Contents lists available at ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

Editorial

Fast updated frequent-itemset lattice for transaction deletion

Bay Vo[a], Tuong Le[b,c,*], Tzung-Pei Hong[d,e], Bac Le[f]

[a] Faculty of Information Technology, Ho Chi Minh City University of Technology, Viet Nam
[b] Division of Data Science, Ton Duc Thang University, Ho Chi Minh City, Viet Nam
[c] Faculty of Information Technology, Ton Duc Thang University, Ho Chi Minh City, Viet Nam
[d] Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung, Taiwan, ROC
[e] Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung, Taiwan, ROC
[f] Department of Computer Science, University of Science, VNU-Ho Chi Minh, Viet Nam

ARTICLE INFO

Available online 11 April 2015

Keywords:
Data mining
Frequent-itemset lattice
Transaction deletion

ABSTRACT

The frequent-itemset lattice (FIL) is an effective structure for mining association rules. However, building an FIL for a modified database requires a lot of time and memory. Currently, there is no approach for updating an FIL with deleted transactions. Therefore, this paper proposes an approach for maintaining FILs for transaction deletion without rescanning the original database if the number of eliminated transactions is smaller than the threshold determined based on the pre-large and diffset concepts. A diffset-based approach is first used for fast building an FIL. Then, two proposed approaches (tidset-based and diffset-based) are used for updating the FIL with transaction deletion. The experiment was conducted to show that the diffset-based approach outperforms the tidset-based and the batch-mode approaches.

1. Introduction

Association rule (AR) mining [1,15,24] is an important problem, which attracts so much attention of scientists, in data mining and knowledge discovery. They have wide applications, such as basket data analysis, semantic web mining, text mining and so on. The traditional methods for mining ARs are divided into two phrases: (i) Mining frequent itemsets (FIs) from databases [5,7,8,18] and streaming databases [4,19] and (ii) mining ARs from FIs. According to the experiments, phase (ii) is easily implemented but it requires a lot of processing time. Recently, frequent-itemset lattices (FILs) and frequent-closed-itemset lattices (FCILs) have been proposed for effectively mining ARs [13,20,21,25]. Building FILs/FCILs takes longer than getting frequent (closed) itemsets, but generating ARs from FILs/FCILs is more efficient than doing so from frequent (closed) itemsets [17,20]. Therefore, mining ARs based on FILs/FCILs outperforms the traditional approach when both phases of mining are considered.

In practical applications, databases are typically modified, meaning that transactions are often inserted or deleted. For instance, inserted transactions will be added to database of the system when customers buy something. Besides, when customers return their orders or there are a number of errors in orders, those will be removed from the transaction databases. Therefore, mining ARs, frequent itemsets, class association rules and high utility patterns from modified databases [6,9,16,23] have attracted much research interest. Fast-UPdate (FUP) [3] is the first algorithm for mining ARs from incremental databases. FUP is an Apriori-based algorithm that generates candidates and repeatedly scans databases. Since then, methods based on FP-tree [9,10,12] and IT-tree [14] have been developed for databases with transaction insertion. Incremental mining from sequence databases has also been developed [2]. Some studies have considered transaction deletion [11]. However, there are no methods proposed for maintaining FILs with transaction deletion.

* Corresponding author at: Division of Data Science, Ton Duc Thang University, Ho Chi Minh City, Viet Nam.
E-mail addresses: bayvodinh@gmail.com (B. Vo), lecungtuong@tdt.edu.vn, tuonglecung@gmail.com (T. Le), tphong@nuk.edu.tw (T.-P. Hong), lhbac@fit.hcmus.edu.vn (B. Le).

http://dx.doi.org/10.1016/j.datak.2015.04.006
0169-023X/© 2015 Elsevier B.V. All rights reserved.

To deal with the problem of maintaining frequent itemsets for transaction modification, the pre-large concept is proposed to reduce the need for rescanning an original database. With this concept, the original database does not need to be rescanned if the number of deleted transactions or inserted transactions is equal to or less than a safety threshold, thus reducing the maintenance cost. The pre-large concept was later used by La et al. [13] and Vo et al. [21] for fast updating FCILs with transaction insertion.

This paper proposes an approach for maintaining FILs with transaction deletion. First, the proposed approach uses the DFIL algorithm based on the diffset concept to build FILs. Then, two methods for updating FILs (tidset-based and diffset-based methods) with transaction deletion are used.

The rest of this paper is organized as follows. Section 2 presents the basic concepts and an effective algorithm based on the diffset concept for building FILs. Two algorithms for maintaining FILs based on the tidset and the diffset concepts, respectively, with transaction deletion are proposed in Section 3. Section 4 presents the results of experiments that compare the run time of the proposed algorithms with that of the batch-mode approaches to show the effectiveness of the proposed algorithms. Finally, Section 5 summarizes the results and offers some future topics.

2. Basic concepts

2.1. Frequent-itemset lattice building algorithm

Given a database D with n transactions, with each transaction including a set of items belonging to I, where I is the set of all items in D. An example of a transaction database D_1 is presented in Table 1. The support of an itemset X, denoted by $\sigma(X)$, where $X \subseteq I$, is the number of transactions in D which contains all the items in X. An itemset X is called a frequent itemset if $\sigma(X) \geq minSup \times n$, where $minSup$ is a given threshold.

Vo et al. [21] proposed the DFIL algorithm for building FILs. It is summarized as follows.

Definition 1. Let $n(X)$ be a node of a k-itemset X. The child-nodes of $n(X)$ based on the equivalence class property associated with $n(X)$ are:

$$y_{EC}(n(X)) = \{n(XA) | \forall A \in I, A \notin X\} \tag{1}$$

Definition 2. Let X be a k-itemset. The child-nodes of $n(X)$ based on the lattice property associated with $n(X)$ are:

$$y_L(n(X)) = \{n(Y) | Y \text{ is a } (k+1)-\text{item set}, n(Y) \notin y_{EC}(n(X)) \text{ and } X \subset Y\} \tag{2}$$

Definition 3. Each node, $n(X)$, in the FIL is a tuple:

$$\langle \mathrm{X}, t(\mathrm{X}), y_{EC}(n(\mathrm{X})), y_L(n(\mathrm{X}))\rangle, \tag{3}$$

where:

- X is an itemset;
- $t(X)$ is the set of IDs associated with the transactions containing X; and
- $\gamma_{EC}(n(X))$ contains the child-nodes based on the equivalence class property associated with X.
- $\gamma_L(n(X))$ contains the child-nodes based on the lattice property associated with X.

Theorem 1. Let $n(XA)$ be a node of a k-itemset XA. $\forall n(XB) \in \gamma_{EC}(n(X))$, if A is before B in the order of frequent 1-itemsets (sorted in ascending order of frequency), then $\nexists n(Y) \in \gamma_{EC}(n(XB)) \cup \gamma_L(n(XB))$ so that $n(Y) \in \gamma_L(n(XA))$.

Theorem 2. Let $n(XA)$ be a node of a k-itemset XA. $\forall n(Z) \in \gamma_L(n(X))$, $\nexists\, n(Y) \in \gamma_L(n(Z))$ so that $n(Y) \in \gamma_L(n(XA))$.

To understand the application of Theorems 1 and 2, the process of updating a lattice when $n(XA)$ (a k-itemset) is created, is described below. The existing algorithms [22] have to consider all nodes of Y in the four cases shown in Table 2.

Table 1
Example of a transaction database D_1.

Transaction	Items
1	*A, C, T, W, V*
2	*C, D, W*
3	*A, C, T, W*
4	*A, C, D, W*
5	*A, C, D, T, W*
6	*C, D, T*