ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 2, Issue 4, pp: (334-340), Month: October - December 2014, Available at: www.researchpublish.com

High Utility Item sets Mining from Transactional Databases

[1]B.JAYANTHI, [2]S.SARASWATHI

[1]Associate Professor, [2]Research Scholar
[1, 2] Department Of Computer Science (P.G)
[1, 2] Kongu Arts and Science College, Erode, Tamil Nadu, India

Abstract: **Mining high utility item sets from transactional databases refers to finding the item sets with high profits. Here, the meaning of item sets utility is interestingness, importance, or profitability of an item to users. A large number of candidate item sets for high utility item sets degrades the mining performance in terms of execution time and space requirement. In this thesis, an algorithm, namely utility pattern growth (UP-Growth) is used for mining high utility item sets with a set of effective strategies for pruning candidate item sets. The information of high utility item sets is maintained in a tree-based data structure which is named as utility pattern tree (UP-Tree) such that candidate item sets can be generated efficiently with scanning the database. To facilitate the mining performance and avoid scanning original database repeatedly, a compact tree structure, named UP-Tree is used, to maintain the information of transactions and high utility item sets. EEGU (Enhanced Eliminating Global Unpromising items) and EEGNU (Enhanced Eliminating Global Node Utilities) strategies are applied to minimize the overestimated utilities stored in the nodes of global UP-Tree. By applying strategy EEGU (Enhanced Eliminating Global Unpromising items), the utilities of the nodes that are closer to the root of a global UP-Tree are further improved. Experimental results show that the proposed algorithms, especially UP-Tree with EEGU and EEGNU, reduce time complexity rate. It also outperform in the dynamic transaction dataset.**

Keywords: **frequent item set, high utility item set, utility mining, data mining.**

1. INTRODUCTION

The main purpose of this research is used to eliminate time complexity rate in tree construction process by using two strategies, namely EEGU (Enhanced Eliminating Global Unpromising items) and EEGNU (Enhanced Eliminating Global Node Utilities). So the time required to run the project is faster compared to existing research. It is mainly used in retail marketing and network logs etc.

TABLE 1 Example Database

TID	Transaction	TU
T1	(cz3,1) (dolo650,10) (eldoper,1)	17
T2	(cz3,2) (dolo650,6) (amlong,2) (concor,5)	27
T3	(cz3,2)(tusq,2) (eldoper,6) (amlong,2) (zincovit,1)	37
T4	(tusq,4) (dolo650,13) (eldoper,3) (amlong,1)	30
T5	(tusq,2) (dolo650,4) (amlong,1) (concor,2)	13
T6	(cz3,1) (tusq,1) (dolo650,1) (eldoper,1) (dianil,2)	12

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 2, Issue 4, pp: (334-340), Month: October - December 2014, Available at: www.researchpublish.com

TABLE 2 profit table

Item	Cz3	tusq	Dolo650	eldoper	Amlong	Zincovit	Concor	dianil
Profit	5	2	1	2	3	5	1	1

2. BACKGROUND

In this section give some definitions and define the problem of utility mining, and then introduce related work in utility mining.

2.1 Preliminary:

Given a finite set of items I= $\{i_1,i_{2,}\ldots\ldots,i_m\}$, each item i_p $(1\leq p \leq m)$ has a unit profit pr(i_p).An item set X is a set of k distinct items $\{i_1,i_{2,}\ldots\ldots i_k\}$, where $i_j \in$ I, $1\leq j \leq k$.k is the length of X. an item set with length k is called a k-item set. A transaction database D = $\{T_1,T_{2,}\ldots\ldots,T_n\}$, 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 item set 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 item set 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 item set is called a high utility item set 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 item set.

Definition 5. Transaction-weight utility of an item set X is the sum of the transaction utilities of all the transaction utilities containing X, which is denoted as TWU(X) and defined as $\sum_{X \subseteq T_d \wedge T_d \in D}$ TU (T_d) .

2.2 Related Work:

FP Growth algorithm [Hanet et al,2000] that combines projection with the use of the FP tree data structure to compactly store in memory the item sets of the original database.

Weight association rule mining C.H. Cai [4], weight association rule mining considers the importance of items, such as transaction databases, items. If the items are treated non-uniformly .The introduction of the weights made the mining of the association rules possible to be interactive with the users. It can provide a means to apply the knowledge to the items. Although two-phase algorithm reduces search space by using TWDC (Downward closure property) property, it still generates too many candidates to obtain HTWUIs (High Transaction Weight Utility Item) and requires multiple database scans.

