

A Review on Efficient Algorithms for Mining Top-K High Utility Item Sets

Reshma Sodanwar*, Prof. Sachin Bere

Computer Engineering, SPPU Pune University, Pune, Maharashtra, India

ABSTRACT

High utility itemsets (HUIs) mining is most important topic in data mining. HUI mining means searching all itemsets whose utility is meeting with user-specified minimum utility threshold value i.e. min-util. However, Deciding min-util value approximately or exactly is found bit difficult to users, since finding an appropriate minimum utility threshold by trial and error is a tedious as well as time consuming process. Also problem with the same was if threshold value set too low, number of high utility itemset generated were too high, which in turn making the mining process ineffective. Whereas if min-util is set too high, it is likely that very less or no HUIs will be found. The proposed algorithm address the above issues by proposing a new framework for top-k high utility itemset mining, where k is the required number of HUIs to be mined. Two types of efficient algorithms named TKU (mining Top-K Utility itemsets) and TKO (mining Top-K utility itemsets in One phase) are proposed for mining such itemsets without the need to set threshold value by user. System provide comparison of the two algorithms with discussions on their advantages and limitations.

Keywords: Utility mining, high utility itemset mining, top-k pattern mining, top-k high utility itemset mining.

I. INTRODUCTION

Frequent pattern mining is a popular problem in data mining, which consists in finding frequent patterns in transaction databases.

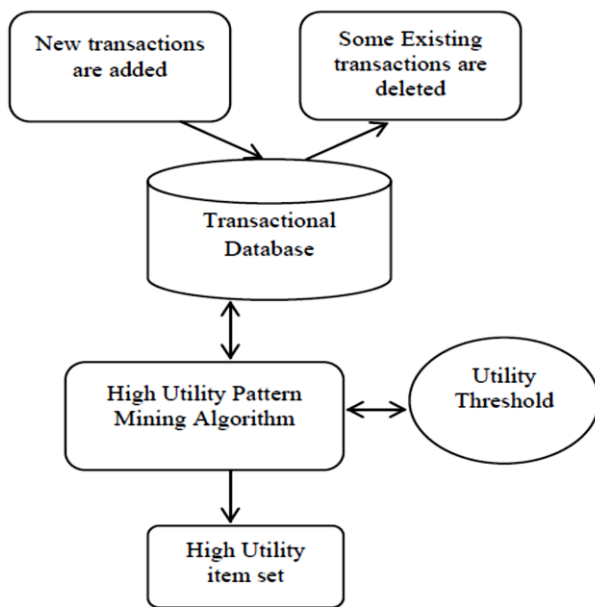


Figure 1 Architecture of utility item set mining

The traditional FIM may yield a large amount of frequent but low-value item sets and may lose the information on valuable item sets having low selling frequencies. Hence, it cannot satisfy the requirement of users who desire to discover item sets with high profits. Even, the association rule mining algorithm named apriori is used to find the candidate itemsets and then derive the frequent itemsets based on the minimum support value. The apriori used join and prune mechanism to find the itemsets. To address the issues of frequent mining, utility mining came into existence. In utility mining, each item is associated with a unit profit and the quantity of that item. An item set is called high utility item set (HUI) if its utility is no less than a user-specified minimum utility threshold min_util. Efficiently mining the high utility itemsets in databases is not an easy task because the downward closure property used in FIM does not hold for the utility of item sets. In other words, pruning search space for HUI mining is difficult because a superset of a low utility item set can be high utility. To tackle this problem, the concept of transaction weighted utilization (TWU) model was introduced. In this model, an item set is called high transaction-weighted utilization item

set (HTWUI) if its TWU is no less than min_util , where the TWU of an item set represents an upper bound on its utility. To precisely control the output size and discover the itemsets with the highest utilities without setting the thresholds, a promising solution is to redefine the task of mining HUIs as mining top-k high utility itemsets (top-kHUIs). The idea is to let the users specify k , i.e., the number of desired itemsets, instead of specifying the minimum utility threshold. Setting k is more intuitive than setting the threshold because k represents the number of itemsets that the users want to find whereas choosing the threshold depends primarily on database characteristics, which are often unknown to users. Using a parameter k instead of the min-util threshold is very desirable.

