International Journal of Current Trends in Engineering & Technology
Volume: 02, Issue: 01 (JAN-FAB, 2016)

A Review on Concise and Lossless Representation of High Utility Item sets with CHUD Algorithm

Abhijit P. Ingale[1], Prof Kailash Patidar[2], Assistant Prof. Megha Jain[3]
apingale83@gmail.com[1], kailashpatidar123@gmail.com[2], 06meghajain@gmail.com[3]
Sri Satya Sai Institute of Science and Technology, Sehore, Madhya Pradesh, India

Abstract — Planning and designing of business strategies for today businesses became really a crucial process, because of the sale of more than hundreds of items in a single transaction. This can be realized by analyzing the sale databases of super shops, Big bazaars, mauls etc. Mining of frequent Item-sets is necessary to improve the business by maximizing the profit and minimizing the loss. Classic Apriori algorithm [1-10] only considers the presence of an item in transactions. It may be possible that item with low profit may occur in maximum transactions and item with high profit may occur in less number of transactions. In such cases, frequency of low profit items makes them to appear in frequent Item-sets, while items with higher profit will be listed under infrequent item-sets. In such cases we can predict an item's significance in our business. Another drawback is generation of large number of frequent patterns in result and unnecessary joining of large number of frequent item-sets at each step. This makes results of Apriori inaccurate and clumsy. Solution to this problem is CHUD (Closed High Utility Item-sets Discovery) [1] algorithm with association matrix. CHUD finds the closed item-sets with high utility by considering an item sets items quantity or units in the transaction as well as the profit unit of those items in item-set. Association matrix will hold the presence or absence of that item in each item. Further, a method called DAHU (Derive All High Utility Item-sets) [1] is applied to recover all HUIs from the set of CHUIs without accessing the original database.

Keywords: - closed high utility item sets, utility Mining, Concise and lossless representation, data mining.

INTRODUCTION

Data mining is the process of mining non-trivial, formerly extraordinary, previously unknown and potentially valuable information from large databases [2-10]. It is also concerned with analysis of large amount of data to discover interesting regularities or relationships which in turn leads to better understanding. Thus data mining refers to extracting or mining knowledge from large amounts of data. Data mining activities uses combination of techniques from database technologies, artificial intelligence, machine learning etc and the application areas which include this are includes bioinformatics, genetics, medicine, clinical research, education, retail and marketing research. Frequent item set mining is a fundamental research topic in data mining. Frequent item sets are the item sets that appear frequently in the transactions. The goal of frequent item set mining is to identify all the item sets in a transaction dataset which occurs frequently. From previous few years frequent item set mining became a crucial task and gains an attention of maximum professionals for the betterment. It is really tedious to search an interesting pattern when we got high number of results in search. Frequent item set mining supports us to find item sets which are collection of items frequently sold together. Classic APRIORI algorithm helps us to find frequent item sets, but there is no strategy to consider the profit units for the business products. Hence the problem "frequency dominates significance" occurs. Here we are using the word significance in terms of the profit or some weight associate with that item. Profit can be thought of as a margin in rupees per unit, or the overall percentage of profit per bunch. To avoid problem of "frequency dominates significance" we can use utility

mining than frequency mining. In utility mining each item has a unit weight and can appear more than once in a transaction. The term utility refers to the importance or the usefulness of the appearance of the item set in the transaction quantified in terms of profit, sales or any other user preference. A transaction database consists of two measures such as internal utility and external utility. Quantity of a product present in a particular transaction is called the internal utility and the profit value of a product in each transaction is called external utility. In utility mining to put frequency and significance measure in a single unit concept of transaction weighted utility can be introduced. Which is a summation of transaction utilities in which item is appearing. Transaction utility can be computed as the summation of the product of internal utility and external utility of each item present in the transaction.

An item set can be said as a promising item set if it satisfies the minimum threshold otherwise listed under unpromising or less significant category. CHUD algorithm will provide a concise and lossless representation of closed high utility item sets. This suggests that item sets with high frequency low profit and item sets with low frequency and high profit both will be discovered guaranteed. Here frequency is a support count of an item set in transaction. Closed high utility item sets do not have any superset which will satisfy the minimum threshold. Hence all the supersets of closed item sets discarded. CHUD prunes the search space very efficiently and outputs the concise set of closed item sets.

Searching strategy in CHUD incurs heuristic information to prune the search space. Classic Apriori algorithms considers each item set in set of frequent item sets in 'k' pass, and joins that set with every other set to form the candidate set for next pass 'k+1'. Instead CHUD finds the closure of item set, tests it against minimum threshold, if satisfies the test, records them in output set and explores the closure further. CHUD Algorithm is loaded with strategies like REG, RML, and DCM which greatly enhance the performance of algorithm as well as quality of results.

