International Journal of Research in Advent Technology, Vol.2, Issue 4, April 2014

E-ISSN: 2321-9637

A Strategic Implementation in Incremental High Utility Pattern Mining Algorithm

Mr. Chetan V. Chaudhari[1], Prof. Sandeep Khanna[2]
Research Scholar[1],
Dr. V.B.Kolte COE, Malkapur[1]
chaudharichetanv@gmail.com
Department of Computer Engineering[2]
Dr. V.B.Kolte COE, Malkapur [2]
sandeepietse@gmail.com[2]

Abstract In recent years, the problem of high utility pattern mining become one of the most important research area in data mining. The existing high utility mining algorithm generates large number of candidate itemsets, which takes much time to find utility value of all candidate itemsets, especially for dense datasets. In this paper we are implementing two more strategies in IHUP(Incremental High Utility Pattern mining) algorithm to further reduce number of PHUIs(Potentially High Utility Itemsets) and to reduce execution time. This algorithm with two strategies is compared with other existing algorithms in various aspects. Experimental results show that the proposed algorithms reduce the number of candidates effectively.

Keywords— IHUP, PHUIs, Dense Database.

1. INTRODUCTION

Extensive studies have been proposed for mining frequent patterns. One of the well-known algorithms for mining association rules is Apriori, which is the pioneer for efficiently mining association rules from large databases. Pattern growth-based association rule mining algorithms such as FP-Growth were afterward proposed. In the framework of frequent itemset mining, the importance of items to users is not considered. Thus, the topic called weighted association rule mining was brought to attention. Although weighted association rule mining considers the importance of items, quantities in transactions are not taken into considerations yet. Thus, the issue of high utility itemset mining is raised and many studies have addressed this problem. Liu et al. proposed an algorithm named Two- Phase which is mainly composed of two mining phases. It still generate many HTWUIs.

Although two-phase algorithm reduces search space by using TWDC(Transaction Weighted Downward Closure) property, it still generates too many candidates to obtain HTWUIs and requires multiple database scans. In phase I, to efficiently create HTWUIs and several times avoid scanning database, Ahmed discovered a IHUP tree-based algorithm. An IHUP was one of the effective algorithm to create utility itemsets.

IHUP algorithm has three stages:

1) construction of IHUP-Tree
2) generation of HTWUIs, and
3) Identification of high utility itemsets.

In stage 1, items are rearranged (lexicographic order) in a fixed order, support descending order or Transaction Weighted Utility descending order. after rearrangement transactions are feed into an IHUP-Tree.

In stage 2, HTWUIs are created from the IHUP-Tree by applying FP-Growth [14].

For the performance result of algorithm, the number of generated HTWUIs is a major issue. Due to that our aim is to reducing Itemset by several strategies. The number of created candidates can be highly minimized in phase I and high utility itemsets can be identified more efficiently in phase II by applying the proposed strategies.[17]

2. RELATED WORK

2.1 *High Utility Itemset Mining*

In some applications such as transaction databases, though weighted association rule mining considers the importance of items, items' quantities in transactions are not taken into considerations yet. Thus, the issue of high utility itemset mining is raised. Liu et al. proposed an algorithm named Two-Phase which is mainly composed of two mining phases. But it generates too many candidates to obtain high transaction weighted utility itemsets and requires multiple database scans.

2.1.1 *IHUP algorithms*

Another tree based algorithm was proposed, named IHUP [3] to efficiently generate HTWUIs and avoid multiple time database scanning. It uses a tree based structure IHUP-Tree [3] to maintain the information about itemsets and their utilities. It first generate IHUP tree and then generate HTWUIs from tree and at last performs mining on that itemset. To perform this operation it uses two database scan. In first scan it generates tree and during second scan it uses FP-Growth algorithm.

However this algorithm also generates too many candidates. Hence also require more execution time. Hence we include a new UP-Tree structure and applies various strategies on IHUP algorithm [3] to reduce HTWUIs(High Transaction Weighted Utility Itemsets).

3. PROBLEM DEFINITION

we first give some definitions and define the problem of utility mining, and then introduce related work in utility mining.

TID	Transaction	TU
T_1	(A,2) (C,1) (D,9)	12
T_2	(A,1) (C,10) (E,1) (G,5)	26
T_3	(A,6) (B,2) (D,2) (E,1) (F,2)	36
T_4	(B,1) (C,3) (D,1) (E,3)	32
T_5	(B,4) (C,1) (E,4) (G,2)	12
T_6	(A,2) (B,1) (C,1) (D,1) (H,1)	13

Table 1 : Database Example

Item	A	B	C	D	E	F	G	H
Profit	2	5	2	1	5	3	2	1

Table 2: Profit table

A finite set of items I ={ i_1, i_2, i_3,…., i_m}, each item i_p(m > p > 1) has a unit profit pr(i_p). An itemset X is a set of k distinct items {i_1, i_2, …., i_k }, where $i_j \in$ I; 1 $\leq$ j $\leq$ k. k is the length of X. An itemset with length k is called a kitemset. A transaction database D ={T_1, $T_{2,….,}$ T_n } contains a set of transactions, and each transaction T_d (1$\leq$d$\leq$ n) has a unique identifier d, called TID. Each item i_p in transaction T_d is associated with a quantity q(i_p, T_d), that is, the purchased quantity of i_p in T_d.