To overcome this problem Li et al[13] Isolated Items Discarding Strategy to reduce the number of candidates. By pruning isolated items during level-wise search, the number of candidate item sets for HTWUs in phase 1 can be reduced. However, this algorithm still scans database for several times and uses a candidate generation and test scheme to find high utility item sets. UP Growth achieves better performance than FP Growth and Weight association rule mining. UP Growth algorithm is mainly composed of two mining phases such as EEGU (Enhanced Eliminating Global Unpromising

items) and EEGNU (Enhanced Eliminating Global Node Utilities).These two strategies are used to minimize the overestimated utilities stored in the nodes of global UP tree.

3. PROPOSED METHODS

The framework of the proposed methods consists of three steps: 1) scan the database to identify TU, TWU 2) Apply EEGU to remove unpromising items from the database 3) Apply EEGNU to construct global UP tree with promising item to identify high utility item sets.

3.1 The proposed Data Structure: UP Tree:

To facilitate mining performance and avoid scanning original database repeatedly, and reduce time complexity use a compact tree structure, named UP-tree, to maintain the information of transactions and high utility item sets.

Two strategies (EEGU&EEGNU) are applied to minimize the overestimated utilities stored in the node of global UP-Tree. First elements of UP-Tree are defined. Then how to construct a global UP-tree with two strategies (EEGU, EEGNU) is illustrated in details.

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 node's item name. N.count is the node's support count. N.nu is the node's node utility.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 header table is employed to facilitate the traversal of UP-Tree.

3.2 strategies EEGU:

Enhanced Eliminating Global Unpromising items during construction of global UP-Tree.The construction of global UP-Tree can be performed with scan the original database. First 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 item sets if its TWU is less than the minimum utility threshold. Such an item is called an unpromising item. According to EEGU global unpromising items and their actual utilities are removed from transactions and transaction utilities of the database. New TU after pruning unpromising item is called reorganized transaction utility (RTU).

3.3 Constructing a global UP-Tree by applying EEGU and EEGNU

During construction of global UP-Tree TU and TWU of each item is computed. From that user get promising and unpromising items. After getting all promising items EEGU is applied. The transactions are reorganized by pruning the unpromising items and sorting the remaining promising items in a lexicographic order. Each transaction after reorganization is called reorganized transaction. Then EEGNU is applied.

Algorithm: UP Growth with (EEGU and EEGNU). Mine high utility item sets from transactional database.

Input: D, a transaction database.

Profit table.

Minsup, minimum support threshold.

Output: frequent high utility item sets.

Method: The UP tree with EEGU and EEGNU is constructed in the following steps

1. Scan the database D to collect TU and TWU(X)
2. Select min_sup user specified threshold.
3. To set TWU(X) ≤ min_sup remove unpromising items and insert the available promising items in a header table H in descending order.
4. Repeat the above step until the end of the database D.
5. During the EEGU process transactions in the header table are reorganized by sorting the promising items and subtracting utilities of unpromising items from their TUs. The promising items and their TUs are stored in Reorganized Transaction Table (RTT).

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 2, Issue 4, pp: (334-340), Month: October - December 2014, Available at: www.researchpublish.com

6. Then a function Insert_Reorganized_Path is called to apply the EEGNU process for global UP tree construction using RTT.

7. Create a root of UP (EEGU and EEGNU) and label it as N_R.

Then Insert_Reorganized_Path (N,i_x) is called. Where N is a node and i_x is an item. First (N_R, i_1) is taken as input. The node for i_1,N_{i_1} is found or created under $N_R$and its support is updated. Then EEGNU is applied to discarding the utilities of descendent node under N_{i_1},i.e., N_{i_2} to N_{i_n} . Finally (N_{i_1},i_2) is taken as input recursively.

Procedure: Insert_Reorganized_Path (N, i_x)

Line 1: if N has a child N_{i_x} such that N_{i_x}.item = i_x , increment N_{i_x}.count by p_j.count where p=<N_{i1} , N_{i_2} ……N_{i_m} >

Otherwise create a new child node N_{i_x} with N_{i_x}.item=i_x,N_{i_x}.count= p_j.count, N_{i_x}.parent=N and N_{i_x}.nu=0.

Line: 2 increase N_{i_x}.nu

Line: 3 if their exists node N_{i_x} in p_j where x+1 < m',

Call Insert_Reorganized_Path (N, i_x)

Example for Constructing Global UP tree with EEGU & EEGNU:

Consider the transaction database in TABLE 1 and the profit table in TABLE 2. Suppose min_util is 50. During scan of the database, TUs of all transactions and TWUs of distinct items are computed. Five promising items, i.e., {cz3}: 93; {tusq}: 92; {dolo650}: 99; {eldoper}: 96 and {amlong}: 107, are sorted in the header table by the descending order of TWU, that is, {amlong}, {dolo650}, {eldoper}, {cz3}, and {tusq}.