Algorithm DAHU (Discovery of all high utility item set) [1] later on takes over and discovers all possible high utility item sets on the output of CHUD algorithm. In this every subset of each closed item set will be tested with different measure that is an absolute utility of an item set, if it's absolute utility satisfying minimum threshold and not present in CHUD Output set that will be recorded. Finally we get the rich set of utility item sets.

Before application of CHUD and DAHU we need to convert the dataset vertically. That means most of the transactions in the database are stored horizontally like

Table 1 (A) (B) Horizontal Transactions

TransID	Item_Sold	....	...	...	...
TID1	*I1(3),I2(2),I3(1)*	....	...	...	...
TID2	*I1(2),I2(1),I5(4),I4(3)*	....	...	...	...
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

OR

TransID	Item_Sold	Quantity	...	...	...
TID1	*I1*	*3*	...	...	...
TID1	*I2*	*2*	...	...	...
TID1	*I3*	*1*	...	...	...
TID2	*I1*	*2*	...	...	...
TID2	*I2*	*1*	...	...	...
TID2	*I5*	*4*	...	...	...
TID2	*I4*	*3*	...	...	...

Vertical representation of above dataset can viewed as

Table 2 Vertical Transactions

TransID	I1	I2	I3	I4	I5
TID1	3	2	1	0	0
TID2	2	1	0	3	4
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

PRELIMINARY DEFINITIONS:

1. Let $I=\{I_1,I_2,I_3,I_4,I_5, \ldots,I_n\}$ is a finite and complete set if items in the our business databases. D is a data set of transactions containing information of items sold with quantity, where $D= \{T_1, T_2 \ldots T_n\}$. $P=\{p_1,p_2,p_3,p_4,p_5,\ldots p_n\}$is a set of profit units

associated with each item in set I, that is profit(I1)=p1, Profit(I2)=p2, ….and so on.

2. **Internal Utility [1]:** Internal utility of an item is the quantity of an item in that transaction. We denote it with q (I_i, Tr). For example q (I_4, T_5) represents item-4 quantity in transaction T_5.
3. **External Utility [1]:** External utility of an item is a profit unit associated with that item or product, and represented as P (I_1). This can be referring from the profit unit table in database or set P mentioned above.
4. **Item set of k length [1]:** An item set of k length can be defined as the set of distinct items which is a subset of I. i.e. X= {I_1, I2, I3….,I_k} such that X ⊆ I.
5. **Support count of an Item set:** Support of an item or Item set can be explained as the count of number of transactions in which item or item set is appearing. Support count of an item set X is denoted as SC(X).
6. **Support of an Item set [1]:** The ration of number transactions in which item set is appearing to the total number of transactions in D.
7. **Absolute utility of an item in transaction [1]:** Absolute utility of an item is the product of its internal utility in transaction and external utility.ie. p(I_k, D) * q(I_k, T_r).
8. **Absolute utility of an Item set in transaction [1]:** Absolute utility of an item set is the sum of absolute utilities of all the items present in the transaction. au(X, T_r)=$\sum_{Ik \in X}$ au(I_k,T_r).
9. **Absolute utility of item set in database[1]:** An absolute utility of an item set in D can defined as the sum of absolute utilities of item set in all the transactions of database in which it is appearing. au(X)=$\sum_{X \subseteq Tr \wedge Tr \in D}$ au(I_k,T_r).
10. **High Utility Item set [1]:** An Item set called as high utility item set if au(X) >= min_utility threshold.
11. **Transaction Weighted Utility (TWU) [1]:** TWU of an item set is the sum of transaction utilities of all the transactions containing that item set**.**

TWU(X)=$\sum_{X \subseteq Tr \wedge Tr \in D}$ TU(Tr).

12. **High Transaction Weighted Utilization Item set(HTWUI) [1]:** An item set X can be said as HTWUI if TWU(X)>=min_utility.
13. **Closure of an Item set [1]:** Closure of an item set X is denoted as C(X) and defined as the largest item set Y∈L such that X is a subset of Y (X⊆Y) SC(X)=SC(Y).
14. **Closed Item set [1,10]: An item set X∈L** is a closed item set if there exists no item set Y which is a *proper subset* of X and SC(X)=SC(Y).

CHUD ALGORITHM

To provide compact & concise representation [12] of high utility itemsets an algorithm CHUD [1] works efficiently because of it provides us with limited number of itemsets and considers quantity and profit unit of an item or item set. This discovers cream layer of high utility itemsets as well as there will be no loss of information.

