EFFICIENT ALGORITHMS FOR MINING OF HIGH UTILITY ITEMSETS

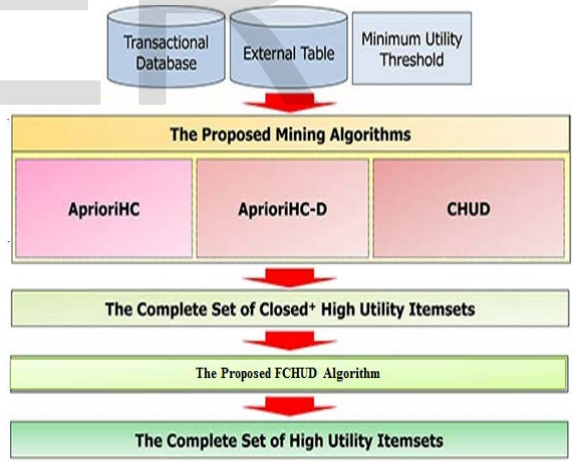
M.A. Shanti, HOD in Information Technology,Idhaya College forWomen, Kumbakonam
Dr.K.Saravanan, Dean, Faculty of Computer Science, PRIST UNIVERSITY, Vallam, Thanjavur

Abstract---The utility of an itemset represents its importance, which can be measured in terms of weight, value, quantity or other information depending on the user specification. High utility itemsets mining identifies itemsets whose utility satisfies a given threshold. It allows users to quantify the usefulness or preferences of items using different values. Thus, it reflects the impact of different items. High utility itemsets is useful in decision making process of many applications, such as retail marketing and Web service, since items are actually different in many aspects in real applications. One of its popular applications is market basket analysis, which refers to the discovery of sets of items (itemsets) that are frequently purchased together by customers. However, in this application, the traditional model of FIM may discover a large amount of frequent but low revenue itemsets and lose the information on valuable itemsets having low selling frequencies. We propose a novel framework for mining closed+ high utility itemsets (CHUIs), which serves as a compact and lossless representation of HUIs. We propose three efficient algorithms named AprioriCH (Apriori-based algorithm for mining High utility Closed+ itemsets), AprioriHC-FD (AprioriHC algorithm with Fast Discarding unpromising and isolated items) and FCHUD (Fast Closed+ High Utility itemset Discovery) and the integration of closed itemset mining and high utility itemset mining and find possible to develop other compact representations of high utility itemsets inspired by our work to reduce the number of redundant high utility patterns.

Index Terms—Frequent itemset, closed+ high utility itemset, lossless and concise representation, utility mining, data mining

1 INTRODUCTION

However, HUI mining is not an easy task since the downward closure property in FIM does not hold in utility mining. In other words, the search space for mining HUIs cannot be directly reduced as it is done in FIM because a superset of a low utility itemset can be a high utility itemset. Many studies were proposed for mining HUIs, but they often present a large number of high utility itemsets to users. A very large number of high utility itemsets makes it difficult for the users to comprehend the results. It may also cause the algorithms to become inefficient in terms of time and memory requirement, or even run out of memory. It is widely recognized that the more high utility.



Itemsets the algorithms generate, the more processing they consume. The performance of the mining task decreases greatly for low minimum utility thresholds or when dealing with dense databases.

We propose three efficient algorithms named AprioriHC (Apriori-based algorithm for mining High utility Closeditemset), AprioriHC-D (AprioriHC algorithm

with Discarding unpromising and isolated items) and CHUD (Closed+ High Utility itemset Discovery) to find this representation. The AprioriHC and AprioriHC-D algorithms employs breadth-first search to find CHUIs and inherits some nice properties from the well-known Apriori algorithm. The CHUD algorithm includes three novel strategies named REG, RML and DCM that greatly enhance its performance.

2 BACKGROUND

In this section, we introduce the preliminaries associated with high utility itemset mining, closed itemset mining and compact representations of high utility itemsets.

