of this approach is that it is hard to take full advantage of a combination of different constraints, since each constraint has different properties. For instance, minimum single constraints above regarding support, confidence and item subsets include both $\mathcal{C}_{\mathrm{am}}$ and $\mathcal{C}_{\mathrm{m}}$ constraints whose properties are opposite. Moreover, the approach could take cost a lot to reconstruct FP-tree when mining frequent itemsets and association rules with different constraints. On the contrary, *ExAMiner* [8] is an Apriori-like algorithm. It uses input data reduction techniques to reduce the problem dimensions as well as the search space. It is good at huge input data. However, *ExAMiner* is not suitable with the problem stated in the paper because when the minimum single constraints are changed, the process of reducing input data needs to be started from the original database and generating rules may have time-consuming, direct checks on the constraints. Moreover, the authors in [20] show that the integration of $\mathcal{C}_{\mathrm{m}}$ can lead to a reduction in the pruning of $\mathcal{C}_{\mathrm{am}}$. Therefore, there is a tradeoff between $\mathcal{C}_{\mathrm{am}}$ and $\mathcal{C}_{\mathrm{m}}$ pruning.

For other related results, a constraint, named maximum constraint, is used by [19] to discover association rules with many minimum support thresholds. Each 1-itemset has a minimum support threshold of its own. The authors propose an Apriori-like algorithm for mining large-itemsets and rules with this constraint. Lee et al. [21] design an algorithm to mine association rules with multi-dimensional constraints. An example, max(S.cost) <6 and 200 <min(S.price), is the one of the multi-dimensional constraints, where S is an itemset, and each item of S has two attributes, cost and price. In [14], the CoGAR framework to mine generalized association rules with constraints is presented. Besides the traditional minimum support and confidence, two new constraints, schema and opportunistic confidence, are considered. The schema constraint is similar to that shown in [2] but the approach to solve the problem is different. An algorithm is proposed to discover generalized rules satisfying both these constraints in three phases: (1) the algorithm *CI-Miner* is used to extract schema constrained itemsets; (2) the generalized association rules are exploited by the Apriori-like rule mining algorithm, *RuleGen*; (3) a post-processing filtering algorithm, named CR-Filter, is designed to get the rules satisfying the opportunistic confidence constraint. The concept of periodic constraints is given in [32,33] and new algorithms for mining association rules with this constraint are mentioned. The mining task, firstly, abstracts the variable and then eliminates the solutions falling outside at axiom constraints. The authors in [24] consider the problem of discovering multi-level frequent itemsets with the existent constraints that are represented as a Boolean expression in disjunctive normal form. A technique to model the constraints in the context of use of concept hierarchies is proposed and the efficient algorithms are developed to gain the aim.

Note that most of the previously proposed algorithms for mining association rules with constraints were designed to work on their own constraints. Thus, using them to discover rules based on minimum single constraints may be inefficient. In addition, these algorithms could encounter two important shortcomings; one is to generate many redundant candidates and duplicates of the solution that are then eliminated (the problem (P_2)); the other is that the algorithms need to be rerun from the initial database whenever the constraints are changed. This reduces the mining speed for users.

While the results above seem to be not suitable with the stated problem, an approach that is based on the condensed representation of frequent itemsets might be more efficient. Instead of mining all frequent itemsets, only the condensed ones are extracted. Using condensed frequent itemsets has three primary advantages. First, it is easier to store because its cardinality is much smaller than the size of the class of all frequent itemsets, especially for dense databases. Second, they are mined only once from the database even when the constraints are changed. Third, they can be used to completely generate all frequent itemsets without having to access the database. There are two types of condensed representation. The first type is maximal frequent itemsets [13,22]. Since their cardinality is very small, they can be discovered quickly. All frequent itemsets can be generated from the maximal ones. However, the generation often produces duplicates. In addition, the frequent itemsets generated can lose information about their supports. Therefore, the supports need to be recomputed when mining association rules. The second type is closed frequent itemsets, called maximal ones, and their generators, called minimal ones [10–12,27]. Each closed frequent itemset represents a class of frequent itemsets. Thus, together with its generators, it can be used to uniquely determine all frequent itemsets in the same class without losing information about their supports.

Among two types of the condensed representation above, the second one is probably better and has been proven to be efficient in our previous works. Therefore, in this paper, we propose a new structure and an efficient representation of constrained association rule set based on closed itemsets and their generators. A new corresponding algorithm, named *MAR_MinSC*, is also developed for mining association rules satisfying the minimum single constraint and the maximum support and confidence thresholds.

3 Mining association rules with minimum single constraints

3.1 Partition of association rule set with minimum single constraints

3.1.1 Rough partition

To considerably reduce the duplication of candidates for the solution, we should partition the rule set into disjoint classes based on a suitable equivalence relation. Because the closure operator h of $\mathcal{LCG}$ has some good features, based on it, we propose the following two equivalence relations on $\text{FS}(s_0, s_1)$ and $\text{ARS}(s_0, s_1, c_0, c_1)$.

Definition 1 (*Two equivalence relations on* $\text{FS}(s_0, s_1)$ *and* $\text{ARS}(s_0, s_1, c_0, c_1)$).

(a) $\forall\, A, B \in \text{FS}(s_0, s_1)$, $A \sim_{\mathcal{A}} B \Leftrightarrow h(A) = h(B)$.
(b) $\forall r_k : L_k \to R_k \in \text{ARS}(s_0, s_1, c_0, c_1)$, $k = 1, 2,$

$$r_1 \sim_r r_2 \Leftrightarrow [h(L_1) = h(L_2) \text{ and } h(L_1 + R_1) = h(L_2 + R_2)].$$

Obviously, these are equivalence relations. For any $L \in \text{FCS}(s_0, s_1)$, we use $[L]_{\mathcal{A}} \equiv \{L' \subseteq L\colon L' \neq \varnothing, h(L') = L\}$ to denote the equivalence class of all frequent itemsets with the same closure L. For two arbitrary sets $L, S \in \text{FCS}(s_0, s_1)$ such that $\varnothing \neq L \subseteq S$, supp($S$)/supp($L$)$\in [c_0; c_1]$, the equivalence class of all rules $r : L' \to R'$ so that $h(L') = L$, $h(L'+R') = S$ is denoted by $\text{AR}(L, S) \equiv \{r : L' \to R' \in \text{ARS}(s_0, s_1, c_0, c_1) | L' \in [L]_{\mathcal{A}},\ S' \equiv L'+R' \in [S]_{\mathcal{A}}\}$.

Remark 1 (a) Due to the features of h, $\forall L \in \text{FCS}(s_0, s_1)$, supp($L'$) = supp($L$), $\forall\, L' \in [L]_{\mathcal{A}}$, i.e., all frequent itemsets in the same equivalence class $[L]_{\mathcal{A}}$ have the same support, supp(L).
(b) With $\forall\ r\ :\ L' \to R' \in \text{ARS}(s_0, s_1, c_0, c_1)$, let us set $L \equiv h(L')$, $S' \equiv L'+R'$, $S \equiv h(S')$, then we have $\varnothing \neq L \subseteq S$, supp($S'$) = supp($S$) $\in [s_0, s_1]$, conf(r) $\equiv$ supp(S')/supp(L') = supp(S)/supp(L)$\in [c_0, c_1]$ and $(L, S) \in \text{NFCS}(s_0, s_1, c_0, c_1)$, where

$$\text{NFCS}(s_0, s_1, c_0, c_1) \equiv \{(L, S) \in \mathcal{CS}^2 | S \in \text{FCS}(s_0, s_1), \varnothing \neq L \subseteq S, \text{supp}(S)/\text{supp}(L) \in [c_0, c_1]\}.$$

Thus, for $\forall (L, S) \in \text{NFCS}(s_0, s_1, c_0, c_1)$, all rules in the same equivalence class $\text{AR}(L, S)$ have the same support supp(**S**) and confidence supp(S)/supp(L). This helps to considerably reduce storage needed for the supports of the frequent itemsets and the confidences of association rules.

(c) From (a) and (b), we have the partition of rule set $\text{ARS}(s_0, s_1, c_0, c_1)$ without the item constraints as follows.

$$\text{ARS}(s_0, s_1, c_0, c_1) = \sum\nolimits_{(L,S)\in\text{NFCS}(s_0,s_1,c_0,c_1)} \text{AR}(L, S).$$

Since $\text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1) \subseteq \text{ARS}(s_0, s_1, c_0, c_1)$, the following rough partition of constrained rule set $\text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$ is derived.

Proposition 1 (The rough partition of constrained rule set). *We have:*

$$\text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1) = \sum\nolimits_{(L,S)\in\text{NFCS}(s_0,s_1,c_0,c_1)} \text{AR}_{\supseteq L_0,\supseteq R_0}(L, S),$$

where $\text{AR}_{\supseteq L_0,\supseteq R_0}(L, S) \equiv \{r{:}L' \to R' \in \text{AR}(L, S)|\ L' \supseteq L_0,\ R' \supseteq R_0^{(t)}\}$.

Based on Proposition 1, we can derive the simple post-processing algorithm PP-MAR-MinSC-2 to generate $\text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$. *However, we find that, on many values of the constraints,* $\text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$ *can be empty. Or there are many pairs of closed frequent itemsets* $(L, S) \in \text{NFCS}(s_0, s_1, c_0, c_1)$ *for which the subclasses* $\text{AR}_{\supseteq L_0,\supseteq R_0}(L, S)$ *are empty. When* $\varnothing \neq \text{AR}_{\supseteq L_0,\supseteq R_0}(L, S) \subseteq \text{AR}(L, S)$*, the cardinality of* $\text{AR}(L, S)$ *might still be too large and still has many redundant rules as can be seen in the following example.*

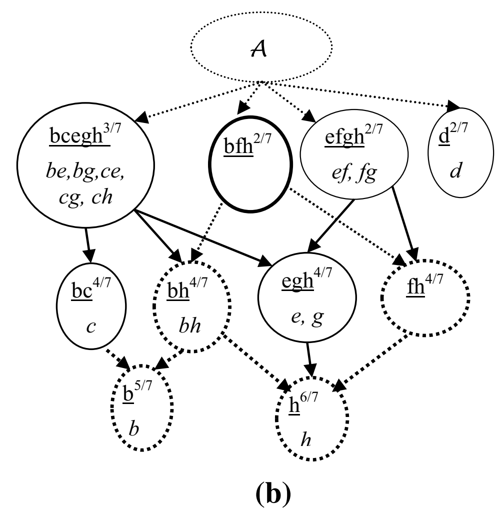
Example 1 (*Illustrating some disadvantages of PP-MAR-MinSC-2*). The rest of this paper considers database $\mathcal{T}$ shown in Fig. 1a. For the minimum support threshold $s_0 = 0.28$, *Charm-L* [37] and *MinimalGenerators* [36] are used to mine a lattice of all closed frequent itemsets and their generators. The result is shown in Fig. 1b. Let us choose the maximum support threshold $s_1 = 0.5$ and the minimum and maximum confidence thresholds $c_0 = 0.4$ and $c_1 = 0.9$, respectively.

(a) Let us consider the constraints L_0= c and $R_0 = f$. The *PP-MAR-MinSC-2* algorithm first generates $|\text{ARS}(s_0, s_1, c_0, c_1)|$ = 134 rules. But after testing them on constraints L_0 and R_0, we obtain $\text{AR}_{\supseteq L_0,\supseteq R_0}(L, S) = \varnothing$ given any rule class of $\text{NFCS}(s_0, s_1, c_0, c_1)$ for the 15 classes. Thus, $\text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1) = \varnothing$.
(b) For another set of constraints $L_0 = h$ and $R_0 = b$, by using *PP-MAR-MinSC-2* to generate 134 rules and to then check them on the constraints, we obtain $|\text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1| = 19$ rules of 4 rule classes (L, S) of $\text{NFCS}(s_0, s_1, c_0, c_1)$, (egh, bcegh), (h,bcegh), (fh,bfh) and (h,bh). The algorithm generates $|\text{ARS}(s_0, s_1, c_0, c_1)\backslash\text{AR}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)| = 115$ redundant candidate rules corresponding to

Fig. 1 **a** Example dataset and **b** the corresponding lattice of closed itemsets

Trans	Items
1	bcegh
2	bcegh
3	bcd
4	defgh
5	bcefgh
6	bfh
7	fh

(a)



$|\text{NFCS}(s_0, s_1, c_0, c_1)| - 4 = 11$ rule classes (L, S) of $\text{NFCS}(s_0, s_1, c_0, c_1)$ so that $\text{AR}_{\supseteq L_0, \supseteq R_0}(L, S) = \varnothing$. Consider the class $(bc, bcegh) \in \text{NFCS}(s_0, s_1, c_0, c_1)$, there are 21 candidate rules in $\text{AR}(bc, bcegh)$ enumerated by *PP-MAR-MinSC-2*. However, after they are tested on the conditions $L_0 \subseteq L'$ and $R_0 \subseteq R'$, the solution subset is empty, $\text{AR}_{\supseteq L_0, \supseteq R_0}(bc, bcegh) = \varnothing$.