Algorithm CHUD [1]

[This algorithm takes dataset D and absolute minimum utility abs_min_utility threshold as input. And output complete set of High Utility Itemsets.]

Initial_Database_Scan(D)

RemoveUtiltiyUnpromisingItems()

For each item Ik∈O do

{

Create_Node N({I_k})

CHUDPhase-I(N({I_k}),GTU,abs_min_utility)

ApplyREG(g({I_k}),GTU)

CHUDPhase-II(D, abs_min_utility)

}

As we know data for transactions or records are maintained horizontally in the database represented in table 1. CHUD first scans the database and convert it into vertical from shown as in table 2. Remove Utility Unpromising Items will discard an unpromising items and there utilities. An unpromising item set's utility will not affect our calculation of estimated utility. Once all this is done we got the

promising items. Then we apply a certain order on these items and sort them, for example, increasing order of their support. Here CHUD will consider each item in the ordered list. It creates a node with an item I_j at position 'j' in ordered list along with PREVSET, POSTSET and tidSet. Then all elements at position less than 'j' will be maintained in the PREVSET(I_j) and all the elements at positions greater than 'j' will be maintained in POSTSET(Ij). The tidSet data structure will hold the set of TIDs (i.e. Transaction IDs) in which item is appearing. This algorithm will work in bottom up approach.

Procedure CHUDPhase-I [1]

[This procedure takes node N(X), GTU, abs_min_utility as input from Algorithm CHUD. Outputs the complete set of PCHUI's]

```
{
If (Subsume_Check(N(X),PREVSET(X))==false)
{
        Xc=Compute_Closure(N(X),POSTSET(X))
        ApplyDCMStartegy(Xc)
        Explore(N(Xc),TU,min_abs_utility)
}
}
```

Procedure Subsume_check(N(X),PREVSET(X)) [1]

[This procedure takes node N(X), PREVSET(X) and returns Boolean value true or false. The return value will be decided as if the current item set X appearing in all transactions in which PREVSET(X) element is appearing then return true else false.]

```
{
For each item a ∈ PREVSET(X)
{
        If(g(X) ⊆ g(a))
        {
                Stop loop & Return true
        }
}
}
Return false
```

This procedure checks that if current item set's transaction set is a subset of transaction set of PREVSET element of current item set. An item set need not to be discovered if it's PREVSET contains an element which has a transaction set

Procedure Compute_Closure [1]

[This procedure takes N(X) and POSTSET(X) as input and returns the closure of X]

```
{
        Xc=X
For each item I∈POSTSET(X)
{
        If(g(X) ⊆g(I))
        {
                Remove 'I' from POSTSET(X)
                Xc=Xc ∪ I
        }
}
Return Xc
```

Compute_Closure will explore supersets of current X item/item set. It joins POSTSET elements of X to current item set, if the tidset of X is a subset of tidSet of POSTSET element of X. In short POSTSET element of X will be added to Xc if it is appearing in all the transactions in which X is appearing.

Procedure Explore [1]

[This procedure takes node N(X), local transaction utility table, and abs_min_utility. Outputs the set PCHUIs containing X]

```
For each item I_k ∈ POSTSET(X)
{
Remove I_k from POSTSET(X) and hold in memory.
        Create_Node N(Y)
        Set Y=X ∪ {I_k}
        Set g(Y)=g(X) ∩ g(I_k)
```

Initialize:

POSTSET(Y)=POSTSET(X)

PREVSET(Y)=PREVSET(X)

Estu(Y)=CalculateESTUtility(g(Y),TU_x)

If((Estu(X) and Estu(Y))>=abs_min_utility)

{

CHUDPhase-I(N(Y),TUx,abs_min_utility)

PREVSET(X)=PREVSET(X) ∪ {I_k}

}

ApplyRMLStrategy(g(Y),TU_x)

}

Procedure *Explore* explores the search space of closed itemsets by joining POSTSET element one by one to X. The strategies that help an algorithm to work efficiently, and prune the search space by discarding unpromising itemsets. Those important strategies are listed below.

Strategy 1: Ignore or discard unpromising items OR Consider only promising items [1]: In this strategy utilities of unpromising items discarded from the database and its absolute utility should be subtracted from the TU(Tr).

Strategy 2: IIDS (Isolated Items Discarding Strategy) [1]: Discard isolated items and their actual utilities from transactions and transaction utilities of the database.

Strategy 3: REG (Removing the Exact utilities of items from the Global TU-Table) [1]: This strategy will discard utilities of an item I_k from ordered list O each time when I_k processed completely, and next item is taken as new current node.

