ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 4, April 2016

CHUIs-Concise and Lossless representation of High Utility Itemsets

Vandana K V[1], Dr Y.C Kiran[2]

P.G. Student, Department of Computer Science & Engineering, BNMIT, Bengaluru, India[1]

Associate Professor, Department of Computer Science & Engineering, BNMIT, Bengaluru, India[2]

ABSTRACT: Mining high utility itemsets (HUIs) from databases is an important data mining task, which refers to the discovery of itemsets with high utilities (e.g. high profits). However, it may present too many HUIs to users, which also degrades the efficiency of the mining process. To achieve high efficiency for the mining task and provide a concise mining result to users, this paper proposes a novel framework for mining**Closed**$^+$ high utility itemsets (CHUIs), which serves as a compact and lossless representation of HUIs and three efficient algorithms named as AprioriHC (Apriori-based algorithm for mining High utility **Closed**$^+$ itemsets), AprioriHC-D (AprioriHC algorithm with Discarding unpromising and isolated items) and CHUD (**Closed**$^+$ High Utility Itemset Discovery) to find this representation. Further, a method called DAHU (Derive All High Utility Itemsets) is proposed to recover all HUIs from the set of CHUIs without accessing the original database.

KEYWORDS: Frequent itemset,**Closed**$^+$high utility itemset, lossless and concise representation, utility mining, data mining

I. INTRODUCTION

Frequent itemset mining (FIM) [1], is a fundamental research topic in data mining. 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 FIM treats all items as having the same importance/unit profit/weight and 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.

To address these issues, utility mining [2], emerges as an important topic in data mining. In utility mining, each item has a weight (e.g. unit profit) and can appear more than once in each transaction (e.g. purchase quantity). The utility of an itemset represents its importance, which can be measured in terms of weight, profit, cost, quantity or other information depending on the user preference. An itemset is called a high utility itemset (HUI) if its utility is no less than a user-specified minimum utility threshold; otherwise, it is called a low utility itemset. Utility mining is an important task and has a wide range of applications such as website click stream analysis, cross-marketing in retail store, mobile commerce environment and biomedical applications. 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 and hence presents a large number of high utility itemsets to users.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 4, April 2016

II. RELATED WORK

- **High Utility Itemset Mining:** The algorithms such as Two-Phase [2] and UP-Growth [3] use TWDC property to find HUIs. They consist of two phases. In Phase I, they find all HTWUIs from the database. In Phase II, HUIs are identified from the set of HTWUIs by calculating the exact utilities of HTWUIs. Although these methods capture the complete set of HUIs, they may generate too many candidates in Phase I, i.e., HTWUIs, which degrades the performance of Phase II and the overall performance. To reduce the number of candidates in PhaseI, various strategies have been proposed such as DGU, DGN, DLU and DLN, for mining HUIs. But a large number of HUIs and candidates cause this method to suffer from long execution time and huge memory consumption.

- **Closed Itemset Mining:** Mining Frequent Closed Itemset (FCI) refers to the discovery of all the closed itemsets having a support no less than a user specified threshold. It is widely recognized that the number of FCIs can be much smaller than the set of frequent itemsets for real-life databases and that mining FCIs can also be much faster and memory efficient than mining FIs (Frequent Itemsets). The set of closed itemsets is lossless since all FIs and their supports can be easily derived from without scanning the original database. Many efficient methods were proposed for mining FCIs, such as A-Close, CLOSET+ and DCI-Closed[4]. However, these methods do not consider the utility of itemsets. Therefore, they may present lots of closed itemsets with low utilities to users and omit several high utility itemsets.

- **Compact Representations of High Utility Itemset Mining:** To present representative HUIs to users, some concise representations of HUIs were proposed. Chan et al. introduced the concept of utility frequent closed patterns [6]. However, it is based on a definition of high utility itemset. Shie et al. proposed a compact representation of HUIs, called maximal high utility itemset and the GUIDE algorithm for mining it [7].A HUI is said to be maximal if it is not a subset of any other HUI. Although this representation reduces the number of extracted HUIs, it is not lossless. The reason is that the utilities of the subsets of a maximal HUI cannot be known without scanning the database. Besides, recovering all HUIs from maximal HUIs can be very inefficient because many subsets of a maximal HUI can be low utility. Another problem is that the GUIDE algorithm cannot capture the complete set ofmaximal HUIs.

