Indian Journal of Science and Technology, Vol 9(48), DOI: 10.17485/ijst/2016/v9i48/107990, December 2016
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Reduced Overestimated Utility and Pruning Candidates using Incremental Mining

M. Indumathi* and V. Vaithiyanathan

School of Computing, SASTRA University, Tirumalaisamudram, Thanjavur – 613401, Tamilnadu, India;
masuinta86@gmail.com, vvn@it.sastra.edu

Abstract

Objectives: We proposed a Maximum Utility Growth (MUG) algorithm and Maximum Item Quantity (MIQ) Tree structure is used to decrease the number of candidates and reduce the candidate Itemset. **Methods:** In high utility Itemset mining profit and quantity of items are important in transaction. Several algorithms have proposed to solve the issues of large number of candidate Itemset generation. The proposed MUG algorithm and MIQ-Tree structure is used to reduce candidate Itemset and overestimated utilities in incremental mining. Moreover, the MUG with second strategy of approach calculates maximum utility of each expanded Itemset from the current prefix with supports in mining process. **Finding:** MUG algorithm and MIQ-Tree are proposed for mining high utility itemset to reduce the overestimated utility and it proposed two strategies for pruning candidate item sets efficiently in the process. It reduces the number of candidate and improves the performance of incremental mining. MIQ-Tree proposed to construct the tree with single-pass. The tree structure can restricted without new database and it employs decreasing the overestimated utility. MUG proposed to prune the number of candidate itemset with two strategies (I) Pruning 1-Itemset Candidates with Real Item Utilities and (II) Pruning Candidates with Estimated Maximum Itemset Utility. The experimental shows that the method developed the performance by decreasing number of candidate itemset. Performance evaluation shows that it reduces the number of candidate and its runtime with equal usage memory. Through the strategy, it can effectively eliminate the search space in the process. **Applications:** There is large amount of real world application such as retail marketing and stock marketing has emerged techniques high utility Itemset mining.

Keywords: Candidate Pruning, Data Mining, High Utility Itemset, Incremental Mining, Single Pass Tree Construction

1. Introduction

Data mining is the computing processes of developing pattern in huge dataset include a method at the intersection of artificial intelligences, statistic, and database system. The Main aim of data mining process is separate instruction from the dataset and alters to a meaningful structure for further use. Data mining is the examiner step of the knowledge discovery. Data mining is a powerful technology to help the company focus on the most important information in the data they have collected about the presence of their customer and potential customer. It finds out information within the data that queries. Data Mining is separating of pattern and knowledge from huge amounts of data. It is frequent apply to all forms of huge scale data processing. The exact data mining work is to automatically proof a huge quantity of data and also to separate previously unknown data. The term data snooping assign to use of data mining method to part of a huger dataset to small the validity of any patterns identity. Data mining employs six classes of tasks: Anomaly detection-The identification of unusual data records. Association rule is to learn or Search for relationship between variable. Clustering-The scheme of discover group and structure in data. Classification-The task of generalize known structure to applying to new data. Regression is designed to find a function that model the data with the less error. Summarization-providing a many compact

*Author for correspondence

represent of the dataset. The Associate rule based mine on the transactional database process since it cost to verify the associative rule in huge databases. With usual market-basket application, a new transaction is developed and transaction may obsolete time. As a result, Incremental update technique should generate for assumption of discovering association rules to quite redoing mining on all update database. A database may allow frequently or occasional update and such updates may only invalidated old association rules but also activate new rules. Thus, it is non-trivial to approve such finding rule in huge database. An incremental mining has a varied types of mining techniques which can apply for varies environment where a database grow and change frequently. An algorithm for incremental mining is to encourage solving the problem of high utility item sets mining. It is based on two-phase mining approach. According to transaction weighted utilization item sets it divides item sets into four parts in the old database transform into new transaction. Every portion is executed by its procedures. High utility Itemset mining establishes itemset whose utilities satisfy a given threshold. It allows user to quantity the useful items using variable values. It considers of varying item. High utility Itemset mine is useful in the decision-making process. Due to need of Downward Closure Property, that number of candidates to generation of utility itemset mining is guilty of time and memory space. It presents a Two-phase algorithm which can evenly reduce the candidate itemset and overall arrangement of utility item sets. The performance of the algorithm is checked by rubbing the database. It performs very effective of speed and memory performance on huge databases secure of transaction, which are demanded for mining high utility item sets algorithms.