(c) For $L_0 = f$ and R_0 = h, the algorithm *PP-MAR-MinSC-2* generates $|\text{ARS}(s_0, s_1, c_0, c_1)| = 134$ rules of 15 pairs $(L, S) \in \text{NFCS}(s_0, s_1, c_0, c_1)$, but there are only 4 rules corresponding to two pairs, (L^1 = fh, S^1 = efgh) and (L^2 = fh, S^2 = bfh) $\in \text{NFCS}(s_0, s_1, c_0, c_1)$ so that $\text{AR}_{\supseteq L_0, \supseteq R_0}(L^i, S^i) \neq \varnothing, i = 1, 2$. For ($L^1$ = fh, S^1 = efgh), it is noted that the number of candidate rules generated in $\text{AR}(L^1, S^1)$ is 9. But there are only 3 rules satisfying the constraints L_0 and R_0, $\text{AR}_{\supseteq L_0, \supseteq R_0}(L^1, S^1) = \{\text{f} \to \text{eh}, \text{f} \to \text{egh}, \text{f} \to \text{gh}\}$. Thus, there exist 6 redundant candidate rules generated in $\text{AR}(L^1, S^1) \backslash \text{AR}_{\supseteq L_0, \supseteq R_0}(L^1, S^1)$.

With the aim of overcoming these disadvantages, we need to find the necessary conditions for the constraint set $(\mathcal{C})$ and the pairs (L, S) so that $\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$ is not empty. As such, we have another representation $\text{AR}^+_{\supseteq L_0, \supseteq R_0}(L, S)$ of $\text{AR}_{\supseteq L_0, \supseteq R_0}(L, S)$ and then obtain a better partition of $\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$.

3.1.2 Necessary conditions for the non-emptiness of $\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$ *and* $\text{AR}_{\supseteq L_0, \supseteq R_0}(L, S)$

Before presenting necessary conditions so that $\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$ and $AR_{L_0, R_0}(L, S)$ are not empty, let us use some additional notations as follows. Assign that

$.S_0^* \equiv L_0 + R_0, C_0 \equiv L_0, C_1 \equiv \mathcal{A} \backslash R_0,\ s_0^* \equiv \max(s_0; c_0.\text{supp}(C_1)),\ s_1^* \equiv \min(s_1; c_1.\text{supp}(L_0))$;

$.S' \equiv L' + R', S \equiv h(S'),\ \text{FCS}_{\supseteq S_0^*}(s_0^*, s_1^*) \equiv \{S \in \text{FCS}(s_0^*, s_1^*) | S \supseteq S_0^*\}$;

$.s_0' \equiv s_0'(S) \equiv \text{supp}(S)/c_1,\ s_1' \equiv s_1'(S) \equiv \min(1; \text{supp}(S)/c_0),\ L \equiv h(L'),\ L_{C_1} \equiv L \cap C_1 = L \backslash R_0, G_{C_1}(L) \equiv \{L_i \in G(L)\ |L_i \subseteq C_1\}, \text{FCS}_{C_0 \subseteq C_1}(s_0', s_1') \equiv \{L_{C_1} \equiv L \cap C_1 | L \in \text{FCS}(s_0', s_1'), L \subseteq C_0, G_{C_1}(L) \neq \varnothing\}, \text{FS}_{C_0 \subseteq L_{C1}} \equiv \{L' \subseteq L_{C_1} | C_0 \subseteq L', L' \neq \varnothing, h(L') = h(L_{C_1})\}$;

$.R_0^* \equiv R_0, R_1^* \equiv R_1^*(L') \equiv S \backslash L', \text{FS}(S \backslash L')_{L', R_0^* \subseteq R_1^*} \equiv \{R' \supseteq R_0^* | \varnothing \neq R' \subseteq R_1^*, h(L' + R') = S\}$;

$.\text{NFCS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1) \equiv \{(L, S) \in \mathcal{CS}^2 | S \in \text{FCS}_{\supseteq S_0^*}(s_0^*, s_1^*), \varnothing \neq L \subseteq S, L_{C_1} \in \text{FCS}_{C_0 \subseteq C_1}(s_0', s_1')\}$.

Then, $\forall (L, S) \in \text{NFCS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$, we have

$$\text{AR}^+_{\supseteq L_0, R_0}(L, S) \equiv \{r : L' \to R' | L' \in \text{FS}_{C_0 \subseteq L_{C1}}, R' \in \text{FS}(S \backslash L')_{L', R_0^* \subseteq R_1^*}\}.$$

We obtain the following proposition.

Proposition 2 (Necessary conditions for the non-emptiness of $\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$ and $\text{AR}_{\supseteq L_0, \supseteq R_0}(L, S)$, and an another representation of $\text{AR}_{\supseteq L_0, \supseteq R_0}(L, S)$).

(a) (*Necessary conditions for* $\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1) \neq \varnothing$) *If* $r : L' \to R' \in \text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1) \neq \varnothing$, *then*

$$(L, S) \in \text{NFCS}(s_0, s_1, c_0, c_1), \quad r \in \text{AR}_{\supseteq L_0, \supseteq R_0}(L, S) \neq \varnothing, \quad \text{where} L = h(L'), \quad S = h(L' + R')$$

and the following necessary conditions are satisfied:

$$L_0 \cap R_0 = \varnothing, s_0^* \leq s_1^*, \text{supp}(S_0^*) \geq s_0^*, \text{supp}(\mathcal{A}) \leq s_1^*. \quad (H_1)$$

Thus, from now on, it is always assumed that (H_1) is satisfied.

(b) (*Necessary conditions for* $\mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L,S) \neq \varnothing$). *For each pair (L, S)* $\in \mathrm{NFCS}(s_0,s_1,c_0,c_1)$, *then for any rule* $r : L' \to R' \in \mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L,S) \neq \varnothing$, *the following necessary conditions are satisfied:*

$$S \in \mathrm{FCS}_{S\supseteq S_0^*}(s_0^*, s_1^*), \quad L_{C_1} \in \mathrm{FCS}_{C_0\subseteq C_1}(s_0', s_1'), \quad L' \in \mathrm{FS}_{C_0\subseteq L_{C1}}, \quad R' \in \mathrm{FS}(S\backslash L')_{L', R_0^*\subseteq R_1^*}.$$

Thus, $(L,S) \in \mathrm{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1) \neq \varnothing$ *and* $\mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L,S) \subseteq \mathrm{AR}^{+}_{\supseteq L_0,\supseteq R_0}(L,S)$.
And we have the result, $\mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L,S) \subseteq \mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)$.

(c) (*Another representation of* $\mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L,S)$). *For each* $(L,S) \in \mathrm{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1) \neq \varnothing$, *then* $\mathrm{FS}_{C_0\subseteq L_{C_1}} \neq \varnothing$ *and*

$$\mathrm{AR}^{+}_{\supseteq L_0,\supseteq R_0}(L,S) = \mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L,S).$$

Corollary 1 (Necessary and sufficient conditions for the non-emptiness of $\mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)$).

(a) *If one or more conditions in* (H_1) *are not satisfied, then* $\mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1) = \varnothing$.

(b) $r : L' \to R' \in \mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1) \neq \varnothing \Leftrightarrow$ *there exist* $(L,S) \in \mathrm{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)$, $L' \in \mathrm{FS}_{C_0\subseteq L_{C1}}$, $R' \in \mathrm{FS}(S\backslash L')_{L',R_0^*\subseteq R_1^*}$ *and* $r : L' \to R' \in \mathrm{AR}^{+}_{\supseteq L_0,\supseteq R_0}(L,S) \neq \varnothing$.

Proof The assertion (a) and the dimension "$\Rightarrow$" of (b) are the obvious consequences of Proposition 2(a) and (b). The reverse dimension "$\Leftarrow$" of (b) is derived from $\mathrm{AR}^{+}_{\supseteq L_0,\supseteq R_0}(L,S) \subseteq \mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L,S) \subseteq \mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)$. □

From Proposition 2 and Corollary 1, we have the following smooth partition of the constrained rule set $\mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)$.

3.1.3 Smooth partition of association rule set with minimum single constraints

Theorem 1 (Smooth partition of constrained rule set) *Assume that the conditions of* (H_1) *are satisfied, then we have:*

$$\mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1) = \sum\nolimits_{(L,S)\in \mathrm{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)} \mathrm{AR}^{+}_{\supseteq L_0,\supseteq R_0}(L,S).$$

This partition is the theoretical basis for the parallel algorithms that independently mine each rule class $\boldsymbol{AR}^{+}_{\supseteq L_0,\supseteq R_0}$ $(\boldsymbol{L},\boldsymbol{S})$ *in the distributed environments. This is an interesting feature when we apply the suitable equivalence relations of mathematics into computer science, a simple yet efficient application of the principle "divide and conquer".*

Example 2 (Illustrating the emptiness of $\mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)$ or $\mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L,S)$ when one of the necessary conditions in (H_1) is not satisfied or $(L,S) \notin \mathrm{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)$).

(a) If one of the necessary conditions in (H_1) is not satisfied, we immediately obtain $\mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1) = \varnothing$. For instant, in Example 1a, we have S_0^* = cf, C_1 = abcdegh, supp(S_0^*) = 1/7 ≈ 0.14 and s_0^* = 0.28, the necessary condition supp(S_0^*) $\geq s_0^*$ is not satisfied. Thus, $\mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1) = \varnothing$ and we do not need to generate $|\mathrm{ARS}(0.28, 0.5, 0.4, 0.9)| = 134$ candidate rules in $\mathrm{ARS}(s_0,s_1,c_0,c_1)$, if only to discard them all afterwards. Another example, for L_0 = d, R_0 = g, then C_0 =d, C_1 = abcdefh, $s_0^* = 0.28$, $s_1^* = 0.26$, we find that the necessary condition $s_0^* \leq s_1^*$ is not satisfied. Thus, $\mathrm{ARS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1) = \varnothing$ and 134 redundant candidate rules are not generated.

(b) If (L, S)∈ $\mathrm{NFCS}(s_0,s_1,c_0,c_1)\backslash\mathrm{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)$, the result $\mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L,S) = \varnothing$ is derived immediately and the pair (L, S) is discarded. In Example 1(b), consider the class (L = bc, S = bcegh), we have S_0^* = bh, C_0 = h, C_1 = acdefgh, $\mathcal{G}$(L)={c} and ($G_{C_1}(L) \neq \varnothing$ and $C_0 \nsubseteq L$). The condition $L_{C_1} \in \mathrm{FCS}_{C_0\subseteq C_1}(s_0', s_1')$ is not satisfied, so (bc,bcegh) $\notin \mathrm{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)$. Thus, we have $\mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L,S) = \varnothing$. Moreover, we also have 10 other redundant candidate classes (L', S') $\in \mathrm{NFCS}(s_0,s_1,c_0,c_1)\backslash\mathrm{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1)$ so that $\mathrm{AR}_{\supseteq L_0,\supseteq R_0}(L',S') = \varnothing$.

We realize that the number of candidate classes (L, S) in $\mathrm{NFCS}(s_0,s_1,c_0,c_1)(\supseteq \mathrm{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0,s_1,c_0,c_1))$ can still be quite large and there remain many redundant candidates that do not satisfy the constraints.

The algorithm *MFCS_FromLattice* $(\mathcal{LCG}^S, C_0, C_1, s_0', s_1')$ shown in Fig. 2 aims to find frequent closed itemsets $\mathrm{FCS}_{C_0\subseteq C_1}(s_0', s_1')$ satisfying the constraints from the lattice $\mathcal{LCG}^S$ (the restricted sub-lattice of $\mathcal{LCG}$ with the root node S). And, especially, we have $\mathrm{FCS}_{\supseteq S_0^*}(s_0^*, s_1^*)$ = *MFCS_FromLattice*($\mathcal{LCG}$, S_0^*, $\mathcal{A}$, s_0^*, s_1^*).