Then the transactions are reorganized by sorting promising items and subtracting utilities of unpromising items from their TUs.

TABLE 3 TWU of each item

Cz3	93
Tusq	92
Dolo650	99
Eldoper	96
Amlong	107
Zincovit	37
Concor	40
Dianil	12

TABLE 4 Reorganized Transaction Table and their RTUs

TID	**TRANSACTION**	**TU**
T_1	(dolo650,10) (eldoper,1) (cz3,1)	17
T_2	(amlong,2) (dolo650,6) (cz3,2)	22
T_3	(amlong,2) (eldoper,6) (cz3,2) (tusq,2)	32
T_4	(amlong,1) (dolo650,13) (eldoper,3) (tusq,4)	30
T_5	(amlong,1) (dolo650,4) (tusq,2)	11
T_6	(dolo650,1) (eldoper,1) (cz3,1) (tusq,1)	10

The reorganized transactions and their RTUs are shown in Table 4. Comparing Tables 4 and 1, the RTUs of T2, T3,T5 and T6 in Table 4 are less than the TUs in Table 1 since the utilities of {zicovit}, {concor}, and {dianil} have been removed by EEGU and also utilities of unpromising items are removed from the promising item sets TUs.

After a transaction has been reorganized, it is inserted into the global UP tree by using the strategy EEGNU. In EEGNU individual items utilities are taken for tree construction instead of TUs. A table named header table is employed to facilitate the traversal of UP-Tree. 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.

4. EXPERIMENTAL EVALUATION

In this section show that the proposed research outperforms the state-of-the-art algorithms almost in all cases on medicine transaction data sets. The reasons are described as follows. First, node utilities in the nodes of global UP-Tree are much less than TWUs in the nodes. EEGU and EEGNU effectively decrease overestimated utilities during the construction of a global UP-Tree.it takes individual utility of each item instead of TWU of each item for tree construction. UP-tree and UP-tree (EEGU and EEGNU) generate much fewer candidates than FP-growth since EEGU, EEGNU and User based transaction retrieval process are applied during the construction of local UP-Trees. UP-Tree (EEGU and EEGNU) outperforms UP-Tree although they have tradeoffs on memory usage based on TWU. Experimental results show that the proposed algorithms, especially UP-Tree with EEGU and EEGNU, reduce time complexity rate to obtained 78% for UP Tree and 22% for UP tree with EEGU-EEGNU.

Time complexity computation for UP tree versus UP tree with EEGU and EEGNU:

The main purpose of this research is used to analyze the scanning time in transaction dataset and tree construction between two processes. Formula for computing time complexity,

UP Tree=O (N)

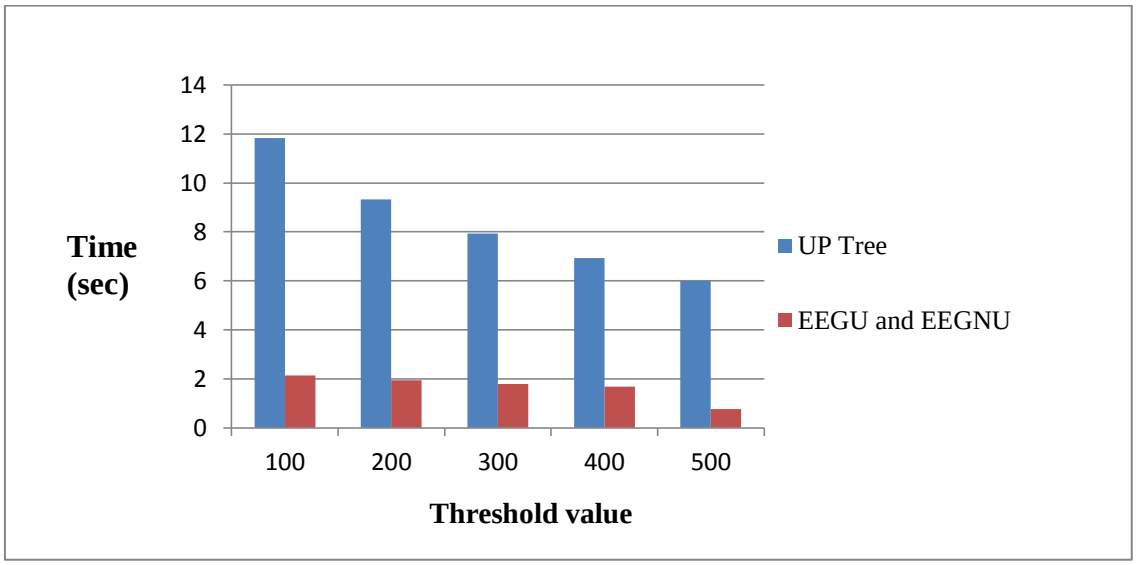
UP Tree (EEGU and EEGNU=O (Log N)