Definition 1. Utility of an item i_p in a transaction T_d is denoted as u(i_p, T_d) and defined as pr(i_p) × q(i_p , T_d).

Definition 2. Utility of an itemset X in T_d is denoted as u(X, T_d) and defined as $\sum_{i_p \in X \wedge X \subseteq T_d}$ u(i_p, T_d).

Definition 3. Utility of an itemset X in D is denoted as u(X) and defined as $\sum_{X \subseteq T_d \wedge T_d \in D}$ u(X, T_d).

Definition 4. An itemset is called a high utility itemset if its utility is no less than a user-specified minimum utility threshold which is denoted as min_util. Otherwise, it is called a low-utility itemset.

Definition 5. Transaction utility of a transaction T_d is denoted as TU(T_d) and defined as u(T_d, T_d).

Definition 6. Transaction-weighted utility of an itemset X is the sum of the transaction utilities of all the transactions containing X, which is denoted as TWU(X) and defined as $\sum_{X \subseteq T_d \wedge T_d \in D}$ TU(T_d).

Definition 7. An itemset X is called a high-transaction weighted utility itemset (HTWUI) if TWU(X) is no less than min_util.

Property 1 (Transaction-weighted downward closure.). For any itemset X, if X is not a HTWUI, any superset of X is a low utility itemset.

4. THE PROPOSED DATA STRUCTURE: UP-TREE

For the performance of mining with avoiding again and again scanning original database, we prefer UP-Tree compact tree structure.[13] So, transactional information and high utility itemsets are maintained. To minimize the overestimated utilities stored in the nodes of global UP-Tree, two stages are used. In following sections, the elements of UP-Tree are first defined. Next, the two strategies are introduced. Finally, how to construct an UP-Tree with the two strategies is illustrated in detail.[14]

4.1 *The Elements in UP-Tree*

In an UP-Tree, each node N consists of N.name, N.count, N.nu, N.parent, N.hlink and a set of child nodes. N.name is the node's item name. N.count is the node's support count. N.nu is the node's node utility, i.e., overestimated utility of the node. N.parent records the parent node of N. N.hlink is a node link which points to a node whose item name is the same as N.name. A table named header table is employed to facilitate the traversal of UP-Tree. In header table, each entry records an item name, an overestimated utility, and a link. The link points to the last occurrence of the node which has the same item as the entry in the UP-Tree. By following the links in header table and the nodes in UP-Tree, the nodes having the same name can be traversed efficiently.

In following sections, two strategies for decreasing the overestimated utility of each item during the construction of a global UP-Tree are introduced.

4.1.1 *Strategy DGU:* Discarding Global Unpromising Items during Constructing a Global UP-Tree

The construction of a global UP-Tree can be performed with two scans of the original database. In the first scan, TU of each transaction is computed. At the same time, TWU of each single item is also accumulated. By TWDC property, an item and its supersets are unpromising to be high utility itemsets if its TWU is less than the minimum utility threshold. Such an item is called an unpromising item.

Property 2 (Antimonotonicity of unpromising items). If i_u is an unpromising item, i_u and all its supersets are not high utility itemsets.

Corollary 1. Only the supersets of promising items are possible to be high utility itemsets.[14]

During the second scan of database, transactions are inserted into a UP-Tree. When a transaction is retrieved, the unpromising items should be removed from the transaction and their utilities should also be eliminated from the transaction's TU according to Property 2 and Corollary 1.

This concept forms our first strategy.

Strategy 1. DGU: *Discarding global unpromising items and their actual utilities from transactions and transaction utilities of the database.*

New TU after pruning unpromising items is called reorganized transaction utility (RTU). RTU of a reorganized transaction Tr is denoted as RTU(Tr). By reorganizing the transactions, not only less information is needed to be recorded in UP-Tree, but also smaller overestimated utilities for itemsets are generated. Strategy DGU uses RTU to overestimate the utilities of itemsets instead of TWU. Since the utilities of unpromising items are excluded, RTU must be no larger than TWU. Therefore, the number of PHUIs with DGU must be no more than that of HTWUIs generated with TWU [3]. DGU is quite effective especially when transactions contain lots of unpromising items, such as those in sparse data sets. Besides, DGU can be easily integrated into TWU-based algorithms [3], [15]. Moreover, before constructing an UP-Tree, DGU can be performed repeatedly till all reorganized transactions contain no global unpromising item. By performing DGU for several times, the number of PHUIs will be reduced; however, it needs several database scans.

4.1.2 *Strategy DGN:* Decreasing Global Node Utilities during Constructing a Global UP-Tree

It is shown that the tree-based framework for high utility itemset mining applies the divide-and-conquer technique in mining processes. Thus, the search space can be divided into smaller subspaces. For example, in Fig. 1, the search space can be divided into the following subspaces:

1. {B}'s conditional tree (abbreviated as {B}-Tree),

2. {A}-Tree without containing {B},

3. {D}-Tree without containing {B} and {A},

4. {C}-Tree without containing {B}, {A}, and {D}, and

5. {E}-Tree without containing {B}, {A}, {D}, and {C}.