It is important to note that, from the Hass diagram on the lattice $\mathcal{LCG}$, if the concepts of positive and negative borders [23], concerning the *anti-monotonic* and *monotonic* properties of the support and item constraints, are added to the algorithm, then the sub-lattices whose closed itemsets satisfy the corresponding constraints will be generated quickly. For instance, with the monotonic property (supp(L) $\leq s_1'$ and $L \supseteq C_0)^{(M)}$, we illustrate the creation of negative border in the algorithm as follows. At line 3, with the movement from up to down in the lattice, starting at node S, when considering a node N for which one of the conditions in $^{(M)}$ has been violated, we eliminate all sub-branches of N and supplement N into the negative border of the sub-lattice containing the class $\text{FCS}_{C_0 \subseteq C_1}(s'_0, s'_1)$. More specifically, in example 1, we have S = bfg, supp(S) = 2/7, s_0 = 2/7, s_1 = 1, c_0 = 3/4, c_1 = 1, and $C_0 = L_0 = e$ in the sub-lattice $\mathcal{LCG}^S$. S has two direct sub-nodes $L \in$\{bh, fh\}; consider the sub-node $L = bh$, then $s(L) = 4/7 > s'_1$ = min(1; (2/7)/(3/4))=8/21, thus, we eliminate all sub-branches that started at L; consider the sub-node L =fh, then $C_0 \nsubseteq L$; therefore, we also do this. Hence, $\text{FCS}_{C_0 \subseteq C_1}(s'_0, s'_1)$ = \{bfg\} (only one frequent closed itemset S).

For $\forall(L, S) \in \text{NFCS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$, each rule $r : L' \to R'$ in the rule class $\text{AR}^+_{\supseteq L_0, \supseteq R_0}(L, S) = \{r : L' \to R' | L' \in \text{FS}_{C_0 \subseteq L_{C1}},\ R' \subseteq \text{FS}(S \backslash L')_{L', R^*_0 \subseteq R^*_1}\}$ has the left-hand and right-hand sides that do not have an explicit representation, and mining them might still generate many redundant candidates.

3.2 Distinctly generating all association rules in each equivalence class $\text{AR}^+_{\supseteq L_0, \supseteq R_0}(L, S)$

To completely eliminate the generation of duplicate candidates for the solution, based on each class $(L, S) \in \text{NFCS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$ and the generators $\mathcal{G}(L)$, $\mathcal{G}(S)$, we will propose an explicitly unique representation of rules in $\text{AR}^+_{\supseteq L_0, \supseteq R_0}(L, S)$. It will also demonstrate how to completely and distinctly generate all rules in each class.

For that, first of all, we need to show a structure and a unique representation for an extended class of frequent itemsets that are restricted by X and contain an item constraint. Thereby, explicitly unique representations and structures of the right-hand side $R' \in \text{FS}(S \backslash L')_{L', R^*_0 \subseteq R^*_1}$ and the left-hand side $L' \in \text{FS}_{C_0 \subseteq L_{C1}}$ of rules $r : L' \to R'$ in each equivalence class $\text{AR}_{\supseteq L_0, \supseteq R_0}(L, S)$ are derived.

3.2.1 The explicitly unique representation and the structure of an extended class

To briefly present the results regarding the representation of the rule sides, we first consider a fairly general representation of frequent sub-items of Y that are restricted on X with minimum single constraint.

Let $Y, X, Z_0 \subseteq \mathcal{A}$: $Y \neq \varnothing$, $Y \cap X = \varnothing$, $Z_0 \subseteq Y$, we denote $\text{FS}(Y)_{X, \supseteq Z_0} \equiv \{R' \supseteq Z_0 | \varnothing \neq R' \subseteq Y, h(X + R') = h(X + Y)\}$,

$R_{\min} \equiv \text{Minimal}\{R_k \equiv S_k \backslash (X + Z_0),\ S_k \in G(X + Y)\}$,

$R^k_U \equiv \bigcup_{R_j \in R_{\min}, j \leq k} R_j$, and $R_{U,k} \equiv \begin{cases} R^{k-1}_U \backslash R_k, & if\ k \geq 1 \\ \varnothing, & if\ k = 0 \end{cases}$,

where $R_k \in R_{\min}$ and $R_{-,k} \equiv Y \backslash (Z_0 + R^k_U)$.

Note that if $Y = \varnothing$ or $X \cap Z_0 \neq \varnothing$, then $\text{FS}(Y)_{X, \supseteq Z_0} \neq \varnothing$. Otherwise, let us set

$$\text{FS}^*(Y)_{X, \supseteq Z_0} \equiv \{R' \equiv Z_0 + R_k + R'_k + R^{\sim}_k | R_k \in R_{\min}, R'_k \subseteq R_{U,k}, R^{\sim}_k \subseteq R_{-,k}, (R_j \not\subset R_k + R'_k, \forall R_j \in R_{\min} : 1 \leq j < k)^{(*)}, R' \neq \varnothing\}.$$

Proposition 3 (The unique representation of sets in FS $(Y)_{X, \supseteq Z_0}$) $\forall$ $X, Y, Z_0 \subseteq \mathcal{A}$: $Y \neq \varnothing$, $Y \cap X = \varnothing$, $Z_0 \cap X = \varnothing$, $Z_0 \subseteq Y$, *then*

(a) *The elements of* $\text{FS}^*(Y)_{X, \supseteq Z_0}$ *are generated distinctly.*
(b) $\text{FS}(Y)_{X, \supseteq Z_0} = \text{FS}^*(Y)_{X, \supseteq Z_0}$.
(c) $\text{FS}^*(Y)_{X, \supseteq Z_0} \neq \varnothing$. *(H₂)*

Remark 2 See in "Appendix".

For special values of Y, X and Z_0 in $\text{FS}(Y)_{X, \supseteq Z_0}$, we obtain the structures of $\text{FS}_{C_0 \subseteq L_{C1}}$ and $\text{FS}(S \backslash L')_{L', R^*_0 \subseteq R^*_1}$ as they are presented in the following section.

3.2.2 Structure and unique representation of sets in $\text{FS}_{C_0 \subseteq L_{C1}}$ *and* $\text{FS}(S \backslash L')_{L', R^*_0 \subseteq R^*_1}$

Assume that the class $(L, S) \in \text{NFCS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$, $S \in \text{FCS}_{\supseteq S^*_0}(s^*_0, s^*_1)$, $L_{C_1} \in \text{FCS}_{C_0 \subseteq C_1}(s'_0, s'_1)$ and $L' \in \text{FS}_{C_0 \subseteq L_{C1}}$, since $L_{C_1} \in \text{FCS}_{C_0 \subseteq C_1}(s'_0, s'_1)$, we have $G_{C_1}(L) \neq \varnothing$ and $\exists L_i \in G(L)$ so that $\varnothing \subset L_i \subseteq L_{C_1}$ and $L_{C_1} \neq \varnothing$.

Lemma 1 $\forall$ $L, C_1 \subseteq \mathcal{A}$, *if* $L_{C_1} \equiv L \cap C_1 \neq \varnothing$ *and* $G_{C_1}(L) \neq \varnothing$, *then* $G(L_{C_1}) = G_{C_1}(L)$.

Proof ."$\Leftarrow$": $\forall L_k \in G_{C_1}(L)$: $L_k \in G(L)$, $L_k \subseteq C_1$, then $L_k \subseteq L_{C_1} \subseteq L$, $L = h(L_k) = h(L_{C_1})$, thus $L_k \in G(L_{C_1})$.
."$\Rightarrow$": Otherwise, $\forall L_k \in G(L_{C_1})$, then $L_k \subseteq L_{C_1} \subseteq C_1$, $h(L_k) = h(L_{C_1})$, since $G_{C_1}(L) \neq \varnothing$, so $\exists L_i \in G(L)$: $L_i \subseteq C_1$, $L_i \subseteq L_{C_1} \subseteq L$ and $L = h(L_i) = h(L_{C_1}) = h(L_k)$, i.e., $L_k \in G(L)$ and $L_k \subseteq C_1$ or $L_k \in G_{C_1}(L)$. □

(a) When $Y \equiv L_{C_1}$, $X \equiv \varnothing$ and $Z_0 \equiv C_0$. As $L_{C_1} \in \text{FCS}_{C_0 \subseteq C_1}(s'_0, s'_1)$, $C_0 \subseteq C_1$, then $L \supseteq C_0$, $C_0 \subseteq L_{C_1}$, and $L_{C_1} \neq \varnothing$. We have:

$$\text{FS}(L_{C_1})_{\varnothing, \supseteq C_0} = \{L' \supseteq C_0 | \varnothing \neq L' \subseteq L_{C_1}, h(L') = h(L_{C_1})\} \equiv \text{FS}_{C_0 \subseteq L_{C1}}.$$

Based on the representation of $R_{\min}$ in $\text{FS}^*(Y)_{X, \supseteq Z_0}$ and Lemma 1, we set $K_{\min} \equiv \text{Minimal}\ \{K_i \equiv L_i \backslash C_0,\ L_i \in G_{C_1}(\text{L})\}$, $K^i_U \equiv \bigcup_{K_k \in K_{\min}, k \leq i} K_k$, $K_{U,i} \equiv \begin{cases} K^{i-1}_U \backslash K_i, & if\ i \leq 1 \\ \varnothing, & if\ i = 0 \end{cases}$ and $K_{-,i} \equiv L_{C_1} \backslash (C_0 + K^i_U)$, and obtain the result as follows.

Fig. 2 *MFCS_FromLattice* algorithm

```
$FCS_{C_0 \subseteq C_1}\ s'_0, s'_1$ MFCS_FromLattice($\mathcal{LCG}^S$, $C_0$, $C_1$, $s'_0$, $s'_1$):
1. $FCS_{C_0 \subseteq C_1}\ s'_0, s'_1$ := $\varnothing$;
// test the necessary conditions for the non-emptyness of $FCS_{C_0 \subseteq C_1}\ s'_0, s'_1$ :
2. if (($s'_0 > s'_1$) or not($C_0 \subseteq C_1$) or (supp($C_0$) < $s'_0$) or ($s'_1$ < supp($C_1$)) then return $\varnothing$; //Corollary 1.a
3. for each ( (L, supp(L), $\mathcal{G}$(L) ) $\in \mathcal{LCG}^S$) do
4.    if ($s'_0 \leq$ supp(L) $\leq s'_1$ and L $\supseteq C_0$) then
5.       if ($\exists$L$_i \in \mathcal{G}$(L) and L$_i \subseteq C_1$) then {
6.          $L_{C_1}$ = L $\cap C_1$;
7.          $G_{C_1}(L)$ = {L$_i \in \mathcal{G}$(L) | L$_i \subseteq C_1$};
8.            $FCS_{C_0 \subseteq C_1}\ s'_0, s'_1$ := $FCS_{C_0 \subseteq C_1}\ s'_0, s'_1$ + ($L_{C_1}$, supp($L_{C_1}$)$\equiv$supp(L), $G_{C_1}(L)$);
9.       }
10.return $FCS_{C_0 \subseteq C_1}\ s'_0, s'_1$ ;
```

$$\mathrm{FS}^*_{C_0 \subseteq L_{C1}} \equiv \{L' \equiv C_0 + K_i + K'_i + K^{\sim}_i | K_i \in \mathcal{K}_{\min}, K'_i \subseteq K_{U,i}, K^{\sim}_i \subseteq K_{-,i} \text{ and } (K_k \not\subset K_i + K'_i, \forall K_k \in \mathcal{K}_{\min} : 1 \leq k < i)^{(**)}, L' \neq \varnothing\}. \quad (1)$$

Since $L_{C_1} \neq \varnothing$ and Proposition 3(c), we have $\mathrm{FS}^*_{C_0 \subseteq L_{C1}} \neq \varnothing$.