II. LITERATURE SURVEY

A. Existing Systems

1) An Algorithm for Mining Association Rules Using Perfect Hashing and Database Pruning. [1]:

In this method, propose an algorithm for finding frequent item sets in transaction databases. The basic idea of our algorithm is inspired from the Direct Hashing and Pruning (DHP) algorithm, which is in fact a variation of the well-known Apriori algorithm. In the DHP algorithm, a hash table is used in order to reduce the size of the candidate $k + 1$ item sets generated at each step. The difference of our algorithm is that, it uses perfect hashing in order to create a hash table for the candidate $k + 1$ item sets. As perfect hashing is used, the hash table contains the actual counts of the candidate $k + 1$ item sets. Hence we do not need to make extra processing to count the occurrences of candidate $k + 1$ item sets as in the DHP algorithm. The algorithm also prunes the database at each step in order to reduce the search space. We also tested our algorithm with real datasets obtained from a large retailing company and observed that our algorithm performs better than the Apriority algorithm. In this work, we studied the problem of finding frequent item sets for association rule mining. An algorithm called Direct Hashing and Pruning (DHP) is discussed in detail, and by using the ideas in the DHP algorithm, we propose a new algorithm PHP that employs the hashing facility of Perl5 in order to keep the actual count of occurrences of each candidate item set of the transaction database. The proposed algorithm also

prunes the transactions, which do not contain any frequent items, and trims the non-frequent items from the transactions at each step. Since our algorithm has less number of steps than the DHP algorithm, we did not compare the performance of these two algorithms. In order to test the performance of our algorithm, we compared it against an implementation of Apriority algorithm over the real dataset that was obtained from the Begendik Corporation. As the experimentation has showed, our algorithm performs better than the Apriority algorithm since at each step it reduces the database size to be scanned, and it generates much smaller sized C2 at the initial step.

2) Efficient Algorithm for Finding High Utility Item sets from Large Transactional Databases Using R-Hashing Technique [2]:

In this technique, proposed an efficient algorithm R-Hashing technique for mining high utility item sets from transaction databases. A data structure named UP-Tree is used for maintaining the information of high utility item sets. Though, the potential high utility item sets can be efficiently generated from the UP-Tree with only two scans of the database, proposed method decreases the scanning process. In the experiments, both of synthetic and real datasets are used to evaluate the performance of our algorithm. The mining performance is improved significantly since both the search space and the number of candidates are effectively reduced by the proposed strategies.

3) Mining Top-k Frequent Patterns in the Presence of the Memory Constraint [3]:

In this method, we have studied a practically important mining problem, namely mining top-k frequent/closed item sets in the presence of the memory constraint. To achieve this, we proposed the MTK/MTK-Close algorithms, which are devised as levelwise search algorithms based on an effective approach to constrain the number of candidates that will be generated-and-tested in each database scan. Since the minimum support to retrieve top-k frequent item sets cannot be known in advance, a novel search approach, called the -stair search, is devised in MTK and MTK-Close to efficiently retrieve top-k frequent/closed item sets. As demonstrated in the empirical study on real data and synthetic data, instead of only providing the flexibility of striking a compromise between the execution efficiency and the memory consumption, MTK can both achieve high efficiency and have a constrained memory bound, showing its prominent

advantage to be a practicable algorithm of mining frequent patterns.

4) UP-Growth: an Efficient Algorithm for High Utility Item set Mining [4]: In this system, proposed an efficient algorithm named UP-Growth for mining high utility item sets from transaction databases. A data structure named UP-Tree is proposed for maintaining the information of high utility item sets. Hence, the potential high utility item sets can be efficiently generated from the UP-Tree with only two scans of the database. Besides, we develop four strategies to decrease the estimated utility value and enhance the mining performance in utility mining. In the experiments, both of synthetic and real datasets are used to evaluate the performance of our algorithm. The mining performance is enhanced significantly since both the search space and the number of candidates are effectively reduced by the proposed strategies. The experimental results show that UP-Growth outperforms the state-of-the-art algorithms substantially, especially when the database contains lots of long transactions.

5) Novel Concise Representations of High Utility Item sets Using Generator Patterns [5]:

This method proposes a new framework for mining concise representations of high utility item sets using generators. We investigate the properties of generators and incorporate the concept of generator into HUI mining. We explore two new concise representations of HUIs, called High Utility Generator (HUG) and Generator of High Utility Item sets (GHUIs). Two efficient algorithms named HUG-Miner and GHUI-Miner are proposed to respectively mine these representations. The algorithms provide different trade-offs between execution time and completeness. GHUI-Miner captures the complete set of GHUIs but spends more time because it needs to consider generators that are not HUIs. On the other hand, HUG-Miner is over 100 times faster than GHUI Miner but misses GHUIs that are LUGs. Experimental results on both real-life and synthetic datasets show that the proposed algorithms are very efficient and achieve a massive reduction in terms of number of patterns found.

6) Fast Algorithms for Mining Interesting Frequent Item sets without Minimum Support [6]: In this method present two novel algorithms for mining interesting frequent item set (N-Most Miner and Top-K-Miner) using bit-vector representation approach, which is very efficient in terms of candidate item set

frequency counting. For projection we present a novel bit-vector projection technique PBR (projected-bit-regions), which is very efficient in terms of processing time and memory requirement. Several efficient implementation techniques of N-Most Miner and Top-K-Miner are also presented, which experienced in implementation.