In data mining, finding the frequent items[1-3], Apriori[4] and Frequent Pattern Growth (FP-Growth)[5] algorithm are basic frequent pattern mining. FP-Growth algorithm is better when compared with Apriori algorithm, it achieve better performances[5]. Though frequent pattern mining is important in pattern mining field, it doesn't consider the importance and quantity of the item. Weighted frequent pattern mining[6] attends to indicate the importance of items in the dataset. In that pattern weight is calculated by using finding ratio between the sum of weight of items and pattern length. In the weighted frequent pattern[6], it doesn't examine the quantity of an item. High-Utility Pattern (HUP) algorithm[7] was proposed, to refuse many databases and generate the number of candidate items. It has three tree structure are HUPL-Tree, HUPTF-Tree and HUPTWU-Tree. In HUP, Each node has Name of the item, Support Count Value and Transaction Weight Utility value. HUP generated high utility item through 3 steps (I) Items in transaction are arranged according to lexicographic order and transaction are interpolated into HUPL-Tree with single data scan. (II) Candidate items educed from HUP-Tree. (III) Absolute utility item are extracted from new database. Though HUP can construct a tree to find high utility itemset with two databases and generated more number of candidates by assign the Transaction Weighted Utility (TWU) model. UP-Growth[8] has been proposed and its usage Potential High Utility (PHU) model. Reduce the candidate itemset; it applies four strategies, Discarding Global Unpromising items (DGU), Decreasing Global Node utilities (DGN), Discarding Local Unpromising items (DLU), Decreasing Local Node utilities (DLN)[8]. It constructs a tree with two database and conduct utility itemset. It needs three databases for finding high utility item sets, (I) TWU values of each item are an aggregated. (II) Item which has less TWU than a user-specified threshold is rejected. (III) Items are arranged according to TWU value decreasing order and inserted into UP-Tree. As above stated algorithms or reduce the number of candidates is a necessary issue in tree based algorithms for mining the highly utilized items with overestimated method. Aim of study is used reducing overestimated utilities and pruning to reducing the number of candidate itemset in the database for that use a sample transaction table Table 1. and profit table Table 2. are used.

2. Proposed Tree Structure and Method

2.1 MIQ-Tree Structure

The proposed tree structure, named MIQ-Tree is use the information of transaction and highly utilities items .Each item quantity is involved for constructing the tree. Tree is used to reduce the overestimated utility and design the global tree with single-pass acquire TWU value are noted as noted utility. Finally, large numbers of candidate are generated in the process.

2.1.1 Element in MIQ-Tree

In Tree, Each and Every node R is consist of R.Name, R.Count, R.Nu, R.Parent element which store the data of item and quantity for estimate rejected the overestimated

Table 1. An example database

TID	Transaction	TU
T1	(P,1)(Q,2)(T,4)	19
T2	(P,2)(Q,1)(S,1)	17
T3	(Q,3)(R,2)(T,3)(U,4)	25
T4	(R,1)(S,2)(T,1)	11
T5	(Q,2)(S,1)(T,1)(U,3)	18
T6	(R,3)(T,2)	7
T7	(P,1)(Q,3)(R,1)(S,2)	23
T8	(P,2)(Q,1)(R,3)(S,1)(T,2)	24

Table 2. Profit table

ITEM	P	Q	R	S	T	U
PROFIT	5	3	1	4	2	2

utility from the initial tree[9]. MIQ-Tree not has memory for local the trees. R.Name is name of the item, R.Count is support count value, and R.Nu denotes the nodes utilities, R.Parent and R.nodelink is parent node of N. An item support is only contained in local header tables, and it is employed for reducing the number of candidates in mining process[10].

