Scientific Journal of Impact Factor (SJIF): 4.72

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 4, Issue 3, March -2017

Implementation of Algorithm for Finding Top-k High Utility Item sets

D. Swapna[1] , P. Hima Bindu[2], P. Sri Ramya[3], A. Siri Chandana[4]

[1]*Assistant Professor, Computer Science and Engineering, BVRIT Hyderabad College of Engineering for Women*
[2,3,4] *Computer Science and Engineering, BVRIT Hyderabad College of Engineering for Women*

Abstract — *High utility item sets (HUIs) mining is an emerging technique in data mining. It helps in discovering all item sets having a utility meeting a user-specified minimum utility threshold min_util. However, setting min_util appropriately is a difficult problem for users. Finding an appropriate min_util by trial and error is a difficult process for users. If min_util is set too low, too many itemsets will be generated, which takes large search space and may cause the mining process to be very inefficient. On the other hand, if min_util is set too high, it is likely that no HUIs will be found. To overcome this we have two phase mining techniques in which scalability and efficiency are bottleneck problems. To solve this, we use an algorithm named TKO(Top-k utility itemsets in one phase) in which the high utility itemsets are generated in one phase. It makes use of utility-list structure. It yields the top k utility itemsets where k is the user specified value.*

Keywords- *Frequent mining, Association mining, High utility itemset mining, Top-k high utility mining, Utility mining.*

I. INTRODUCTION

Frequent item set mining (FIM) is a fundamental research topic in data 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.

Depending on the threshold value, the search space can be very small or very large. Besides, the choice of the threshold greatly influences the performance of the algorithms. If the threshold is set too low, many high utility itemsets are generated and it is difficult for the users to comprehend the results. A large search space makes mining algorithms inefficient or even run out of memory, because the more HUIs the algorithms generate, the more resources they consume. On the contrary, if the threshold is set too high, no HUI will be found. To find an appropriate value for the min_util threshold, users need to try different thresholds by guessing and re-executing the algorithms over and over until being satisfied with the In this paper, we address all of the above challenges by proposing a novel framework for top-k high utility item set mining, where k is the desired number of HUIs to be mined. Top-K utility mining in One phase(TKO) is proposed for mining the complete set of top-k HUIs in databases without the need to specify the min_util threshold. The TKO algorithm uses a list-based structure named utility-list to store the utility information of itemsets in the database. It uses vertical data representation techniques to discover top-k HUIs in only one phase. To reduce the search space in TKO, strategies such as RUZ, RUC and EPB are merged together. RUC is Raising the threshold by Utility of Candidates. This strategy is concerned with any kind of one phase algorithm which have item set with their utility.

II. EXISTING SYSTEM

The frequent itemset mining techniques and association rule mining techniques which is being used yields a large set of frequent and low-value itemsets which may result in low profit.

In the existing system, 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.In other words, pruning search space for HUI mining is difficult because a superset of a low utility itemset can be high utility.

III. PROPOSED SYSTEM

In this paper, 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.

3.1. Advantages of proposed system

- The proposed algorithm has less search space so it needs less memory.
- It scans the database only once .
- It is easy to implement.
- It's performance is good in dense databases.

IV. IMPLEMENTATION

In order to perform mining in one phase we are using TKO algorithm. The TKO algorithm uses the utility list structure. This algorithm scans the database only once and hence requires less memory compared to the present algorithms. It uses HUI-Miner and its utility-list structure, which is a basic search procedure. When TKO generates the item set, without scanning the database the utility-list calculates the utility. Initially the threshold is set to zero.

4.1. Utility mining

In this, we will discuss about utility-list structure and its properties. In TKO algorithm, each itemset is linked with a utility-list. This utility-lists of items are constructed by searching the database twice. This is called is initial utility-lists. In its first search, utility values of items and Transaction Weighted utilities(TWU) are computed. In its second search, according to TWU values items are sorted in each transaction and each items utility-list is built. The utility-list of an item consists of one or more tuples. Each tuple represents the information in a transaction Tr . It has three fields. They are Tid, iutil and rutil. Fields Tid and iutil contains the identifier of Tr and the utility in Tr. Field rutil indicates the remaining utility in Tr. It follows a strategy to raise the minimum border utility. The itemsets whose utility is more than mininum border utility is the high utility itemset which has more profit.
Let us consider an example and find the utility list. Let us consider a table consisting of six items A,B,C,D,E,F and 8 transactions. Each transaction represents the quantity of that particular item bought. Here, in transaction1 the item A's quantity rate is 3, B's 2 and so on.

Table 1. Transaction table

.

Tid	A	B	C	D	E	F
1	3	2	0	3	0	0
2	2	0	0	4	2	0
3	3	0	5	0	0	3
4	1	0	3	0	1	2
5	1	0	0	3	2	0
6	1	2	0	4	0	0
7	2	3	2	0	1	1
8	0	0	0	0	0	1

Let us consider a profit table, each item in the transaction has a profit value.