7) Mining Correlated High-Utility Item sets using the Bond Measure[7]: This method proposed an algorithm named FCHM (Fast Correlated high-utility item set Miner) to efficiently measure. An extensive experimental study shows that FCHM is up to two orders of magnitude faster than FHM, and can discover more than five orders of magnitude less patterns by only mining correlated HUIs. The source code of algorithms and datasets can be downloaded as part of the SPMF open source data mining library.

8) Mining high on-shelf utility item sets with negative values from dynamic updated database [8]: In this system proposed an algorithm FUPTHOUIN for finding high on-shelf utility item sets with negative item values from dynamic updated database. In first phase, an algorithm builds utility tree by scanning original database. In second phase, it updates utility tree. It rescans an original database whenever necessary. Database is not rescanned for every modification which reduces the execution time of an algorithm. Whenever transactions are modified, utility tree is updated based on predefined four cases. Utility tree avoids unnecessary candidate item set generations and thus improves execution time.

9) Efficient Mining of High Utility Sequential Patterns Over Data Streams[9]: In this system proposed a novel framework for mining high utility sequential patterns over data a stream. Proposed a novel algorithm named HUSP-Stream to discover high utility sequential patterns in a transaction-sensitive sliding window over an item set-sequence stream. Two data structures named Item Util Lists and HUSP-Tree (High Utility Sequential Pattern Tree) are proposed to maintain the essential information of potential high utility sequences over data streams. When data arrive at or leave from the sliding window, HUSP-Stream incrementally updates HUSP-Tree and Item Util Lists online to find high utility sequential patterns based on previous mining results. We also defined a new over-estimated sequence utility measure named Suffix Utility (SFU), and used it to effectively prune the HUSP-Tree. Both real and synthetic datasets

are used to show the performance of HUSP-Stream. In the experiments, we compared HUSP-Stream with U Span, a state-of-the art algorithm for mining high utility sequential patterns in static databases.

10) Isolated items discarding strategy for discovering high utility item sets [10]: Traditional methods of association rule mining consider the appearance of an item in a transaction, whether or not it is purchased, as a binary variable. However, customers may purchase more than one of the same item, and the unit cost may vary among items. Utility mining, a generalized form of the share mining model, attempts to overcome this problem. Since the Apriority pruning strategy cannot identify high utility item sets, developing an efficient algorithm is crucial for utility mining. This study proposes the Isolated Items Discarding Strategy (IIDS), which can be applied to any existing level-wise utility mining method to reduce candidates and to improve performance. The most efficient known models for share mining are SHFSM and DCG, which also work adequately for utility mining as well. By applying IIDS to SHFSM and DCG, the two methods FUM and DCG+ were implemented, respectively. For both synthetic and real datasets, experimental results reveal that the performance of FUM and DCG+ is more efficient than that of SHFSM and DCG, respectively. Therefore, IIDS is an effective strategy for utility mining.

B. Comparison

1) **Existing System:** (FREQUENT item set mining is a fundamental research topic in data mining (FIM) mining. However, the traditional FIM may discover a large amount of frequent but low-value item sets and lose the information on valuable item sets having low selling frequencies. Hence, it cannot satisfy the requirement of users who desire to discover item sets with high utilities such as high profits. To address these issues, utility mining emerges as an important topic in data mining and has received extensive attention in recent years. In utility mining, each item is associated with a utility (e.g. unit profit) and an occurrence count in each transaction (e.g. quantity). The utility of an item set represents its importance, which can be measured in terms of weight, value, quantity or other information depending on the user specification. An item set is called high utility item set (HUI) if its utility is no less than a user-

specified minimum utility threshold min-util. HUI mining is essential to many applications such as streaming analysis, market analysis, mobile computing and biomedicine.

2) Disadvantages of Existing System:

1. Efficiently mining HUIs in databases is not an easy task because the downward closure property used in FIM does not hold for the utility of item sets.
2. In other words, pruning search space for HUI mining is difficult because a superset of a low utility item set can be high utility