2.1.2 Construction of MIQ-Tree

Algorithm: Insert_Transaction (Tree, T)

1. R ←the root node of Tree
2. for each item (l, q) in T do
3. If R has not a child node R_l such that R_l.name=l then
4. Create a new child node R_l under R
5. Set R_l.name=l, R_l.count=0, R_l.max=0
6. Increase R_l.count by 1
7. If R_l.max<q then R_l.max=q
8. R←R_l

MIQ-Tree constructs the initial tree using the transaction database. Each transaction is included into tree and its TU values, TWU value each item also calculated and accumulates the entry at the same time. To admit transaction $T_d(l_1, q_1)$ (I_2, q_2) (l_n,q_n)..... (l_k,q_k), where l_k denotes item and $_{qk}$ denotes the quantity of the item Insert-transaction function is called. Each item and quantity of tree is added in the transaction. Initially R. Count, R.sum are zero then increased R. Count by 1, it check current R.max and pervious R.max value and quantity value, repeat the step for remaining items. The other transactions are inserted to the initial tree is shown in the Figure 1.

2.1.3 Restructuring of MIQ-Tree

Each path P is removed from the tree and reordered according to TWU descending order at the time, each node in P path utility is estimated. Path utility assign to reduced overestimating utility[11]. Let {R_1, R_2, . . ., R_l } be nodes of a tree T and the nodes be in TWU descending order, where an item l_1 has the highest TWU value. That is, R_1 is a child of the root node and it is can have a leaf node. Mining is performed in bottom-up manner is shown in the Figure 2.

2.1.4 Conditional Pattern Tree

Utility of each node in {Nl1, N l2, . . ., Nlk} can be discarded from path utilities. Each node in {*Nl1, Nl2, . . ., Nlk*}.If there is a path $P = \{N_A, N_B, N_C\}$in TWU descending order[12]. From the Figure 3. the proposed tree structure

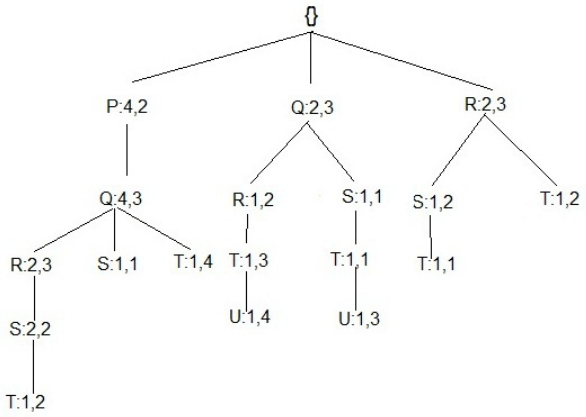

Figure 1. Constructed MIQ-tree.

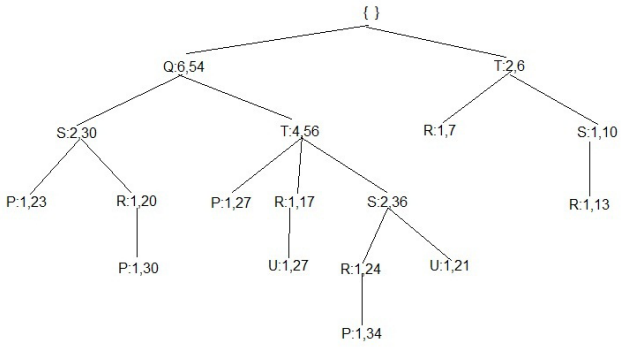

Figure 2. Restructured MIQ-tree.