(b) When Y$\equiv$ $R^*_1 = S \backslash L'$, X $\equiv L'$, $Z_0 \equiv R^*_0 = R_0$, and assume that $S \backslash L' \neq \varnothing$. Since $S \in \mathrm{FCS}_{\supseteq S^*_0}(s^*_0, s^*_1)$ and $L' \in \mathrm{FS}_{C_0 \subseteq L_{C1}}$, then $S \supseteq S^*_0 \supseteq R_0$, $L' \subseteq L_{C_1} \subseteq C_1 \equiv \mathcal{A} \backslash R_0$ and $R_0 \subseteq S \backslash L'$. We have

$$\mathrm{FS}(S \backslash L')_{L', \supseteq R^*_0} \equiv \mathrm{FS}(S \backslash L')_{L', R^*_0 \subseteq R^*_1} = \{R' \supseteq R^*_0 | \varnothing \neq R' \subseteq R^*_1, h(L' + R') = S\}.$$

We assign $\mathcal{R}_{\min} \equiv$ Minimal $\{R_k \equiv S_k \backslash (L' + R^*_0), S_k \in \mathcal{G}(S)\}$, $R^k_U \equiv \bigcup_{R_j \in \mathcal{R}_{\min}, j \leq k} R_j$ and $R_{U,k} \equiv \begin{cases} R^{k-1}_U \backslash R_k, & if\ k \geq 1 \\ \varnothing, & if\ k = 0 \end{cases}$, where $R_k \in \mathcal{R}_{\min}$, and $R_{-,k} \equiv S \backslash (L' + R^*_0 + R^k_U)$, we derive the following result.

$$\mathrm{FS}^*(S \backslash L')_{L', R^*_0 \subseteq R^*_1} \equiv \{R' \equiv R^*_0 + R_k + R'_k + R^{\sim}_k | R_k \in \mathcal{R}_{\min}, R'_k \subseteq R_{U,k}, R^{\sim}_k \subseteq R_{-,k}, (R_j \not\subset R_k + R'_k, \forall R_j \in \mathcal{R}_{\min} : 1 \leq j < k)^{(*)}, R' \neq \varnothing\} \quad (2)$$

As $S \backslash$L$' \neq \varnothing$ and Proposition 3(c), we obtain $\mathrm{FS}^*(S \backslash L')_{L', R^*_0 \subseteq R^*_1} \neq \varnothing$.

The general procedure *MFS-ExtendedMinSC (mining frequent itemsets—Extended Minimum Single Constraints)*, shown in Fig. 3, distinctly generate the elements of $\mathrm{FS}^*(Y)_{X, \supseteq Z_0}$:

$\mathrm{FS}^*(Y)_{X, \supseteq Z_0}$ = *MFS-ExtendedMinSC*$(Y, X, Z_0, \mathcal{G}(X + Y))$.

When using Remark 2, we add lines 4–10 and 25 to the procedure.

In each equivalence class, two cases, $\mathrm{FS}^*_{C_0 \subseteq L_{C1}}$ = *MFS-ExtendedMinSC* $(L_{C1}, \varnothing, C_0, \mathcal{G}(L_{C1}))$ and $\mathrm{FS}^*(S \backslash L')_{L', R^*_0 \subseteq R^*_1}$ = *MFS-ExtendedMinSC*$(S \backslash L', L', R^*_0, \mathcal{G}(S))$, are to distinctly generate the left-hand sides L' and the right-hand sides R' of constrained rules, respectively. In addition, two cases, $\mathrm{FS}^*(L)$= *MFS-ExtendedMinSC*$(L, \varnothing, \varnothing, \mathcal{G}(L))$ and $\mathrm{FS}^*(S \backslash L')_{L'}$ = *MFS-ExtendedMinSC*$(S \backslash L', L', \varnothing, \mathcal{G}(S))$, are used to produce the sets $L' \in \mathrm{FS}^*(L)$ and $R' \in \mathrm{FS}^*(S \backslash L')_{L'}$ in each class $\mathcal{AR}(L, S)$ without the item constraints, as the post-processing algorithm *PP-MAR-MinSC-2* presented in Sect. 3.1.1.

According to Proposition 3, we have the following result.

Corollary 2 (Uniquely representing and distinctly generating two sides of rules in each equivalence class $\mathcal{AR}^+_{\supseteq L_0, \supseteq R_0}(L, S)$). *∀(L, S) ∈* $\mathrm{NFCS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$*, then*

(a) *The elements of* $\mathrm{FS}^*(S \backslash L')_{L', R^*_0 \subseteq R^*_1}$ *and* $\mathrm{FS}^*_{C0 \subseteq L_{C1}}$ *are generated distinctly.*
(b) $\mathrm{FS}(S \backslash L')_{L', R^*_0 \subseteq R^*_1} = \mathrm{FS}^*(S \backslash L')_{L', R^*_0 \subseteq R^*_1}$, $\mathrm{FS}_{C_0 \subseteq L_{C1}} = \mathrm{FS}^*_{C0 \subseteq L_{C1}}$.
(c) $\mathrm{FS}^*_{C_0 \subseteq L_{C1}} \neq \varnothing$
(d) ∀L$'$ ∈ $\mathrm{FS}^*_{C_0 \subseteq L_{C1}}$*, then* $\mathrm{FS}^*(S \backslash L')_{L', R^*_0 \subseteq R^*_1} \neq \varnothing \Leftrightarrow S/L' \neq \varnothing$.

3.2.3 Structure and unique representation of rules in $\mathcal{AR}^+_{\supseteq L_0, \supseteq R_0}(L, S)$

For $\forall(L, S) \in \mathrm{NFCS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$, let us denote

$$\mathcal{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S) \equiv \{r : L' \to R' | L' \in \mathrm{FS}^*_{C_0 \subseteq L_{C1}}, R' \in \mathrm{FS}^*(S \backslash L')_{L', R^*_0 \subseteq R^*_1}\}$$
and $\mathrm{Suff\_FS}^*_{C_0 \subseteq L_{C1}}(S) \equiv \{L' \in \mathrm{FS}^*_{C_0 \subseteq L_{C1}} | S \backslash L' \neq \varnothing\}$.

From Corollary 2, we have the following corollary.

Corollary 3 (Necessary and sufficient conditions for the non-emptiness of $\mathcal{AR}^+_{\supseteq L_0, \supseteq R_0}(L, S$ and its representation). *For* $\forall(L, S) \in \mathrm{NFCS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$*, then*

Fig. 3 *MFS-ExtendedMinSC* algorithm

```
$FS^*(Y)_{X,\supseteq Z_0}$ MFS-ExtendedMinSC($Y, X, Z_0, G(X+Y)$)
1. $FS^*(Y)_{X,\supseteq Z_0} = \varnothing$;
2. $R_{min}$ = Minimal{$R_k = S_k \backslash (X+Z_0) \mid S_k \in G(X+Y)$};        // Lemma 1 a, b
3. if ($R_{min}$ = {$\varnothing$}) then {                                         // remark 2.a
4.     if ($Z_0 = \varnothing$) then
5.         for each (R" | $\varnothing \subset$ R" $\subseteq$ Y) do
6.                 $FS^*(Y)_{X,\supseteq Z_0} = FS^*(Y)_{X,\supseteq Z_0}$ + {R"};
7.     else for each (R" | $Z_0 \subseteq$ R" $\subseteq$ Y) do
8.                 $FS^*(Y)_{X,\supseteq Z_0} = FS^*(Y)_{X,\supseteq Z_0}$ + {R"};
9. }
10. else
11. {  $R_0 = \varnothing$; $R_{U,0} = \varnothing$; $R_{-,0} = Y \backslash Z_0$;                      // remark 2.c
12.  for (k=1; $R_k \in R_{min}$; k++) do {                                   // Lemma 1 a, b
13.      $R_{U,k} = (R_{U,k-1} + R_{k-1}) \backslash R_k$;
14.      for each ($R'_k \subseteq R_{U,k}$) do {
15.          IsDuplicate = false;
16.          for (j=1; $R_j \in R_{min}$, j < k; j++) do
17.              if ($R_j \subset R_k + R'_k$) then {
18.              IsDuplicate = true; break;
19.          }
29.      }
21.      if (not(IsDuplicate)) then {
22.          $R_{-,k} = R_{-,k-1} \backslash R_k$;
23.          for each ($R_{\tilde{k}} \subseteq R_{-,k}$) do // khi $R_{min} \neq \{\varnothing\}$, ta biết chắc chắn $Z_0 + R_k + R'_k + R_{\tilde{k}} \neq \varnothing$
24.              $FS^*(Y)_{X,\supseteq Z_0} = FS^*(Y)_{X,\supseteq Z_0}$ + {$Z_0 + R_k + R'_k + R_{\tilde{k}}$};
25.       }
26.    }
27.}
28. return $FS^*(Y)_{X,\supseteq Z_0}$;
```

(a) *The rules of* $\mathrm{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S)$ *are generated distinctly.*
(b) $\mathrm{AR}^+_{\supseteq L_0, \supseteq R_0}(L, S) = \mathrm{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S)$.
(c) $\mathrm{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S) \neq \varnothing \Leftrightarrow \mathrm{Suff\_FS}^*_{C_0 \subseteq L_{C1}}(S) \neq \varnothing$.
(d) $\mathrm{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S) = \sum_{L' \in \mathrm{Suff\_FS}^*_{C_0 \subseteq L_{C1}}(S)} \{r : L'R' : R' \in \mathrm{FS}^*(S \backslash L')_{L', R_0^* \subseteq R_1^*}\}$.

Remark 3 (a) The result (1) is an improvement from our previous one (see Theorem $2^{(+)}$ in [17]). Since $K_{U,i}$ (that is used to check the condition $K'_i \subseteq K_{U,i}$) is smaller than $K_{U,C_0,C_1,i}$ and $K_{-,i}$ is larger than K_{-,C_0,C_1} in $^{(+)}$, checking the condition $^{(**)}$ in (1) is simpler.
(b) When considering $\mathrm{FS}^*(S \backslash L')_{L', R_0^* \subseteq R_1^*}$, if $S \equiv L \in G(L)$, then $\exists ! L' \equiv \in [L]$ and $S \backslash L' = L \backslash L' = \varnothing$. Thus, $\mathrm{FS}^*(L \backslash L')_{L', R_0^* \subseteq R_1^*} = \varnothing$ and $\mathrm{AR}^*_{\supseteq L_0, \supseteq R_0}(L, L) = \varnothing$. Hence, if $L \equiv S$, we always assume that $L \notin G(L)$.

The algorithm *MAR-MinSC-OneClass*, shown in Fig. 4, distinctly generates all constrained rules in each class $\mathrm{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S)$ based on each pair $(L, S) \in \mathrm{NFCS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$.