C. Proposed System:

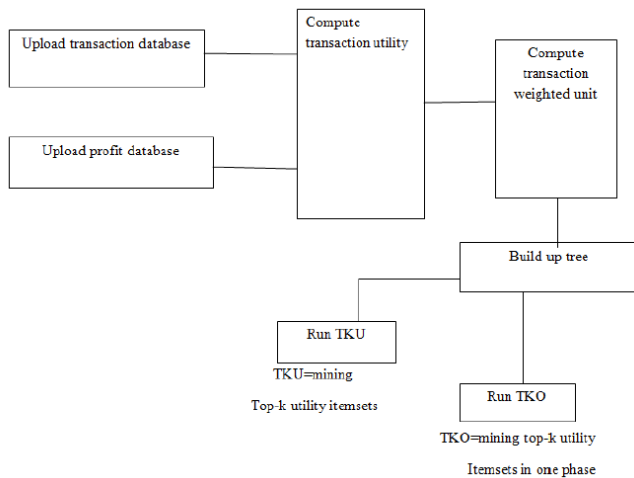
The concept of transaction weighted utilization (TWU) model was introduced to facilitate the performance of the mining task. In this model, an item set is called high transaction-weighted utilization item set (HTWUI) if its TWU is no less than min-util, where the TWU of an item set represents an upper bound on its utility. Therefore, a HUI must be a HTWUI and all the HUIs must be included in the complete set of HTWUIs. A classical TWU model-based algorithm consists of two phases. In the first phase, called phase I, the complete set of HTWUIs are found. In the second phase, called phase II, all HUIs are obtained by calculating the exact utilities of HTWUIs with one database scan.

D. Advantages of Proposed System:

1. Two efficient algorithms named TKU (mining Top-K Utility items ets) and TKO (mining Top-K utility item sets in one phase) are proposed for mining the complete set of top-k HUIs in databases without the need to specify the min-util threshold.
2. The construction of the UP-Tree and prune more unpromising items in transactions, the number of nodes maintained in memory could be reduced and the mining algorithm could achieve better performance.

III. SYSTEM ARCHITECTURE

System Architecture



The System Architecture depicts the overall flow of proposed system to find out HUI using top-k framework mechanism. Basically proposed system is using two algorithms TKU & TKO for mining HUI.

A. Read Transaction Database: Using this module we are uploading entire transaction file to the application, each transaction is in separate line and each item is separated with comma.

B. Read Profit Database:

Using this module we are uploading profit table where each line represent Item name and its profit, separated with comma.

C. Compute TU (Transaction Unit):

Using this module we are calculating total profit for each transaction.

D. Compute TWU:

Using this module application will calculate entire profit for each item exists in all **transaction**.

E. Build UP tree:

Application will generate up tree from generated TWU itemsets.

F. TKU and TKO:

Run code to generate TKU (mining top-k utility items sets) and TKO (generate top k utility item sets in one phase)

IV. CONCLUSION

In this system studied the problem of top-k high utility itemsets mining, where k is the desired number of high utility itemsets to be mined. Two efficient algorithms TKU (mining Top-K Utility itemsets) and TKO are proposed for mining such itemsets without setting minimum utility thresholds. TKU is the first two-phase algorithm for mining top-k high utility itemsets., which incorporates five strategies PE, NU, MD, MC and SE to effectively raise the border minimum utility thresholds and further prune the search space. On the other hand, TKO is the first onephase algorithm developed for top-k HUI mining, which integrates the novel strategies.

V. REFERENCES

- [1]. R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proc. 20th Intl Conf. Very L 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.
- [2]. K. Chuang, J. Huang, and M. Chen, "Mining top-k frequent patterns in the presence of the memory constraint," VLDB J., vol. 17, pp. 1321-1344, 2008.
- [3]. R. Chan, Q. Yang, and Y. Shen, "Mining high-utility itemsets," in Proc. IEEE Int. Conf. Data Mining, 2003, pp. 1926.
- [4]. Fournier-Viger and V. S. Tseng, "Mining top-k sequential rules," in Proc. Int. Conf. Adv. Data Mining Appl., 2011, pp. 180-194.
- [5]. P. Fournier-Viger, C. Wu, and V. S. Tseng, "Mining top-k association rules," in Proc. Int. Conf. Can. Conf. Adv. Artif. Intell., 2012, pp. 6173.
- [6]. P. Fournier-Viger, C. Wu, and V. S. Tseng, "Novel concise representations of high utility itemsets using generator patterns," in Proc. Int. Conf. Adv. Data Mining Appl. Lecture Notes Comput. Sci., 2014, vol. 8933, pp. 3043.
- [7]. J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in Proc. ACM SIGMOD Int. Conf. Manag. Data, 2000, pp. 112.
- [8]. J. Han, J. Wang, Y. Lu, and P. Tzvetkov, "Mining top-k frequent closed patterns without minimum support," in Proc. IEEE Int. Conf. Data Mining, 2002, pp. 211-218.
- [9]. S. Krishnamoorthy, "LP pruning strategies for mining high utility itemsets," Expert Syst. Appl., vol. 42, no. 5, pp. 2371-2381, 2015.