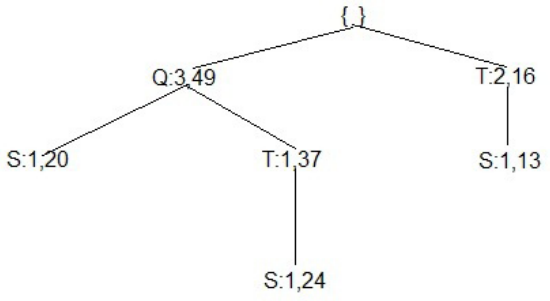

Figure 3. Conditional pattern tree.

maintains node quantities for calculating reduced overestimating utilities with a single-pass.

2.2 MU-Growth

MUG algorithm can contract the number of candidate items with minimum and maximum of item utility, support count of local items and real item utility. If estimated maximum utility value is calculated for each candidate item which itemset has value is less than minutil, that candidate itemset is not developed in mining process.

2.2.1 Pruning 1-Itemset Candidates with Real Item Utilities

Mining high utility item sets is operated in bottom-up manner. Total MIQ-tree is cover by successive each item in the header table, which is a global item in tree. A condition pattern tree is structured from global tree selecting all paths starting from nodes. Using the Table 3. Candidate item sets along with the condition pattern tree, where TWU minutil. The method prunes 1-item candidate using real item utility from the global tree[13].

2.2.2 Pruning Candidates with Estimated Maximum Itemset Utility

MUG achieves mining high utility itemset with items has no less than TWU value than minutil in initial tree. Each item in a tree is based on the conditional pattern at same time support counts are calculated from the Table 4. Each candidate itemset estimated the maximum utility to reject the candidate which has less maximum utility than minutil. It uses minimum utility for estimating the path utility of each item. Local tree is constructed to eliminate the item which has less TWU value than minutil. In the stage, the cancel utilities of items are calculated by

Table 3. Real item utility Table

ITEM	P	Q	R	S	T	U
UTILITY	30	36	10	28	26	14

Table 4. Minimum and maximum item utility

ITEM	P	Q	R	S	T	U
MIN	5	3	1	4	2	6
MAX	10	9	3	8	8	8

the minimum item utility are pruned and support count value of the item[14].

MUG (Tree)

1. For each item l_g such l_g.twu≥minutil in the header of Tree do
2. If $U_{real\geq}$ minutil then generate a candidate itemset $\{l_g\}$
3. twu←Ø
4. For each node R_g such that R_g.name=l_g in Tree do
5. Increase twu by R_g.nu
6. If twu≥minutil then
7. Create$\{l_g\}$-CPT T_g
8. Call MUG(T_g,$\{l_g\}$)

MUG (T_x, Y)

9. For each l such that l. twu≥minutil in the header of T_x do
10. twu←Ø
11. For each node R_g such that R_g.name=l_g in TX do
12. Increase twu by R_g.nu
13. If twu≥minutil then
14. Y←Y {l}
15. If R.count=0 then Remove R from Tree
16. If emciu(Y) ≥minutil then generate itemset Y
17. Create CPT for Y, T_x
18. Call MUG (T_x, Y)

3. Experiment

3.1 Performance Comparison on Different Datasets

The experiment differentiate the performance of varies algorithms on real utility datasets; Chain-store and Foodmart, in terms are results of the total runtime and the number of candidates on Chain-store. The total runtime of MUG applying two pruning strategies is the best, followed by the first strategy MU-Strategy 1, the second one

MU-Strategy 2, and both UP-Growth and UP-Growth+ are the worst. That is, MUG mines with the fastest speed of all utility item sets. Utility threshold is fixed to larger than or equal to 0.03, previous tree-based algorithms.