III. CLOSED+ HIGH UTILITY ITEMSET MINING

The Closed$^+$ High Utility Itemset Mining method uses efficient algorithms and a method to mine CHUIs which are discussed below:

- **AprioriHC and AprioriHC-D Algorithms:** The AprioriHC and AprioriHC-D are based on Apriori [1] and the Two-Phase [2] algorithms. They use a horizontal database and explore the search space of CHUIs in a breadth-first search. The algorithm AprioriHC is regarded as a baseline algorithm and AprioriHC-D is an improved version of AprioriHC. Both algorithms consist of two phases named as Phase I and Phase II. In Phase I, potential Closed$^+$ high utility itemsets (PCHUIs) are found, which are defined as a set of itemsets having an estimated utility (e.g. TWU) no less than abs_min_utility. In Phase II, by scanning the database once, CHUIs are identified from the set of PCHUIs found in Phase I and their utility unit arrays are computed. The AprioriHC-D algorithm is an improvement of AprioriHC. It includes two effective strategies such as DGU (Discarding Global Unpromising items) and IIDS (Isolated Items Discarding Strategy)to reduce the number of PCHUIs generated in Phase I.

AprioriHC-D algorithm

*Input:*D:the database; abs_min_utility;
pCHUI:the set of PCHUIs pCHUI
*Output:*The complete set of CHUIs

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 4, April 2016

01. pCHUI:=ϕ
02. L_1:=1-HTWUIs in D
03. D_1:=DGU_Strategy(D,L_1)
04. L_1:=1-HTWUIs in D_1
05. AprioriHC-D_Phase-I(D_1,pCHUI,abs_min_utility,L_1)
06. AprioriHC-D_Phase-II(D_1,pCHUI,abs_min_utility)

- **CHUD Algorithm:** An efficient depth-first search algorithm named CHUD (Closed^+ High Utility itemset Discovery) uses a vertical database to discover CHUIs. CHUD is an extension of DCI-Closed [4]; one of the currently best methods to mine closed itemsets. CHUD is adapted for mining CHUIs and include several effective strategies such as REG (Removing the Exact utilities of items from the Global TU-Table), RML (Removing the Mius of items from Local TU-Tables) and DCM (Discarding Candidates with a MAU that is less than the minimum utility threshold)for reducing the number of candidates generated in Phase I. Similar to the DCI-Closed algorithm; CHUD adopts an Itemset-Tidset pair Tree (IT-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). A data structure called transaction utility table (TU-Table) is adopted for storing the transaction utilities of transactions. It is a list of pairs <R, TU(T_R)> where the first value is a TID R and the second value is the transaction utility of T_R. Finally, Phase II is performed on these candidate itemsets to obtain all CHUIs.

CHUD algorithm

*Input:*D:the database; abs_min_utility;
*Output:*The complete set of CHUIs

01. InitialDatabaseScan(D)
02. RemoveUtilityUnpromisingItems(O,GTU)
03. **for each** item a_k € O **do**
04. { Create node N({ a_k })
05. CHUD_Phase-I(N({ a_k }),GTU,abs_min_utility)
06. REG_Strategy(g(a_k),GTU)}
07. CHUD_Phase-II(D,abs_min_utility)

- **DAHU Method:** A top-down method named DAHU (Derive All High Utility itemsets) for efficiently recovering all the HUIs and their absolute utilities from the complete set of CHUIs.

DAHU algorithm

*Input:*ML: the maximum length of itemset in HC;abs_min_utility;
HC= { HC_1, HC_2, ….., HC_{ML}}:the complete set of CHUIs;
*Output:*H: the complete set of HUIs

01. H_{ML}:= HC_{ML}
02. **for** (k:=ML-1;k>0;k--) **do**
03. { **for each** k-itemset X= {$a_1, a_2, \ldots a_k$ }€ HC_k**do**
04. { **if** (au(X) < abs_min_utility) **then delete** X **from** HC_k
05. **else** add X and its absolute utility au(X) to H.
06. { **for each** item a_i € X **do**
07. { Y:=X-{a_i}
08. au(Y):=au(X)-V(X,a_i)
09. **if**(au(Y) ≥ abs_min_utility) **then**

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 4, April 2016

10. { **if** Y € HC_{k-1} and SC(X) > SC(Y) **then**
11. { SC(Y):=SC(X)}
12. **else if** Y¢ HC_{k-1}**then**
13. { HC_{k-1}:= HC_{k-1} U Y
14. SC(Y):=SC(X) }}}}}}

The Fig. 1 shows the overall architecture of the proposed Closed^{+} High Utility Itemsets (CHUIs) representation.