2.1 High utility itemset

Items are actually different in many aspects in a number of real applications, such as retail marketing, network log, etc. The difference between items makes a strong impact on the decision making in these applications. In reality, a retail business may be interested in identifying its most valuable customers (customers who contribute a major fraction of the profits to the company). These are the customers, who may buy full priced items, high margin items, or gourmet items, which may be absent from a large number of transactions because most customers do not buy these items. Another example is web log data. A sequence of webpages visited by a user can be defined as a transaction. Since the number of visits to a webpage and the time spent on a particular webpage is different between different users, the total time spent on a page by a user can be viewed as utility. The website designers can catch the interests or behavior patterns of the customers by looking at the utilities of the page combinations and then consider re-organizing the link structure of their website to cater to the preference of users. Frequency is not sufficient to answer questions, such as whether an itemset is highly profitable, or whether an itemset has a strong impact.In utility based mining the term utility refers to the quantitative representation of user preference i.e. the utility value of an itemset is the measurement of the importance of that itemset in the user's perspective. Utility mining is likely to be useful in a wide range of practical applications. One of its popular applications is *market basket analysis*, which refers to the discovery of sets of items (itemsets) that are frequently purchased together by customers. However, in this application, the traditional model of FIM may discover a large amount of frequent but low revenue itemsets and lose the information on valuable itemsets having low selling frequencies. These problems are caused by the facts that (1) FIM treats all items as having the same importance/unit profit/weight and (2) it assumes that every item in a transaction appears in a binary form, i.e., an item can be either present or absent in a transaction, which does not indicate its purchase quantity in the transaction. Hence, FIM cannot satisfy the requirement of users who desire to discover itemsets with high utilities such as high profits.

Let $I = \{a1, a2, \ldots, aM\}$ be a finite set of distinct items. A *transactional database* $D = \{T1, T2, \ldots, TN\}$ is a set of transactions, where each transaction $TR \subseteq D$ ($1 \le R \le N$) is a subset of I and has an unique identifier R, called *Tid*. Each item $ai \in I$ is associated with a positive real number $p(ai, D)$, called its *external utility*. Every item ai in the transaction TR has a real number $q(ai, TR)$, called its *internal utility*. An *itemset* $X = \{a1, a2, \ldots, aK\}$ is a set of K distinct items, where $ai \in I$, $1 \le i \le K$, and K is called the length of X. A *K-itemset* is an itemset of length K. An itemset X is said to be contained in a transaction TR..

2.2 Closed Itemset Mining

In this subsection, we introduce definitions and properties related to closed itemsets and mention relevant methods.

The Tidset of an itemset X is denoted as $g(X)$ and defined as the set of Tids of transactions containing X. The support count of X is expressed in terms of $g(X)$ as $SC(X) = |g(X)|$.

Closed itemset

An itemset $X \subseteq I$ is a closed itemset iff there exists no itemset $Y \subseteq I$ such that (1) $X \subset Y$ and (2) $SC(X) = SC(Y)$. Otherwise, X is non-closed itemset. An equivalent definition is that X is closed iff $C(X) = X$.

Complete set of closed itemset in the database

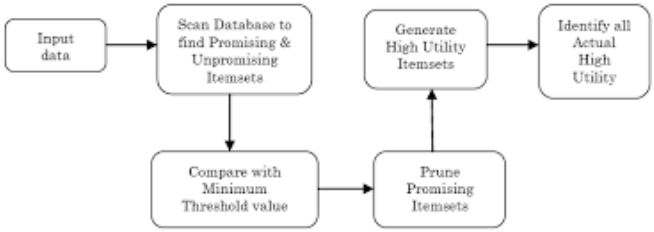
Let S be a set of itemsets and a function $fC(S) = \{X | X \subseteq S, \nexists Y \subseteq S$ such that $X \subset Y$ and $SC(X) = SC(Y)\}$. The complete set of closed itemsets in D is denoted as C ($C \subseteq L$) and defined as $fC(L)$.

2.3 CLOSED+ HIGH UTILITY ITEMSET MINING

The first point that we should discuss is how to incorporate the closed constraint into high utility itemset mining. There are several possibilities. First, we can define the closure on the utility of itemsets. In this case,

a high utility itemset is said to be closed if it has no proper superset having the same utility.

There are seven HUIs in Example 1 and only one itemset {E} is non-closed, since {E}⊐ {ABE} and *au*({E}) = *au*({ABE}) = 12. A second possibility is to define the closure on the supports of itemsets.