The UP tree time complexity is computed based on the O (N). The 'O' is the order of the value and 'N' refers the number of transaction by per item in the medical transaction dataset. The UP tree with EEGU and EEGNU is computed using the formula for O (Log N). The value is providing the logarithm function of 'N'. The 'N' represents the number of transaction by per item in the medical transaction dataset. The time complexity of an algorithm is commonly expressed using big O notation, which excludes coefficients and lower order terms. When expressed this way, the time complexity is said to be described asymptotically, i.e., as the input size goes to infinity. An O (Log N) is said to run in logarithmic time if its time execution is proportional to the formula of the input size. Time Algorithm that has running time O (log n) is slight faster than O (n). Linear Time the O (N) accepts n input size; it would perform n operations as well.

Table 5: Time comparison between UP Tree verses EEGU and EEGNU

Threshold value	UP Tree	EEGU and EEGNU
100	11.83	2.14
200	9.32	1.93
300	7.93	1.79
400	6.92	1.68
500	6	0.77

CHART 1: comparison of Time process between UP Tree versus EEGU and EEGNU



4.1 Summary of experimental results:

This thesis is used to eliminate time complexity rate while finding high utility item sets in a transaction database. In this proposed research, tree construction process using two strategies, namely EEGU (Enhanced Eliminating Global Unpromising items) and EEGNU (Enhanced Eliminating Global Node Utilities). It also used to reduce number scans to the database.

5. CONCLUSION

The proposed research is used to eliminate time complexity rate in tree construction process by using two strategies, namely EEGU (Enhanced Eliminating Global Unpromising items) and EEGNU (Enhanced Eliminating Global Node Utilities). It also used to reduce number scans to the database. So the time required to run the project is faster compared to existing research. At final, thesis is used to predict the frequent moved item details in the transaction dataset and calculate the time complexity rate for both two tree construction by user specified manner. And also dynamic updating is available in this research.

6. FUTURE ENHANCEMENTS

The proposed research is used to predict the frequent moved item in the transaction dataset. It is very fast in applying algorithm for analyzing the profit level based transaction. This implementation is very particular in predict the profit based item sale and user based item details in the efficient manner.

The proposed work become more useful if the below enhancements are made in future. In future work, tree will be constructed for both frequent and infrequent moved items in dataset.

In addition, the advanced Mining algorithm can be applied to other applications with the aim to enhance precision for predicting user behaviors.

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,T.Imielinski, and A.Swami.Mining association rules between set of items in large databases.In Proc .of the ACM SIGMOD conference on management of data ,Washington,D.C.,May 1993.

[3] M.Houtsma and A.Swami. Set-oriented mining of association rules.Research Report RJ 9567.IBM Almaden Research center, Sanjose, California, October 1993.

[4] 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.

[5] C.M.Kuok,A.Fu and M.H.Wong. Mining fuzzy association rules in databases. In SIGMOD Record,27(1),March,1998.

[6] C. Creighton and S. Hanash, "Mining Gene Expression Databases for Association Rules," Bioinformatics, vol. 19, no. 1, pp. 79-86, 2003.

[7] Doddi,S., Marathe,A., Ravi,S.S. and Torney,D.C. (2001) Discovery of association rules in medical data. Med. Inform. Internet. Med., 26, 25–33.

[8] M.Y. Eltabakh, M. Ouzzani, M.A. Khalil, W.G. Aref, and A.K. Elm agarmid, "Inc remental Mining for Frequent Patterns in Evolving Time Series Databases," Technical Report CSD TR#08-02, Purdue Univ., 2008.

[9] M.Y. Eltabakh, M. Ouzzani, M.A. Khalil, W.G. Aref, and A.K. Elm agarmid, "Inc remental Mining for Frequent Patterns in Evolving Time Series Databases," Technical Report CSD TR#08-02, Purdue Univ., 2008.

[10] Dan Lim, "Ubiquitous mobile Computing: UMC's model and success", Educational Technology & society 2(4) 1999 ISSN 1436-4522.

[11] C.Y Chang, M.S Chen, "Integrating web caching and web prefetching in client-side proxies", in: Proceedings of the ACM 11th International conference Information and knowledge Management, 2002.

[12] Bai-En Shie, Hui-Fang Hsiao, Vincent S. Tseng, and Philip S. Yu, Mining High Utility Mobile Sequential Patterns in Mobile Commerce Environments, National Cheng Kung University.

[13] Y.C. Li,J.S. Yeh,and C.C.Chang ,"Isolated Items Discarding Strategy for Discovering High Utility Item sets, "Data and Knowledge Eng.,vol.64,no.1,pp.198-217,Jan.2008.