Example 3 Illustrating the advantages of the algorithm MAR-MinSC-OneClass for distinctly generating rules in each equivalence class $\mathrm{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S)$).

```
$AR^*_{\supseteq L_0, \supseteq R_0}(L, S)$ MAR-MinSC-OneClass ($C_0$, $L_{C1}$, $G(L_{C1})$, $R_0^*$, S, G(S), S )
1. $AR^*_{\supseteq L_0, \supseteq R_0}$  L, S  = $\varnothing$;
2. $FS^*_{C_0 \subseteq L_{C1}}$ = MFS-ExtendedMinSC($L_{C1}$, $\varnothing$, $C_0$, $G(L_{C1})$);
3. for each (L' $\in FS^*_{C_0 \subseteq L_{C1}}$ and S\L' $\neq \varnothing$) do {      // Corollary 3.c
4.         $R_1^*$ = S\L';
5.         $FS^*(S \backslash L')_{L', R_0^* \subseteq R_1^*}$ = MFS-ExtendedMinSC(S\L', L', $R_0^*$, G(S));
6.         for each R' $\in (FS^*(S \backslash L')_{L', R_0^* \subseteq R_1^*})$  do
7.                 $AR^*_{\supseteq L_0, \supseteq R_0}(L, S) = AR^*_{\supseteq L_0, \supseteq R_0}(L, S)$ + {r:L'$\to$R'}; // Corollary 3.d

8. }
9. return $AR^*_{\supseteq L_0, \supseteq R_0}(L, S)$;
```

Fig. 4 *MAR-MinSC-OneClass* algorithm

(a) For the constraints L_0, R_0 in Example 1(b), we consider the rule class $(L, S) = (\mathrm{egh}, \mathrm{bcegh}) \in \mathcal{NFCS}(s_0, s_1, c_0, c_1)$, $\mathcal{G}(\mathrm{L}) = \{\mathrm{e}, \mathrm{g}\}$ and $\mathcal{G}(S) = \{\mathrm{be}, \mathrm{bg}, \mathrm{ce}, \mathrm{cg}, \mathrm{ch}\}$. We have $S_0^* = \mathrm{bh}$, $C_0 = h$, $C_1 = \mathrm{acdefgh}$, $S \supseteq S_0^*$, $s_0^* = 0.28$, $s_1^* = 0.5$, $s_0^* \leq \mathrm{supp}(S) = 2/7 \leq s_1^*$. Thus, $S\mathrm{FCS}_{\supseteq S_0^*}(s_0^*, s_1^*)$. On the other hand, since $L \supseteq C_0$ and $\exists \mathrm{e} \in \mathcal{G}(L)$ so that $e \subseteq C_1$, then $s'_0 = 0.32$, $s'_1 = 0.71$ and $s'_0 \leq \mathrm{supp}(L) = 4/7 \leq s'_1$. Therefore, $L_{C_1} = \mathrm{egh} \in \mathrm{FCS}_{C_0 \subseteq C_1}(s'_0, s'_1)$ and $G_{C_1}(L) = \{\mathrm{e}, \mathrm{g}\}$. First is to generate the left-hand sides of rules, $\mathrm{FS}^*_{C_0 \subseteq L_{C1}}$. We have the result, $K_{\min} = \{K_1 = e, K_2 = g\}$. For $K_1 = e$, since $K_U^1 = e$, $K_{U,1} = \varnothing$, and $K_{-,1} = g$,

Fig. 5 *MAR-MinSC* algorithm

```
$ARS_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$ MAR-MinSC($s_0, s_1, c_0, c_1, L_0, R_0, \mathcal{LCG}$)
1. $ARS_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1) = \emptyset$;
   // test the necessary conditions for the non-emptyness of $ARS_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$:
2. if ($s_0 > s_1$ or $c_0 > c_1$ or ($L_0 \cap R_0 \neq \emptyset$)) then return $\emptyset$;   //Corollary 1.a
3. $S_0^* = L_0 \cup R_0$; $C_0 = L_0$; $C_1 = \mathcal{A} \backslash R_0$ ; $R_0^* = R_0$;
4. $s_0^* \equiv \max(s_0; c_0.\text{supp}(C_1))$, $s_1^* \equiv \min(s_1; c_1.\text{supp}(L_0))$;
5. $FCS_{\supseteq S_0^*}\ s_0^*, s_1^*$ = MFCS_FromLattice($\mathcal{LCG}, S_0^*, \mathcal{A}, s_0^*, s_1^*$);
6. for each ((S, supp(S), $G(S)$) $\in FCS_{\supseteq S_0^*}\ s_0^*, s_1^*$ ) do {        //Theorem 2
7.     $s_0'$= supp(S)/$c_1$;$s_1'$ = min(1; supp(S)/$c_0$);
8.     $FCS_{C_0 \subseteq C_1}\ s_0', s_1'$ =MFCS_FromLattice($\mathcal{LCG}^S, C_0, C_1, s_0', s_1'$);
9.     for each (($L_{C1}$, supp($L_{C1}$), $G_{C_1}(L)$) $\in FCS_{C_0 \subseteq C_1}\ s_0', s_1'$ ) do {
10.            $AR^*_{\supseteq L_0,\supseteq R_0}(L, S)$ =MAR-MinSC-OneClass($C_0, L_{C1}, G_{C_1}(L), R_0^*, S, G(S), S$);
11.            $ARS_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1) = ARS_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1) + AR^*_{\supseteq L_0,\supseteq R_0}(L, S)$;
12.    }
13. }
14. return $ARS_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$;
```

we obtain the left-hand sides: $h + e + \emptyset + \emptyset$ and $h + e + \emptyset + g$. For $K_2 = g$, as $K_U^2 = \text{eg}$, $K_{U,2} = e$, and $K_{-,2} = \emptyset$, then, an additional left-hand side is derived, $h + g + \emptyset + \emptyset$. Note that, without checking the condition $K_1 \subset g + e$, *the left-hand side,* $h + g + e + \emptyset$, *making a duplicate will be generated*. Thus, $\text{FS}^*_{C_0 \subseteq L_{C1}} \equiv \{\text{he}, heg, hg\}$. Second is to generate the right-hand sides of rules based on (2) as follows. For $L' = \text{he}$, since $S \backslash L' = \text{bcg} \neq \emptyset \Leftrightarrow \text{FS}^*(S \backslash L')_{L', R_0^* \subseteq R_1^*} \neq \emptyset$, where $R_0^* = b$ and $R_1^* = \text{bcg}$, we have $R_{\min} = \{R_1 \equiv \emptyset\}$, $R_U^1 = R_{U,1} = \emptyset$ and $R_{-,1} = \{\text{bcegh}\} \backslash \{\text{he} + \text{b}\} = \text{cg}$. Therefore, $\text{FS}^*(S \backslash L')_{L', R_0^* \subseteq R_1^*} \equiv \{R' = R_0^* + R_i + R_i' + R'' |\ R_i \in R_{\min}, R_i' \subseteq R_{U,i}, R'' \subseteq R_{-,i}\} = \{\text{b, bc, bg, bcg}\}$ and the constrained rules are derived: $\{\text{he} \to \text{b, he} \to \text{bc, he} \to \text{bg, he} \to \text{bcg}\}$. Similarly, for $L' = \text{heg}$ and $L' = \text{gh}$, we obtain the rules $\{\text{heg} \to \text{b, heg} \to \text{bc}\}$ and $\{\text{hg} \to \text{b, hg} \to \text{bc, hg} \to \text{be, hg} \to \text{bce}\}$, respectively. Finally, using Corollary 3, the rules that satisfy minimum single constraints are found: $\text{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S) = \{\text{he} \to \text{b, he} \to \text{bc, he} \to \text{bg, he} \to \text{bcg, heg} \to \text{b, heg} \to \text{bc, hg} \to \text{b, hg} \to \text{bc, hg} \to \text{be, hg} \to \text{bce}\}$.

(b) For the constraints L_0, R_0 in Example 1(c), we consider the rule class ($L = \text{fh}$, $S = \text{efgh}$), where $\mathcal{G}(L) = \{f\}$, $\mathcal{G}(S) = \{\text{ef, fg}\}$, $C_0 = f$, $C_1 = \text{abcdefg}$ and $L_{C_1} = f$. Then, $\text{FS}^*_{C_0 \subseteq L_{C1}} = \{f\}$. For $L' = f$, then $\text{S} \backslash L' = \text{egh}$, $R_0^* = h$ and $R_1^* = \text{egh}$, we obtain $R_{\min} = \{R_1 = e, R_2 = g\}$. With $R_1 = e$, since $R_U^1 = e$, $R_{U,1} = \emptyset$ and $R_{-,1} = \{\text{efgh}\} \backslash \{\text{f} + \text{h} + \text{e}\} = \text{g}$, the following right-hand sides are derived: $\text{h} + \text{e} + \emptyset + \emptyset$ and $\text{h} + \text{e} + \emptyset + g$. For $R_2 = g$, as $R_U^2 = \text{eg}$, $R_{U,2} = e$ and $R_{-,1} = \{\text{efgh}\} \backslash \{\text{f} + \text{h} + \text{eg}\} = \emptyset$, an additional right-hand side is derived, $\text{h} + \text{g} + \emptyset + \emptyset$. *The subset* $h + g + e + \emptyset$ *is not generated due to* $K_1 \subset g + e$. Hence, $\text{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S) = \{\text{f} \to \text{he, f} \to \text{heg, f} \to \text{hg}\}$.

Table 1 Dataset characteristics

Dataset	#Items	#Records	Avg. length
Connect (Co)	129	67,557	43
Mushroom (M)	119	8,124	23
Pumsb (P)	7,117	49,046	74
Chess (Ch)	75	3,196	37
T10I4D100K (T10)	1,000	100.000	40

It is important to note that *MAR-MinSC-OneClass* completely and exactly discovers constrained rule set $\text{AR}^*_{\supseteq L_0, \supseteq R_0}$ (L, S) without generating any the redundancy of candidate rules as well as the duplication of the solution. Thus, *MAR-MinSC-OneClass* is much more efficient than *PP-MAR-MinSC-2* and *PP-MAR-MinSC-1*.

3.3 Completely and distinctly generating all association rules in $\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$

From Theorem 1 and Proposition 3, we obtain the Theorem 2 below.

Theorem 2 (Generating all rules of $\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)$ completely and distinctly) *Assume that* (H$_1$) *is satisfied, we derive the result*

$$\text{ARS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1) = \sum\nolimits_{(L,S) \in \text{NFCS}_{\supseteq L_0, \supseteq R_0}(s_0, s_1, c_0, c_1)} \text{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S),$$

where $\text{AR}^*_{\supseteq L_0, \supseteq R_0}(L, S) = \sum\nolimits_{L' Suff\_\text{FS}^*_{C_0 \subseteq L_{C1}}(S)} \{r : L' \to R' : R' \in \text{FS}^*(S \backslash L')_{L', R_0^* R_1^*}\}$.

Then, we design an algorithm, named *MAR-MinSC*, shown in Fig. 5, to distinctly generate all rules in $\text{ARS}_{\supseteq L_0, \supseteq R_0}$ (s_0, s_1, c_0, c_1) without producing any candidate.

4 Experimental results

We implemented the *PP-MAR-MinSC-1, PP-MAR-MinSC-2* and *MAR-MinSC* algorithms in C# on Windows platforms. Experiments were performed on a PC with an i5-2400 CPU, 3.10 GHz@ 3.09 GHz PC and 3.16 GB of main memory. The source code for *Charm-L*, *MinimalGenerators* and *dEclat* algorithms [6] was converted to C#. *Charm-L* and *MinimalGenerators* were used to mine the lattice of the closed itemsets and their generators. *dEclat* was used to exploit all frequent itemsets. To evaluate the proposed algorithm, we compare its performance to those of *PP-MAR-MinSC-2* and *PP-MAR-MinSC-1* algorithms. *PP-MAR-MinSC-1* includes three phases: (1) executing *dEclat* to mine frequent itemsets; (2) integrating the constraints satisfying the monotone and anti-monotone properties into *Gen-Rules* [26] (using *Apriori* principle [1]) to generate candidate rules; (3) filtering the ones satisfying the remained constraints in a post-processing step. *PP-MAR-MinSC*-2 also includes three phases as *PP-MAR-MinSC-1*. But, in phase 2, *PP-MAR-MinSC-2* uses *MFS-ExtendedMinSC*(L, $\varnothing$, $\varnothing$, $G(L)$) and *MFS-ExtendedMinSC*($S\backslash L'$, L', $\varnothing$, $G(S)$) to generate the left-hand and right-hand sides of rules, respectively, without the item constraints.

For the performance test, five benchmark datasets in FIMDR [15] were chosen. Connect, Mushroom, Pumsb and Chess are real and dense datasets, i.e., they produce many long frequent itemsets even for very high support values. T10I4D100K is synthetic and sparse. Table 1 shows their characteristics.

For each tested dataset, the thresholds of s_1 and c_1 are fixed at 0.9 and 0.95, respectively. With a pair of minimum confidence (MC) and minimum support (MS), the size of L_0 and R_0 constraints ranges from 4 to 22 % of $|\mathcal{A}^{\mathcal{F}}|$ (step 2 %). For a pair of L_0's size and R_0's size, we choose 10 values of (L_0, R_0) (each (L_0, R_0) comprises of items taken randomly from $\mathcal{A}^{\mathcal{F}}$). Let T-MAR-MinSC, T-PP-MAR-MinSC-2 and T-PP-MAR-MinSC-1 be the average execution times of *MAR-MinSC, PP-MAR-MinSC-2*, and *PP-MAR-MinSC-1* , respectively, for 100 constraint pairs (L_0, R_0). For each tested dataset together with two minimum thresholds of support and confidence, we execute three algorithms and find that their outputs are the same. The average running times of the algorithms are shown in Figs. 6, 7 and 8.

As can be seen from the figures, *MAR-MinSC* outperforms *PP-MAR-MinSC-2* and *PP-MAR-MinSC-1*, especially for the lower values of minimum support and confidence.