3. Efficient Algorithms for Mining Closed+ High Utility Itemsets

We propose a novel framework for mining closed+ high utility itemsets (CHUIs), which serves as a compact and lossless representation of HUIs. We propose three efficient algorithms named AprioriCH (Apriori-based algorithm for mining High utility Closed+ itemsets), AprioriHC-FD (AprioriHC algorithm with Fast Discarding unpromising and isolated items) and FCHUD (Fast Closed+ High Utility itemset Discovery) and the integration of closed itemset mining and high utility itemset mining and find the possible to develop other compact representations of high utility itemsets inspired by our work to reduce the number of redundant high utility patterns.

Algorithm 1: The FHM algorithm

input : D: a transaction database, minutil: a user-specified threshold

output: the set of high-utility itemsets

1 Scan D to calculate the TWU of single items;

2 I ∗ ← each item i such that TWU(i) <minutil;

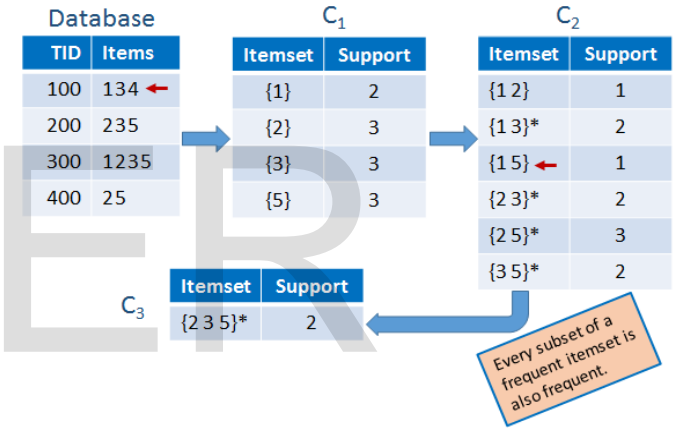
3 Let be total order of TWU ascending values on I ∗ ;

4 Scan D to built the utility-list of each item i ∈ I ∗ and build the EUCS structure;

5 Search (∅, I ∗ ,minutil, EUCS);

4. AprioriHC Algorithm

A variable *k* is set to 1. The algorithm performs a database scan to compute the transaction utility of each transaction (Definition 4). At the same time, the TWU of each item is computed. Each item having a TWU no less than *abs_min_utility*is added to the set of 1-HTWUIs *Ck*. Then the algorithm proceeds recursively to generate itemsets having a length greater than *k*. During the *k*-th iteration, the set of *k*-HTWUIs *Lk*is used to generate (*k*+1)- candidates *Ck+1* by using the *Apriori-gen function* [1]. Then the algorithm computes TWUs of itemsets in *Ck+1* by scanning the database *D* once.



An Example for Apriori Algorithm

5. The AprioriHC-FD Algorithm

The AprioriHC-FD algorithm is an improvement of AprioriHC. It includes two effective strategies to reduce the number of PCHUIs generated in Phase I that are inspired by the UP-Growth [24] and IIDS algorithms [19].

PROCEDURE: AprioriHC-FD_Phase-I

Input: *D*k: the trimmed database; *abs_min_utility*;

pCHUI: the set of PCHUIs

Output: the complete set of PCHUIs

01. *Lk+1* := *Lk*

02. ***for***(*k* = 1; *Lk+1* ≠∅ *k*++)

03. { *Ck+1* := ***Apriori-gen***(Lk)

04. *Lk+1* := ***EstimatedUtilityOf_Itemsets***(*Dk*, *Ck+1*)

05. *Lk*:=***MarkClosed_Itemsets***(*Lk*, *Lk+1*)

06. *pCHUI*:= *pCHUI*∪*Lk*-1
07. *Dk+1* := ***IIDS_Strategy***(*Dk*, *Lk+1*)
08. }

PROCEDURE: *AprioriHC-D_Phase*-II

Input: *D1*: the database containing no unpromising items;
pCHUI: set of PCHUIs; *abs_min_utility*
Output: the complete set of CHUIs
01. ***for***(*k* := 1; *Lk*≠ □*k*++)
02. { *Lk*:= *k*-itemsets in *pCHUI*
03. ***for all*** *X* in *Lk****do***
04. { ***Calculate****au*(*X*) and utility unit array of *X* ***from*** *Dk*
05. ***If*** *au*(*X*) ≥ *abs_min_utility****then*** { ***output****X* }
06. }
07. *Dk+1* := ***IIDS_Strategy***(*Dk*, *Lk*)
08. }