Table 2. Profit table

Tid	A	B	C	D	E	F
Profit	4	150	1	60	100	20

Now, let us build a utility list. The first column is the transaction id of that particular itemset. The second column is the Iutility which is the product of the quantity rate and profit rate of that particular itemset in that particular transaction. The third column is the Rutility value which is calculated by multiplying the quantity and profit values of the remaining unprocessed items.

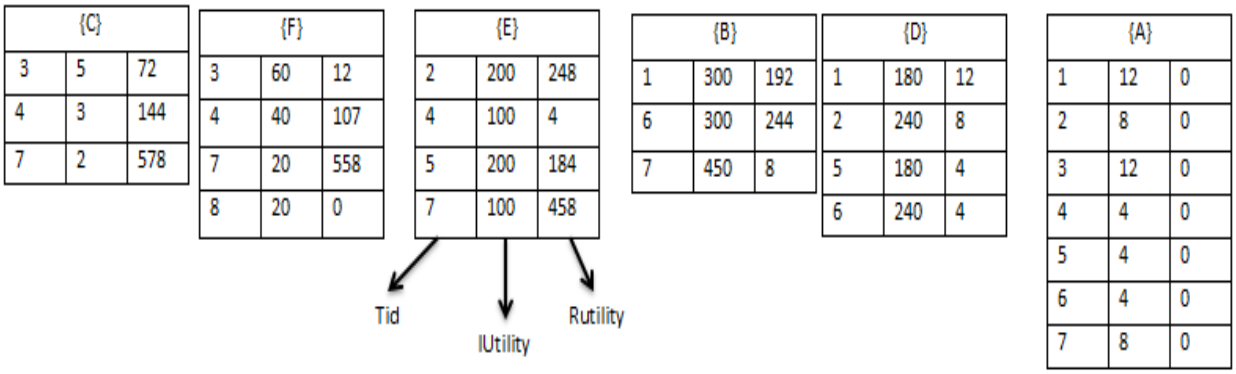

Fig 1. Utility list

In the construction process, the itemsets are sorted in ascending order of their transaction-weighted utility (TWU). For the Rutility of an itemset in a transaction, it keeps the rest utilities in the transaction except the processed itemset . Since the TWU values of the itemsets are changed with transaction insertion, the sorted order of the utility-list structures and the Rutility value should also be changed. The number of inserted transactions is, however, very small compared to the original database. In the proposed algorithm, the sorted order of the itemsets in the inserted transactions follows the initially TWU ascending order of itemsets in the original database. Based on the raised border utility, the top itemsets are retreived.

By comparision, the apriori algorithm uses join and prune technique and takes large amount of memory space as it generates large set of candidate keys. The execution time is more as it needs to generate large set of candidates. Comparitively, TKO uses utility list structure and scans database only once. TKO needs less memory. The execution time is also less.

V. CONCLUSIONS

In conclusion, the TKO algorithm derives high utility itemsets by using utility list structure.With the help of user specified k value the utility thresholds are considered. When TKO generates the itemset, without scanning the database the utility-list calculates the utility. Thus, this algorithm takes less memory and easy to implement.

REFERENCES

[1] Vincent S Tseng, Bai-En Shie, Cheng-Wei Wu, and S Yu Philip.Efficient algorithms for mining high utility itemsets from transactional databases. IEEE transactions on knowledge and data engineering.
[2] Junqiang Liu, Ke Wang, and Benjamin CM Fung. Mining high utility patterns in one phase without generating candidates. IEEE Transactions on Knowledge and Data Engineering, 28(5):1245–1257, 2016.
[3] R. Chan, Q. Yang, and Y.-D. Shen, “Mining high utility itemsets,” in Proceedings of the 3rd IEEE International Conference on Data Mining , pp. 19–26, IEEE, November 2003.
[4] H. Yao and H. J. Hamilton, “Mining itemset utilities from transaction databases,” Data and Knowledge Engineering, vol. 59, no. 3, pp. 603–626, 2006.
[5] M. Liu and J. Qu, “Mining high utility itemsets without candidate generation,” in Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM '12), pp. 55–64, November 2012.
[6] C. W. Wu, B.-E. Shie, V. S. Tseng, and P. S. Yu, “Mining top-K high utility itemsets,” in Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '12), pp. 78–86, August 2012.
[7] A. Erwin, R. P. Gopalan, and N. R. Achuthan, “Efficient mining of high utility itemsets from large datasets,” in Proc. Int. Conf. Pacific- Asia Conf. Knowl.Discovery Data Mining, 2008.IEEE- Vol. 27, No. 3, March 2015.

[8] A. Savasere, E. Omiecinski, and S. B. Navathe, "An efficient algorithm for mining association rules in large databases," in Proc. 21st Int. Conf. Very Large Databases, 1995.
[9] 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. Knowl. Data Eng., vol. 21, no. 12, pp. 1708– 1721, Dec. 2009.
[10] V. S. Tseng, B.-E. Shie, C.-W. Wu, and P. S. Yu, "Efficient algorithms for mining high utility itemsets from transactional databases," IEEE Trans. Knowl. Data Eng., vol. 25, no. 8, pp. 1772–1786, Aug. 2013.