To explain the efficiency of *MAR-MinSC* in comparison with those of *PP-MAR-MinSC-2* and *PP-MAR-MinSC-1*, we also take into account the percent ratio of the number of redundant candidate rules (not satisfying the constraints) to the total of all generated rules after executing *PP-MAR-MinSC-2* and *PP-MAR-MinSC-1* on a given triple of dataset, minimum support and minimum confidence, called DS-MS-MC. The results are shown in Table 2.

Table 2 shows the average execution times of *MAR-MinSC*, *PP-MAR-MinSC-2*, and *PP-MAR-MinSC-1*, where columns R-OvPP2 and R-OvPP1 represent the ratios of T-MAR-MinSC to T-PP-MAR-MinSC-2 and T-PP-MAR-MinSC-1, respectively, and columns RR-PP2 and RR-PP1 show the ratios of the number of redundant rules, which do not satisfy the constraints, to the total of generated rules when *PP-MAR-MinSC-2* and *PP-MAR-MinSC-1* are used, respectively. Compared to *PP-MAR-MinSC-2* and *PP-MAR-MinSC-1, MAR-MinSC* is faster for all tested datasets. The time is reduced by 48.23 to 17.77 % in column R-OvPP2 and by 6.86 to 0.12 % in column R-OvPP1. The reason for the reduction is that there are a large number of generated redundant rules (RR-PP2 and RR-PP1 range from 94.5 to 99.96 % and from 98.7 to 99.98 %, respectively) that fail the last test of *PP-MAR-MinSC-2* and *PP-MAR-MinSC-1*, leading to lower performance.

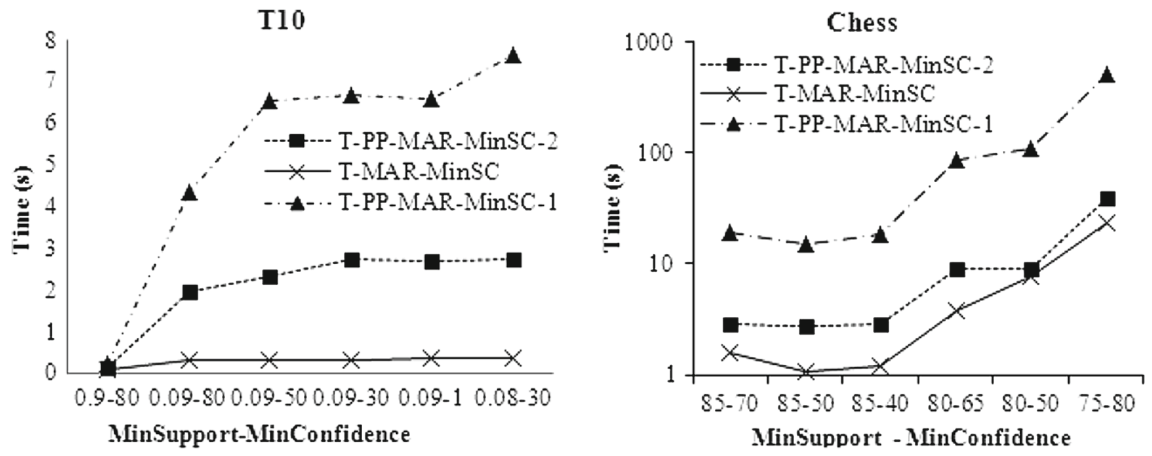
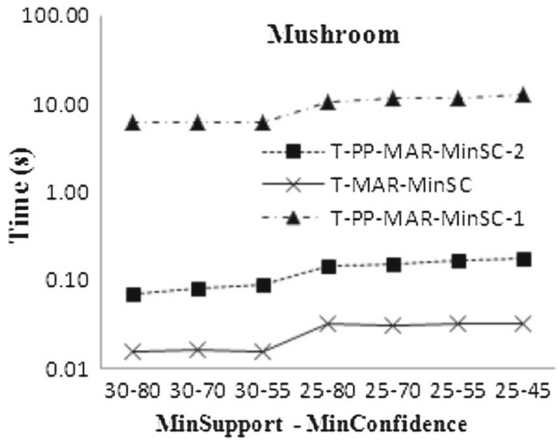


Fig. 6 Performance of *MAR-MinSC, PP-MAR-MinSC-2* and *PP-MAR-MinSC-1* on Chess and T10I4D100K



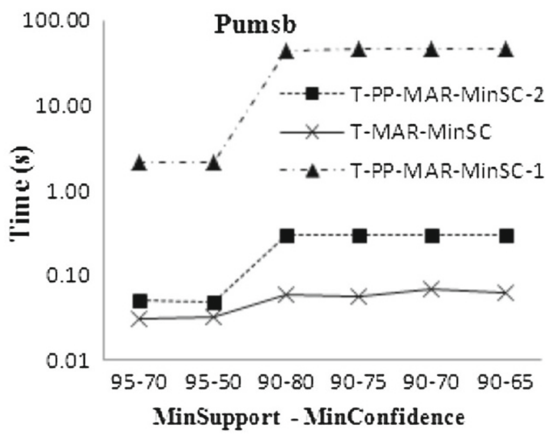


Fig. 7 Performance of *MAR-MinSC, PP-MAR-MinSC-1* and *PP-MAR-MinSC-2* on Mushroom and Pumsb

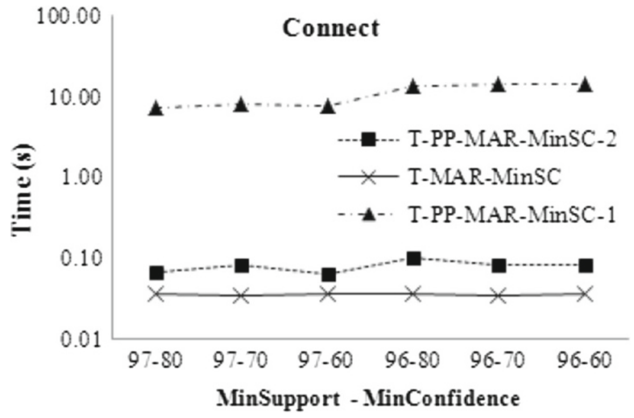


Fig. 8 Performance of *MAR-MinSC, PP-MAR-MinSC-1* and *PP-MAR-MinSC-2* on Connect

Table 2 The time reduction of *MAR-MinSC*

DB-MS-MC	R-OvPP2 (%)	R-OvPP1 (%)	RR-PP2 (%)	RR-PP1 (%)
Co-97-80	45.52	0.47	*94.5*	*98.7*
Co-96-60	43.28	0.26	99.8	99.9
Ch-85-50	33.62	*6.86*	99.92	99.95
Ch-80-35	20.38	4.72	99.91	99.944
P-95-80	*48.23*	1.53	98.57	99.62
P-90-65	21.11	*0.12*	99.89	99.96
M-30-70	20.41	0.27	*99.96*	99.98
M-25-45	*17.77*	0.29	99.95	*99.98*

5 Conclusions and future works

The paper proposed a new, efficient algorithm called *MAR-MinSC* for mining association rules with minimum single constraints. It uses the lattice of closed itemsets and their generators as the input data. *MAR-MinSC* neither leads to the redundancy of generated rules nor directly checks rules with the constraints. Thus, it obtains the high performance. Our method that is based on the lattice and suitable equivalence relations reveals many significant implications in theory and practice.

In theory, the method demonstrates the explicit structure and unique representation of the solution set based on essentially basic factors such as closed itemset, support and generator. Therefore, this is the representation without losing information. The correctness of the theoretical results was proven.

In practice, the method is a basis to design parallel algorithms for efficiently mining the solution set in distributed environments. Moreover, the efficiency of the algorithm is minimally affected by the frequent change of the constraints in online systems.

In the future, we will use this method to study problems with other extended constraints.

Appendix: Proofs and Remarks

Proposition 4

Proof Since $\text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1) \subseteq \text{ARS}(s_0, s_1, c_0, c_1)$, we obtain $\text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1) = \sum_{(L,S)\text{NFCS}(s_0,s_1,c_0,c_1)} \text{AR}_{\supseteq L_0,\supseteq R_0}(L, S)$, where $\text{AR}_{\supseteq L_0,\supseteq R_0}(L, S) = \text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1) \cap \text{AR}(L, S) = \{r : L' \to R' \in \text{AR}(L, S) \mid L' \supseteq L_0,\ R' \supseteq R_0\}$. Moreover, with $S' \equiv L'+R'$, since $(L, S) \in \text{NFCS}(s_0, s_1, c_0, c_1)$, so $s_0 \leq \text{supp}(S')=\text{supp}(S)\leq s_1$, $c_0 \leq \text{supp}(S')/\text{supp}(L')=\text{supp}(S)/\text{supp}(L)\leq c_1$.

Proposition 5

Proof (a) Since $r : L' \to R' \in \text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$, we have $L', R' \neq \varnothing$, $L' \cap R' = \varnothing$, $S' \equiv L'+R'$, $L_0 \subseteq L'$, $R_0 \subseteq R'$. Let $L \equiv h(L')$, $S \equiv h(L'+R')$, since $L' \neq \varnothing$, so $\varnothing \neq L \subseteq S$ (if $L = \varnothing$, then $\varnothing \subset L' \subseteq h(L') \subseteq h(L) = \varnothing$!), supp($S$) = supp($S'$) $\in [s_0, s_1]$, supp(S)/supp(L) = supp(S')/supp(L') = $c(r) \in [c_0, c_1]$. Thus, $(L, S) \in \text{NFCS}(s_0, s_1, c_0, c_1)$ and $r \in \text{AR}_{\supseteq L_0,\supseteq R_0}(L, S)$.

Due to $L_0 \subseteq L'$, $R_0 \subseteq R'$, $L' \cap R'=\varnothing$, so $L_0 \cap R_0 = \varnothing$, $S_0^* \subseteq S' \subseteq \mathcal{A}$, $L' = S' \setminus R' \subseteq \mathcal{A}\backslash R_0$, $L' \subseteq L$, therefore, $L_0 \subseteq L' \subseteq L\backslash R_0 = L_{C_1} \subseteq C_1$, supp($C_1$) $\leq$ supp(L')$\leq$ supp(L_0), supp(C_1).$c_0 \leq$ supp(L').$c_0 \leq$ supp(S') $\leq$ supp(L').$c_1 \leq$ supp(L_0).c_1 và $s_0^* \leq$ supp(S') = supp(S) $\leq s_1^*$. Thus, $s_0^* \leq s_1^*$ and supp(S_0^*) $\leq$ supp(S') $\leq s_0^*$, supp($\mathcal{A}$) $\leq$ supp(S') $\leq s_1^*$.

(b) With $\forall(L,S) \in \text{NFCS}(s_0, s_1, c_0, c_1)$, $\forall\ r : L' \to R' \in \text{AR}_{\supseteq L_0,\supseteq R_0}(L, S)$, we have $\varnothing \neq L \subseteq S$, $L', R' \neq \varnothing$, $L' \cap R' = \varnothing$, $h(L') = L$, $h(S') = S$, $s_0 \leq$ supp(S') = supp(S) $\leq s_1$, $c_0 \leq$ supp(S)/supp(L) $\leq c_1$, $R_0 \subseteq R'$, $L_0 \subseteq L'$. Since $L' = S' \setminus R' \subseteq \mathcal{A}\backslash R_0 = C_1$, $L_0 \subseteq L' \subseteq L_{C_1} \subseteq L$, $L = h(L') = h(L_{C_1})$, so $L' \in \text{FS}_{C_0 \subseteq L_{C1}}$. On the other hand, since supp(S)/$c_1 \leq$ supp(L') = supp(L) $\leq$ supp(S)/c_0, $s'_0 \leq$ supp(L') $\leq s'_1$, $S \supseteq S' \equiv L'+R' \supseteq S_0^*$, supp($C_1$).$c_0 \leq$ supp(L').$c_0 \leq$ supp(S') = supp(S) $\leq$ supp(L').$c_1 \leq$ supp(L_0).c_1, so $s_0^* \leq$ supp(S) $\leq s_1^*$. Due to $\mathcal{G}(L')\neq \varnothing$ [5], we take $L_i \in \mathcal{G}(L') \subseteq \mathcal{G}(L)$, then $L_i \subseteq L' \subseteq C_1$, thus $\mathcal{G}_{C_1}(L) \neq \varnothing$, $L_{C_1} \in \text{FCS}_{C_0 \subseteq C_1} \subseteq (s'_0, s'_1)$, $S \in \text{FCS}_{S_0^* \subseteq S_1^*}(s_0^*, s_1^*)$ or $(L, S) \in \text{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$. Since $R_0^* \equiv R_0 \subseteq R'=S' \setminus L' \subseteq S \setminus L'=R_1^*$, so $R' \in \text{FS}(S\backslash L')_{L', R_0^* \subseteq R_1^*}$, $r \in \text{AR}^+_{\supseteq L_0,\supseteq R_0}(L, S)$ and $\text{AR}_{\supseteq L_0,\supseteq R_0}(L, S) \subseteq \text{AR}^+_{\supseteq L_0,\supseteq R_0}(L, S)$. Finally, due to $c_0 \leq c(r)$ = supp(S')/supp(L') = supp(S)/supp(L)$\leq c_1$, so r $\in \text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$. Hence, $\text{AR}_{\supseteq L_0,\supseteq R_0}(L, S) \subseteq \text{ARS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$.