It takes as parameters, the database *D1*, the set of PCHUIs *pCHUI*and *abs_min_utility*. The procedure performs *k* iterations starting from *k* = 1 to the maximum length of PCHUIs in *pCHUI*. During the *k*-th iteration, the procedure considers the set of *k*-PCHUIs *Lk*in *pCHUI*. For each PCHUI *X* in *Lk*, the absolute utility and utility unit array is calculated by scanning *Dk*. If the absolute utility of *X* is no less than *abs_min_utility*, *X* is outputted as a CHUI. Then, the algorithm applies the IIDS strategy to remove isolated items of level *k* from *Dk*.

6. The FCHUD Algorithm

We present an efficient depth-first search algorithm named F*CHUD* (Fast and *Closed+ High Utility itemset Discovery*) to discover FCHUIs. FCHUD is an extension of DCI-Closed, one of the currently best methods to mine closed itemsets. FCHUD is adapted for mining CHUIs and include several effective strategies for reducing the number of candidates generated in Phase I. Similar to the DCI-Closed algorithm, CHUD adopts an *IT-Tree* (*Itemset-Tidsetpair Tree*) to find CHUIs. In an IT-Tree, each node *N*(*X*) consists of an itemset *X*, its *Tidset g*(*X*), and two ordered sets of items named *PREV-SET*(*X*) and *POST-SET*(*X*). The IT-Tree is recursively explored by the CHUD algorithm until all closed itemsets that are HTWUIs are generated. Different from the DCI-Closed algorithm, each node *N*(*X*) of the IT-Tree is attached with an estimated utility value *EstU*(*X*).

7. Redundant high utility patterns

The possible to develop other compact representations of high utility itemsets inspired by our work to reduce the number of redundant high utility patterns. This is an interesting research question. Although closed+ high utility itemset mining is essential to many research topics and industrial applications, it is still a novel and challenging problem.

8. CONCLUSIONS

We proposed three efficient algorithms named *AprioriHC* (*Apriori-based approach for mining High utility Closed itemset*), *AprioriHC-FD* (*AprioriHC algorithm with Fast Dis-carding unpromising and isolated items*) and F*CHUD* (Fast and *Closed+ High utility itemset Discovery*). AprioriHC-D is an improved version of AprioriHC, which incorporates strategies DGU [24] and IIDS [19] for pruning candidates. *AprioriHC* and *AprioriHC-D* perform a breadth-first search for mining closed+ high utility itemsets from horizontal database, while CHUD performs a depthfirst search for mining closed+ high utility itemsets from vertical database.

ACKNOWLEDGMENT

We would like to thank everyone who has participated in the evaluation of the prototype system. The authors aregrateful for the constructive comments of the three referees on an earlier version of this article. This research was supported in different part by real time transaction data set.

Reference

[1] R. Agrawal and R. Srikant, “Fast Algorithms for Mining Association Rules,” in *Proc. of the 20th Int'l Conf. on Very Large Data Bases*, pp. 487-499, 1994.

[2] C. F. Ahmed, S. K. Tanbeer, B.-S. Jeong, and Y.-K. Lee, “Efficient Tree Struc-tures for High utility Pattern Mining in Incremental Databases,” *IEEE Trans actions on Knowledge and Data Engineering*, Vol. 21, Issue 12, pp. 1708-1721, 2009.

[3] J. -F. Boulicaut, A. Bykowski, and C. Rigotti, “Free-sets: A Condensed Repre-sentation of Boolean Data for the Approximation of Frequency Queries,” *Da-ta Mining and Knowledge Discovery*, Vol. 7, Issue 1.

[4] T. Calders and B. Goethals, “Mining All Non-derivable Frequent Itemsets,” in *Proc. of the Int’l Conf. on European Conference on Principles of Data Mining \ and Knowledge Discovery*, pp. 74-85, 2002.

[5] K. Chuang, J. Huang, M. Chen, “Mining Top-K Frequent Patterns in the Presence of the Memory Constraint,” *VLDB Journal*, Vol. 17, pp. 1321-1344, 2008.

[6] R. Chan, Q. Yang, and Y. Shen, “Mining High Utility Itemsets,” *in Proc. of the IEEE Int'l Conf. on Data Mining*, pp. 19-26, 2003.