Strategy 4: RML (Removing the Minus of items from Local TU-Tables) [1]: This strategy works in *Explore* procedure, each time when an item from the POSTSET of current node gets processed. The Minimum item utility removed from the LTU (Local Transaction Utility) table. A minimum utility of an item is defined as the miu(X, T_r) such that there is no transaction T_s present that have u(X, T_s) < u(X, T_r) .

Strategy 5: DCM (Discarding Candidates with a MAU) [1]: This strategy states that a candidate XC can be discarded from Phase-II if it's estimated utility Estu (Xc) or mau (Xc) is less than abs_min_utility. Maximum item utility of item is defined as mau(X,Tr) such that there is no transaction Ts present that have u(X,T_s) > u(X,T_r). XC is output with its estimated utility if it's Estu(Xc) or mau(X_c) >= abs_min_utility otherwise discarded.

CONCLUSION

A person who likes to analyze, wants the accurate but compact result set at the end. Hence, only the creamy or rich result set must be expected at the output end, which makes user comfortable. CHUD algorithm discovers all closed high utility itemsets efficiently. Such efficiency of CHUD comes from the very good strategies (1, 2, 3, 4, 5) works with it. One can easily understand that, FIM's best is its downward closure property, which prunes search space. But in utility mining we cannot apply the same. TWU will be used to prune the search space along with strategies mentioned above. An important fact is that there will be less number of items set present which have exactly same utility as their supersets have. This will prune the search space. CHUD had one more benefit that the numbers of joins it have to perform are less. POSTSET will help to find the new items which are not processed yet and join them to current node and form a new item set. And PREVSET will help to check whether an item is already processed. Hence the redundancy of processing the same node will be drastically reduced. Overall CHUD will work fine to discover closed high utility itemsets.

REFERENCE

[1]. Vincent S. Tseng, Cheng-Wei Wu, Philippe Fournier-Viger, And Philip S. Yu, "Efficient Algorithms For Mining The Concise And Lossless Representation Of High Utility Item sets", IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 27, NO. 3, MARCH 2015, Pp.726-739.

[2]. R. Agrawal And R. Srikant, "Fast Algorithms For Mining Association Rules", In Proc. 20th Int. Conf. Very Large Data Bases, 1994, Pp. 487–499.

[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 Data Engineering, Vol. 21, No. 12, Pp. 1708– 1721, Dec. 2009.

[4]. J.-F. Boulicaut, A. Bykowski, And C. Rigotti, "Free-Sets: A Condensed Representation Of Boolean Data For The Approximation Of Frequency Queries," Data Mining Knowledge Discovery, Vol. 7, No. 1, Pp. 5–22, 2003.

[5]. T. Calders And B. Goethals, "Mining All Non-Derivable Frequent Item sets," In Proc. Int. Conf. Eur. Conf. Principles Data Mining Knowledge Discovery, 2002, Pp. 74–85.

[6]. 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.

[7]. R. Chan, Q. Yang, And Y. Shen, "Mining High Utility Item sets," In Proc. IEEE Int. Conf. Data Min., 2003, Pp. 19–26.

[8]. A. Erwin, R. P. Gopalan, And N. R. Achuthan, "Efficient Mining Of High Utility Item sets From Large Datasets," In Proc. Int. Conf. Pacific-Asia Conf. Knowledge Discovery Data Mining, 2008, Pp. 554–561.

[9]. K. Gouda And M. J. Zaki, "Efficiently Mining Maximal Frequent Item sets," In Proc. IEEE Int. Conf. Data Mining, 2001, Pp. 163–170.

[10]. T. Hamrouni, "Key Roles Of Closed Sets And Minimal Generators In Concise Representations Of Frequent Patterns," Intellectual Data Anal., Vol. 16, No. 4, Pp. 581–631, 2012.

[11]. J. Han, J. Pei, And Y. Yin, "Mining Frequent Patterns Without Candidate Generation," In Proc. ACM SIGMOD Int. Conf. Manage. Data, 2000, Pp. 1–12.

[12]. T. Hamrouni, S. Yahia, And E. M. Nguifo, "Sweeping The Disjunctive Search Space Towards Mining New Exact Concise Representations Of Frequent Item sets," Data Knowledge Eng., Vol. 68, No. 10, Pp. 1091–1111, 2009.

[13]. H.-F. Li, H.-Y. Huang, Y.-C. Chen, Y.-J. Liu, And S.-Y. Lee, "Fast and Memory Efficient Mining of High Utility Item sets In Data Streams," In Proc. IEEE Int. Conf. Data Mining, 2008, Pp. 881–886.