(c) As $(L, S) \in NFCS_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1)$, then $L_{C_1} \in \text{FCS}_{C_0 \subseteq C_1} s'_0, s'_1$, $L \supseteq L_0$ and $\exists L_i \in \mathcal{G}(L)$: $L_i \subseteq C_1 \equiv \mathcal{A}\backslash R_0$. Set $L' \equiv L_i \cup L_0$, we have $\varnothing \subset L_i \subseteq L'$. Thus, we obtain $L_i \subseteq L_{C_1}$. Due to $L_0 \cap R_0 = \varnothing$, then $L_0 \subseteq C_1$ and $L_0 \subseteq L_{C_1}$. Therefore, $C_0 \equiv L_0 \subseteq L' \subseteq L_{C_1}$, $L_i \subseteq L' \subseteq L_{C_1} \subseteq L$, $L = h(L_i)$=$h(L')$= $h(L_{C_1})$. Hence, $L' \in \text{FS}_{C_0 \subseteq L_{C1}} \neq \varnothing$.

Since $(L, S) \in \text{NFCS}_{\supseteq L_0,\supseteq R_0}(s_0, s_1, c_0, c_1) \subseteq \text{NFCS}(s_0, s_1, c_0, c_1)$ and b., we have $\text{AR}_{\supseteq L_0,\supseteq R_0}(L, S) \subseteq \text{AR}^+_{\supseteq L_0,\supseteq R_0}(L, S)$. Therefore, we only need to prove the reverse dimension, "$\supseteq$". With $\forall$r: $L' \to R' \in \text{AR}^+_{\supseteq L_0,\supseteq R_0}(L, S)$, $L', R' \neq \varnothing$, $L' \in \text{FS}_{C_0 \subseteq L_{C1}}$, $R' \in \text{FS}(S\backslash L')_{L', R_0^* \subseteq R_1^*}$, we have $L_0 \subseteq L' \subseteq L_{C1} \subseteq C_1 \subseteq L_1$, $h(L') = h(L_{C1})$, $R_0 \subseteq R' \subseteq R_1^* = S\backslash L'$, $L' \cap R' = \varnothing$, $h(L'+R')$=S. As $L_{C_1} \in \text{FCS}_{C_0 \subseteq C_1}(s'_0, s'_1)$, so $\exists L_i \in \mathcal{G}(L)$: $L_i \subseteq C_1$, thus, $L_i \subseteq L_{C1} \subseteq L$, $L = h(L_i) = h(L_{C1}) = h(L')$. Therefore, r $\in \text{AR}_{\supseteq L_0,\supseteq R_0}(L, S)$ and $\text{AR}^+_{\supseteq L_0,\supseteq R_0}(L, S) \subseteq \text{AR}_{\supseteq L_0,\supseteq R_0}(L, S)$. Hence, $\text{AR}^+_{\supseteq L_0,\supseteq R_0}(L, S) = \text{AR}_{\supseteq L_0,\supseteq R_0}(L, S)$. □

Proposition 6

Proof (a) Assume that there exist two duplicate subsets R'^1, R'^2 in $\text{FS}^*(Y)_{X, Z_0 \subseteq Z_1}$, i.e., $\exists$k2>k1$\geq$ 1, $R'^j \equiv \mathbf{Z_0}+R_{kj}+R'_{kj}+R^\sim_{kj}$, $R_{kj} \in R_{\min}$, $R'_{kj} \subseteq R_{U,kj}$, $R^\sim_{kj} \subseteq R_{-,kj}$. , $\forall$j =1,2, such that $R'^1. = R'^2$. Thus, $R_{k1} \subseteq R_{k1} + R'_{k1} + R^\sim_{k1} = R_{k2} + R'_{k2} + R^\sim_{k2}$. Since $R_{k1} \cap R^\sim_{k2} \subseteq R_{k1} \cap R_{-,k2} \subseteq R_{k1} \cap R_{-,k1}= \varnothing$ and R_{k1}, R_{k2} are two different minimal sets in $R_{\min}$, so $R_{k1} \subset R_{k2} + R'_{k2}$: it contradicts with the method for selecting R'_{k2} in $^{(*)}$.

(b) . "$\subseteq$": For any $R' \in \text{FS}(Y)_{X,\supseteq Z_0} \neq \varnothing$, we have $Z_0 \subseteq R' \subseteq$ Y, $R' \neq \varnothing$, $S' \equiv$ X + R', $h(S') = h(X + Y)$. Since Y$\cap$X = $\varnothing$, so R' $\cap$X = $\varnothing$, X+Z_0 $\subseteq S' \subseteq$ X+Y. Since $S' \neq \varnothing$, let $S_k \in G(S') \subseteq G$(X+Y) (see[5]), $S_k \subseteq S'$, so $R_k \equiv S_k\backslash$(X+Z_0) $\subseteq S'$ $\backslash$(X+Z_0)=R' $\backslash Z_0 \subseteq R'$. Set B$\equiv\{R_i \equiv S_i\backslash$(X+$Z_0$): $S_i \in G(S')\}$, C $\equiv\{R_i \equiv S_i\backslash$(X+$Z_0$): $S_i \in G(X + Y)\}$, then $R_k \in B$. Since B and C are finite sets and $\varnothing \neq B \subseteq C$, there exist the minimal sets $R_{\min,S'} \equiv$ Minimal(B)$\neq \varnothing$, $R_{\min} \equiv$Minimal(C)$\neq \varnothing$. We always have the lowest index k of sets R_i in $R_{\min,S'} \equiv$ Minimal(B). Assume that $R_k \notin R_{\min}$, as $R_k \in R_{\min,S'}$, $R_k \in C$, so $\exists R_j \in R_{\min}$: $R_j \subset R_k$, where $R_j \equiv S_j\backslash(X + Z_0)$, $S_j \in G(X + Y)$ and $h(S_j) = h(X + Y)$, $S_j \subseteq (X + Z_0) \cup S_j = (X + Z_0) + R_j \subseteq (X + Z_0) + R_k \subseteq X + R'$=$S' \subseteq X + Y$, $h(X + Y) = h(S_j)$=$h(S')$. Then, $S_j \in G(S')$, $R_j \in B \cap R_{\min}$. Thus $R_j \in R_{\min,S'}$ and $R_j \subset R_k \in R_{\min,S'}$: it is contradictory because R_k is a minimal set in B, Hence, $R_k \in R_{\min} \neq \varnothing$.

Then, it is realized that, if $R_{\min} = \varnothing$ then $\text{FS}(Y)_{X,\supseteq Z_0} = \varnothing \neq \text{FS}^*(Y)_{X,\supseteq Z_0}$.

. We have $S' = S_k + S''_k$, where $S''_k \equiv S'$ $\backslash S_k$. Since $S' \supseteq X + Z_0$, so $S' = X + Z_0 + R_k + R'_k + R^\sim_k. = X + R'$, where $R' \equiv Z_0 + R_k + R'_k + R^\sim_k$, $R_k \equiv S_k\backslash(X + Z_0) \in R_{\min}$, $R'_k \equiv [S''_k\backslash(X + Z_0)] \cap R^k_U$=$[S'$ $\backslash(X + Z_0)\backslash S_k] \cap R^{k-1}_U \subseteq R^{k-1}_U\backslash S_k \subseteq R^{k-1}_U\backslash R_k \equiv R_{U,k}$ (since $R_k \cap$ $[S'$ $\backslash(X + Z_0)\backslash S_k] \subseteq R_k\backslash S_k$=$\varnothing$), $R^\sim_k \equiv [S''_k\backslash(X + Z_0)]\backslash R^k_U \subseteq (S'$ $\backslash$ $X)\backslash(Z_0 + R^k_U) \subseteq Y$ $\backslash(Z_0 + R^k_U) \equiv R_{-,k}$. Assume that $\exists R_j \equiv S_j\backslash(X + Z_0) \in R_{\min}$: $1 \leq j < k$ and $R_j \subset R_k + R'_k$, then $h(S_j) = h(X + Y)$, $S_j \subseteq (X + Z_0) \cup S_j = (X + Z_0) + R_j \subseteq (X + Z_0) + R_k + R'_k \subseteq X + R' \equiv S' \subseteq X + Y$, $h(X + Y) = h(S_j) = h(S')$. Thus, $S_j \in G(S')$ and $R_j \in R_{\min,S'}$ but $j < k$: it is contradictory to the method for selecting the index k. Hence, $R' \in \text{FS}^*(Y)_{X,\supseteq Z_0}$.

. "$\supseteq$": For any $R' \in \text{FS}^*(Y)_{X,\supseteq Z_0}$, we have $R' = Z_0 + R_k + R'_k + R^\sim_k$, where $R_k \equiv S_k\backslash(X + Z_0) \in R_{\min}$ and $S_k \in G(X + Y)$, $h(S_k) = h(X + Y)$, $R' \neq \varnothing$, $R_k \subseteq (X+Y)\backslash(X+Z_0) = Y\backslash Z_0 \subseteq Y$. Moreover, $R'_k \subseteq R_{U,k} \subseteq Y$, $R^\sim_k \subseteq R_{-,k} \subseteq Y$, $Z_0 \subseteq R' \subseteq Y$ and $R' \cap X = \varnothing$. On the other hand, since $X + Y \supseteq X + R' \supseteq X + Z_0 + R_k$= $(X + Z_0) \cup S_k \supseteq S_k$, so $h(S_k) = h(X + R')$=$h(X + Y)$. Therefore, $R' \in \text{FS}(Y)_{X,\supseteq Z_0}$.

(c) Since $Y \cap X = \varnothing$ and $Z_0 \subseteq Y$, so $Z_0 \cup X = \varnothing$. Set $R^* \equiv \mathbf{Y}$ and $\forall R_k \equiv S_k \backslash (X + Z_0) \in R_{\min} \neq \varnothing$: $S_k \in G(X + Y)$, $R_k \subseteq (X+Y) \backslash (X+Z_0) = Y \backslash Z_0 \subseteq R^*$, then $R^* \neq \varnothing$ and $Z_0 \subseteq R* \subseteq Y$, $S_k \subseteq S_k \cup (X + Z_0) = (X + Z_0) + R_k \subseteq X + R* = X + Y$. Therefore, $h(X + Y) = h(S_k) = h(X + R*)$. Hence, $\exists\, R^* \in \mathrm{FS}(Y)_{X,\supseteq Z_0} = \mathrm{FS}^*(Y)_{X,\supseteq Z_0} \neq \varnothing$. □

Remark 4. From the proof of Proposition 3, we have remark as follows.

(a) For $\forall\, R' \in \mathrm{FS}^*(Y)_{X,\supseteq Z_0}$ with the representation $R' \equiv Z_0 + R_k + R'_k + R^{\sim}_k$. If $\exists R' = \varnothing$, then $Z_0 = \varnothing$ and $\exists S_k \in G$ (X+Y) so that $R_k \equiv S_k \backslash (X + Z_0) = \varnothing$. Thus, $S_k \subseteq X \subseteq X + Y and h(X+Y) = h(X) = h(S_k)$. Moreover, $R_{\min} \equiv \{\, R_1 \equiv \varnothing\}$, $R_{U,1} = R^1_U = \varnothing$, $R_{-,1} \equiv Y \neq \varnothing$ and $R' = R^{\sim}_1 \subseteq R_{-,1}$. Therefore, when representing R'in $\mathrm{FS}^*(Y)_{X,\supseteq Z_0}$ based on the sets$R_k \in R_{\min}$, $R'$is only empty in the cases $Z_0 = \varnothing$, $h(X + Y) = h(X)$ and $R_{\min} = \{\varnothing\}$. Then, $R' \in \mathrm{FS}^*(Y)_{X,\supseteq Z_0} \Leftrightarrow \varnothing \subset R' \subseteq Y \neq \varnothing$. In practice, we consider three cases as follows.