It can be observed that in the subspace {A}-Tree, all paths are not related to {B} since the nodes {B} are below the nodes {A} in global IHUP-Tree. In other words, the items that are descendant nodes of the item i_m will not appear in $\{i_m\}$- Tree; only the items that are ancestor nodes of i_m will appear in $\{i_m\}$-Tree. From this viewpoint, our second proposed strategy for decreasing overestimated utilities is to remove the utilities of descendant nodes from their node utilities in global UP-Tree. The process is performed during the construction of the global UP-Tree.

Strategy 2. DGN: *Decreasing global node utilities for the nodes of global UP-Tree by actual utilities of descendant nodes during the construction of global UP-Tree.*

By applying strategy DGN, the utilities of the nodes that are closer are further reduced. DGN is basically suitable for the long transactional databases. Means, the more transactional items, the more utilities can be discarded by DGN. Our traditional TWU mining model is not suitable for such databases since the more items a transaction contains, the higher TWU is. In following sections, we describe the process of constructing a global UP-Tree with strategies DGU and DGN.

4.2 *Making a Global UP-Tree by using DGU and DGN*

The construction of a global UP-Tree by using two database scans. In the first scan, each transaction's TU is computed; at the same time, each 1-item's TWU is also collected. Then, we can get promising items and unpromising items. After getting all promising items, DGU is applied. The transactions are reorganized by pruning the unpromising items and sorting the remaining promising items in a fixed order. Lexicographic, support, or TWU order can be used. Then above rearrangement is called a reorganized transaction.[14]

CONCLUSION

We proposed a UP Tree structure to mine High Utility Itemsets with two strategies DGU and DGN in IHUP Algorithm to improve mining performance. Algorithm performed in Java and tested on various data sets.

REFERENCES

[1] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proc. 20th Int'l Conf. Very Large Data Bases (VLDB), pp. 487-499, 1994.

[2] R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proc. 11th Int'l Conf. Data Eng., pp. 3-14, Mar. 1995.

[3] C.F. Ahmed, S.K. Tanbeer, B.-S. Jeong, and Y.-K. Lee, "Efficient Tree Structures for High Utility Pattern Mining in Incremental Databases," IEEE Trans. Knowledge and Data Eng., vol. 21, no. 12, pp. 1708-1721, Dec. 2009.

[4] Jyothi Pillai, O.P.Vyas "Overview of Itemset Utility Mining and its Applications" IJCA(0975 – 8887) Volume 5– No.11, August 2010

[5] C.H. Cai, A.W.C. Fu, C.H. Cheng, and W.W. Kwong, "Mining Association Rules with Weighted Items," Proc. Int'l Database Eng. and Applications Symp. (IDEAS '98), pp. 68-77, 1998.

[6] J. Han and Y. Fu, "Discovery of Multiple-Level Association Rules from Large Databases,"

Proc. 21th Int'l Conf. Very Large Data Bases, pp. 420-431, Sept. 1995.

[7] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," Proc. ACM-SIGMOD Int'l Conf. Management of Data, pp. 1-12, 2000.

[8] J. Pei, J. Han, H. Lu, S. Nishio, S. Tang, and D. Yang, "H-Mine: Fast and Space-Preserving Frequent Pattern Mining in Large Databases," IIE Trans. Inst. of Industrial Engineers, vol. 39, no. 6, pp. 593-605, June 2007.

[9] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Moal, and M.C. Hsu, "Mining Sequential Patterns by Pattern-Growth: The Prefixspan Approach," IEEE Trans. Knowledge and Data Eng., vol.16, no.10, pp. 1424-1440, Oct. 2004.

[10] K. Sun and F. Bai, "Mining Weighted Association Rules without Preassigned Weights," IEEE Trans. Knowledge and Data Eng., vol. 20, no. 4, pp. 489-495, Apr. 2008.

[11] *Guo-Cheng Lan, Tzung-Pei Hong, Vincent S. Tseng* "Mining High Transaction-Weighted Utility Itemsets" IEEE 2010

[12] Chithra Ramaraju, Nickolas Savarimuthu "A Conditional Tree Based Novel Algorithm for High Utility Item set Mining" IEEE 2011

[13] Vincent S. Tseng, Bai-En Shie, Cheng-Wei Wu, and Philip S. Yu, Fellow, "Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases" IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 25, NO. 8, AUGUST 2013.

[14] IJCSET Volume 3 Issue 1/ijcset2013030104.pdf

[15] Y. Liu, W. Liao, and A. Choudhary, "A Fast High Utility Itemsets Mining Algorithm," Proc. Utility-Based Data Mining Workshop, 2005.

[16] Frequent Itemset Mining Implementations Repository, http://fimi.cs.helsinki.fi/, 2013.

[17] IOSRJOURNAL Volume 16Issue 2 Version 1 p016219193.pdf