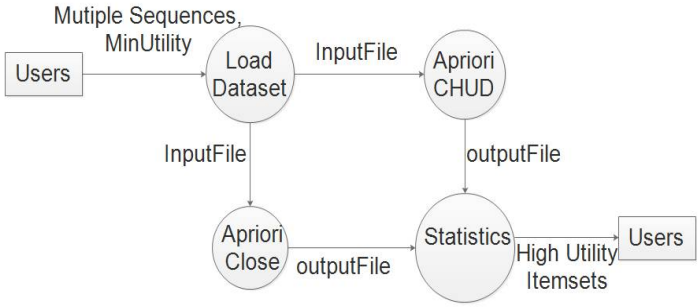

Fig. 1.Data Flow Diagram

IV. EXPERIMENTAL RESULTS

The performance of the proposed algorithms is evaluated by conducting experiments on Mushroom dataset obtained from FIMI Repository [8]. Fig. 2(a) shows the total execution time required by AprioriHC-D algorithm against Two-Phase algorithm [2] algorithm. Similarly Fig. 2(b) shows the total execution time required by both CHUD algorithm and DAHU method together against UP-Growth [3] algorithm.In Fig. 3 total memory usage by CHUD algorithm is depicted.

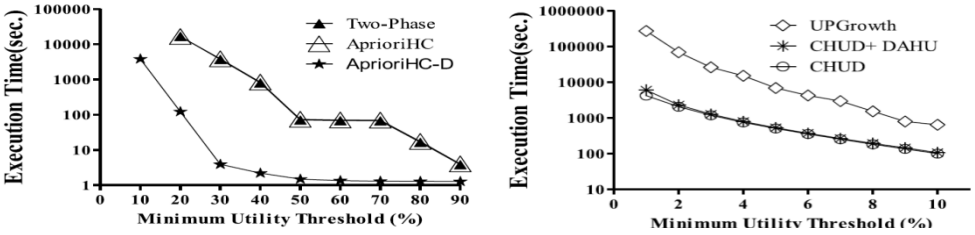

Fig. 2 Execution Time (a) Total Execution Time by AprioriHC-D (b) Total Execution Time by CHUD+DAHU

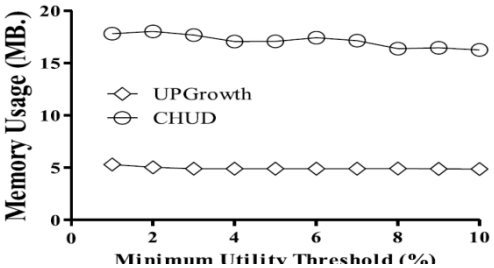

Fig. 3.Memory consumption

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 4, April 2016

V. CONCLUSION

In this paper, we addressed the problem of redundancy in high utility itemset mining by proposing alossless and compact representation named Closed$^+$ high utility itemsets, which has not been explored so far. To mine this representation, three efficient algorithms named AprioriHC (Apriori-based approach for mining High utility Closed itemset), AprioriHC-D (AprioriHC algorithm with Discarding unpromising and isolated items) and CHUD (Closed$^+$High Utility itemset Discovery). AprioriHC-D is an improved version of AprioriHC, which incorporates strategies DGU and IIDS 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 depth-first search for mining Closed$^+$ high utility itemsets from vertical database. The strategies incorporated in CHUD are efficient and novel. They have never been used for vertical mining of high utility itemsets and Closed$^+$high utility itemsets. To efficiently recover all high utility itemsets from CHUIs, an efficient method named DAHU (Derive All High Utility itemsets) has been proposed.

REFERENCES

[1] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in Proc. 20th Int. Conf. Very Large Data Bases, 1994, pp. 487–499.
[2] C.-W.Wu, B.-E. Shie, V. S. Tseng, and P. S. Yu, "Mining top-k high utility itemsets," in Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2012, pp. 78–86.
[3] V. S. Tseng, C.-W.Wu, B.-E.Shie, and P. S. Yu, "UP-Growth: An efficient algorithm for high utility itemset mining," in Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2010, pp. 253–262.
[4] C. Lucchese, S. Orlando, and R. Perego, "Fast and memory efficient mining of frequent closed itemsets", IEEE Trans. Knowl. Data Eng., vol. 18, no. 1, pp. 21–36, Jan. 2006.
[5] B.-E. Shie, V. S. Tseng, and P. S. Yu, "Online mining of temporal maximal utility itemsets from data streams, in Proc. Annu. ACM Symp. Appl. Comput., 2010, pp. 1622–1626.
[6] Y. Liu, W. Liao, and A. Choudhary, "A fast high utility itemsets mining algorithm," in Proc. Utility-Based Data Mining Workshop, 2005, pp. 90 - 99.
[7] C. Lucchese, S. Orlando, and R. Perego, "Fast and memory efficient mining of frequent closed itemsets," IEEE Trans. Knowl. Data Eng., vol. 18, no. 1, pp. 21–36, Jan. 2006.
[8] Frequent itemset mining implementations repository http://fimi.ua.ac.be/data/