· If $R_{\min} = \{\varnothing\}$ and $Z_0 = \varnothing$, then $R_{U,1} = R^1_U = \varnothing$, $R_{-,1} \equiv Y \neq \varnothing$ and $\mathrm{FS}^*(Y)_{X,\supseteq Z_0} \equiv \{R' | \varnothing \neq$ R' $\equiv R^{\sim}_1$, $R^{\sim}_1 \subseteq Y\}$ =$\{R' | \varnothing \subset R' \subseteq Y\}$.
· If $R_{\min}$=$\{\ \varnothing\}$ and $Z_0 \neq \varnothing$, then $R_{U,1} = R^1_U = \varnothing$, $R_{-,1} \equiv$ Y $\backslash Z_0$ and $\mathrm{FS}^*(Y)_{X,\supseteq Z_0} \equiv \{R' \mid R' \equiv Z_0 + R^{\sim}_1$, $R^{\sim}_1 \subseteq$Y$\backslash Z_0\}$=$\{R' \mid\ Z_0 \subseteq R' \subseteq$Y $\}$.
· If $R_{\min} \neq \{\ \varnothing\}$, then $\mathrm{FS}^*(Y)_{X,\supseteq Z_0} \equiv \{$R' $\equiv Z_0 + R_k + R'_k + R^{\sim}_k |\ \ R_k \in R_{\min}$, $R'_k \subseteq R_{U,k}$, $R^{\sim}_k \subseteq R_{-,k}$, $(R_j \not\subset R_k + R'_k, \forall R_j \in R_{\min}$: $1 \leq j < k)\}$.

For the last two cases, we do not need to check the obvious condition $R' \neq \varnothing$ when generating R'.

(b) (*The advantage of the condition* $^{(*)}$*in decreasing redundant candidates accompanying exponential reduction*). In the process of forming set R', which originated from $R^*_0 + R_k$, when finding growing subsets $R'_k \subseteq R_{U,k}$ and then $R^{\sim}_k \subseteq R_{-,k}$ to supplement R', if the condition $^{(*)}$ is violated, then we neither need to continue considering the supersets R'' (estimated$2^{|R_{U,k} \backslash R'_k|}$ supersets) of $R'_k (R'_k \subset R" \subseteq R_{U,k})$ nor add all subsets $R^{\sim}_k$ (estimated $2^{|R_{-,k}|}$ subsets) of $R_{-,k}$ to R' (i.e., there are $(2^{|R_{U,k} \backslash R'_k|} - 1)$*$(2^{|R_{-,k}|})$ subsets eliminated because all of them are redundant candidates for R'). Then, we go on considering other R''_k sets $(R'_k \not\subset R''_k \subseteq R_{U,k})$ or other R_k sets of $R_{\min}$. The necessary and sufficient condition $^{(*)}$for distinctly generating the right-hand side R' of rules helps to eliminate many redundant candidates for them. This condition also helps to completely eliminate the duplication of the solutions, and the checking process is only based on minimal sets or generators with small quantity and size. It makes an important contribution to explain the efficiency of the corresponding algorithms.

c. (*Improving the calculation of the border sets*). It is realized that, for each $k > 1$, to calculate $R^{k-1}_U = R^{k-2}_U \cup R_{k-1}$, $R_{U,k} \equiv R^{k-1}_U \backslash R_k$, $R_{-,k} \equiv R^* \backslash R^k_U$, where $R^* \equiv Y \backslash Z_0$, we must perform two subtractions and one union on the generators that cannot be disjoint. To decrease the calculation of the border sets $R_{U,k}$ and $R_{-,k}$, we note that

$$\cdot S_k,\ S'_k,\ R_k \in R_{\min},\ R'_k,\ R^{\sim}_k,\ R' \equiv R^*_0 + R_k + R'_k + R^{\sim}_k;$$
$$R^k_U,\ R_{U,k},\ R_{-,k},\ R^*$$
$$R_{U,k} = [(R^{k-2}_U \backslash R_{k-1}) + R_{k-1}] \backslash R_k = (R_{U,k-1} + R_{k-1}) \backslash R_k,\ R_{-,k} = R_{-,k-1} \backslash R_k, \forall k \geq 1 \quad \text{and}$$
$$R_0 \equiv R_{U,0} \equiv \varnothing,\ R_{-,0} \equiv R^*.$$

Thus, $R_{U,k} = \begin{cases} (R_{U,k-1} + R_{k-1}) \backslash R_k, if k \geq 1 \\ \varnothing, \quad if k = 0 \end{cases}$, $R_{-,k} = \begin{cases} R_{-,k-1} \backslash R_k, \quad if k \geq 1 \\ Y \backslash Z_0, \quad if k = 0 \end{cases}$ and $R_0 \equiv \varnothing$.
For each k≥1, there remains only a disjoint union $R_{U,k} = R_{U,k-1} + R_{k-1}$ on two small sets in size (thus, its calculation is faster than that of normal union $R^{k-1}_U = R^{k-2}_U \cup R_{k-1}$ on two sets that may not be disjoint and have large sizes), and two subtractions, where one $R_{-,k} = R_{-,k-1} \backslash R_k$ is performed on two sets $R_{-,k-1} \subseteq R^*$ and $R_k \subseteq R^k_U$ that their sizes are less than those of sets in the subtraction $R_{-,k} \equiv R^* \backslash R^k_U$.

References

1. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast discovery of association rules. In: Advances in Knowledge Discovery and Data Mining, pp. 307–328. AAAI Press, Menlo Park (1996)
2. Anh, T., Hai, D., Tin, T., Bac, L.: Efficient algorithms for mining frequent itemsets with constraint. In: Proceedings of the Third International Conference on Knowledge and Systems Engineering, pp. 19–25 (2011)
3. Anh, T., Hai, D., Tin, T., Bac, L.: Mining frequent itemsets with dualistic constraints. In: Proceedings of PRICAI 2012, LNAI, vol. 7458 , pp. 807–813. Springer, Berlin (2012)
4. Anh, T., Tin, T., Bac, L.: Structures of association rule set. Lecture Notes in Artificial Intelligence, vol. 7197, pp. 361–370. Springer, Berlin (2012)
5. Anh, T., Tin, T., Bac, L.: An approach for mining concurrently closed itemsets and generators. Advanced computational methods for knowledge engineering, SCI, vol. 479, pp. 355–366. Springer, Berlin (2013)
6. Anon: http://www.cs.rpi.edu/~zaki/wwwnew/pmwiki.php/Software/Software#patutils (2010). Accessed 2010
7. Bayardo Jr, R.J.: Efficiently mining long patterns from databases. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 85–93. ACM, New York (1998)

8. Bonchi, F., Giannotti, F., Mazzanti, A., Pedreschi, D.: Examiner: optimized level-wise frequent pattern mining with monotone constraints. In: Proceedings of IEEE ICDM'03, pp. 11–18 (2003)
9. Bonchi, F., Goethals, B.: FP-Bonsai: The Art of Growing and Pruning Small FP-Trees (2004)
10. Bonchi, F., Lucchese, C.: On closed constrained frequent pattern mining. In: Proceedings of IEEE ICDM'04 (2004)
11. Boulicaut, J.F., Bykowski, A.: Frequent closures as a concise representation for binary Data Mining. In: Proceedings of PAKDD'00, vol. 1805, pp. 62–73. Springer, Berlin (2000)
12. Boulicaut, J.F., Bykowski, A., Rigotti, C.: Free-sets: a condensed representation of boolean data for the approximation of frequency queries. Data Min. Knowl. Discov **7**, 5–22 (2003)
13. Burdick, D., Calimlim, M, Gehrke, J.: MAFIA: A maximal frequent itemset algorithm for transactional databases. In: Proceedings of IEEE ICDE'01, pp. 443–452 (2001)
14. Elena, B., Luca, C., Tania, C., Paolo, G.: Generalized association rule mining with constraints. Inf. Sci. Int J. 68–84 (2012)
15. FIMDR, Frequent Itemset Mining Dataset Repository: http://fimi.cs.helsinki.fi/data/ (2009). Accessed 2009
16. Hai, D., Tin, T., Bac, L.: An efficient algorithm for mining frequent itemsets with single constraint. In: Advanced Computational Methods for Knowledge Engineering, SCI, vol. 479, pp. 367–378. Springer, Berlin (2013)
17. Hai, D., Tin, T., Bay, V.: An efficient method for mining frequent itemsets with double constraints. Int. J. Eng. Appl. Artif. Intell. (EAAI) **27**, 148–154 (2014)
18. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: SIGMOD'00, pp. 1–12 (2000)
19. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: a frequent-pattern tree approach. Data Min. Knowl. Discov **8**(1), 53–87 (2004)
20. Jeudy, B., Boulicaut, J.F.: Optimization of association rule mining queries. Intell. Data Anal. **6**(4), 341–357 (2002)
21. Lee, A.J., Lin, W.C., Wang, C.S.: Mining association rule with multi-dimensional constraints. J. Syst. Softw. **79**, 79–92 (2006)
22. Lin, D.I., Kedem, Z.M.: Pincer search: an efficient algorithm for discovering the maximum frequent sets. IEEE Trans Knowl. Data Eng. **14**, 553–566 (2002)
23. Mannila, H., Toivonen, H.: Levelwise search and borders of theories in knowledge discovery. Data Min. Knowl. Discov. **1**, 241–258 (1997)
24. Mohamed, S.G., Amine, F.: Mining multi-level frequent itemsets under constraints. Int. J. Database Theory Appl. **3**, 15–34 (2010)
25. Nguyen, R.T., Lakshmanan, V.S., Han, J., Pang, A.: Exploratory mining and pruning optimizations of constrained association rules. In: Proceedings of the 1998 ACM-SIG-MOD International Conference on the Management of Data, pp. 13–24 (1998)
26. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Efficient mining of association rules using closed itemset lattice. Inf. Syst. **24**(1), 25–46 (1999)
27. Pasquier, N., Taouil, R., Bastide, Y., Stumme, G., Lakhal, L.: Generating a condensed representation for association rules. Intell. Inf. Syst. **24**, 29–60 (2005)
28. Pei, J., Han, J., Lakshmanan, L.V.S.: Mining frequent itemsets with convertible constraints. In: Proceedings of IEEE ICDE'01, pp. 433–442 (2001)
29. Russel, P., Sangeetha, K.: FGC: An efficient constraint based frequent set miner. In: Proceedings of IEEE, pp. 424–431 (2007)
30. Srikant, R., Vu, Q., Agrawal, R.: Mining association rules with item constraints. In: Proceedings of KDD'97, pp. 67–73 (1997)
31. Szathmary, L., Valtchev, P., Napoli, A.: Efficient vertical mining of frequent closed itemsets and generators. IDA **2009**, 393–404 (2009)
32. Uday Kiran, R., Krishna Reddy, P.: Towards Efficient Mining of Periodic-Frequent Patterns in Transactional Databases. Database Expert Syst. Appl. **6262**, 194–208 (2010)
33. Varsha, M., Anju, S.: Efficient approach for extracting frequent pattern and association rules with periodic constraints. Int. J. Comput. Sci. Eng. Inf. Technol. Res. **3**, 65–78 (2013)
34. Wille, R.: Concept lattices and conceptual knowledge systems. In: Computers and Mathematics with Applications, pp. 493–515 (1992)
35. Woon, Y.-K., W.-K. Ng, et al.: A support-ordered trie for fast frequent itemset discovery. IEEE Trans. Knowl. Data Eng. **16**(7), 875–879 (2004)
36. Zaki, M.J.: Mining non-redundant association rules. Data Min. Knowl. Discov. **9**(3), 223–248 (2004)
37. Zaki, M.J., Hsiao, C.J.: Efficient algorithms for mining closed itemsets and their lattice structure. IEEE Trans. Knowl. Data Eng. **17**, 462–478 (2005)