3.2 Performance Comparison under Varied Strategies

MUG employs two strategies for effectively pruning candidate item sets. The First strategy is pruning 1-itemset candidates with real item utility and the second one is pruning candidates using estimated maximum item set utility. From the Figure 4. All experiments because techniques for reducing overestimated utilities are applied to the proposed tree structures and each strategy effectively decreases the number of candidate item sets and improved the performance.

4. Result

Experiments for performance of evaluation of the compared algorithm mentioned above are conducted. The performance of MIQ-tree and MUG algorithm in mining high utility item set. The algorithm used to calculate the TWU values for every item in transaction and arranging item according to TWU value decreasing order. Next algorithm accomplished the candidate item set repeatedly from the tree. Finally, they determinate absolute high utility from the candidates. To show efficient pruning method for MUG algorithm. They give

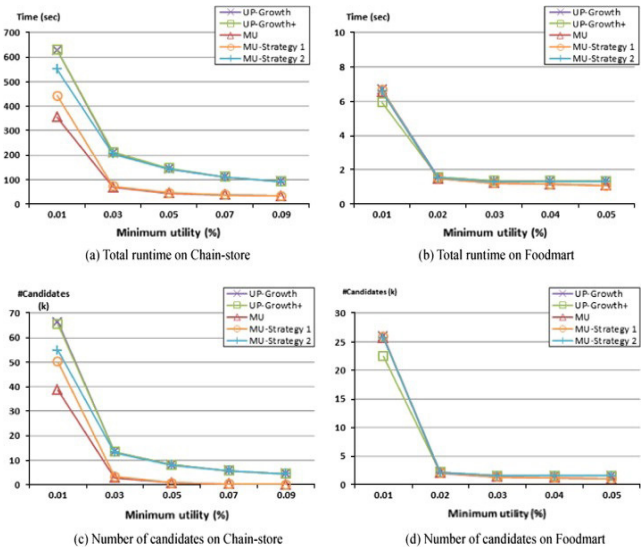

Figure 4. Performance comparison under varies dataset and strategies.

better performance, scalability and memory usage of tree structure and algorithm.

5. Conclusion

MUG algorithm and MIQ-Tree are proposed for mining high utility item set to reduce the overestimated utility and it proposed two strategies for pruning candidate item sets efficiently in the process. It reduces the number of candidate and improves the performance of incremental mining. MIQ-Tree proposed to construct the tree with single-pass. The tree structure can restricted without new database and it employs decreasing the overestimated utility. The experimental shows that the method developed the performance by decreasing number of candidate item set.

6. Reference

1. Caldersa T, Dextersb N, Gillisc JJM, Goethalsb B. Mining frequent item sets in a stream. Information Systems. 2014 Jan; 39:233–55.
2. Chuang KT, Huang JL, Chen MS. Mining top-k frequent patterns in the presence of the memory constraint. The International Journal on Very Large Data Bases. 2008 Aug; 17(5):1321–44.
3. Duonga H, Truonga T, Vob B. An efficient method for mining frequent item sets with double constraints. Engineering Applications of Artificial Intelligence. 2014 Jan; 27:148–54.
4. Agrawal R, Srikant R. Fast algorithms for mining association rules. In Proceeding of the 20th International Conference on Very Large Data Bases (VLDB 1994); 1994. p. 487–99.
5. Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation. In Proceeding of the Association for Computing Machinery Special Interest Group on Management of Data (ACM SIGMOD) International Conference on Management of Data; 2000 Jun. p. 1–12.
6. Yun U, Ryu K. Efficient mining of maximal correlated weight frequent patterns. Intelligent Data Analysis. 2013 Sep; 17(5):917–39.
7. Manimaran J, Velmurugan T. Analysing the quality of association rules by computing an interestingness measures. Indian Journal of Science and Technology. 2015 Jul; 8(15):1–12.
8. Ahmed CF, Tanbeer SK, Jeong BS, Lee YK. Efficient tree structures for high utility pattern mining in incremental databases. Institute of Electrical and Electronics Engineers (IEEE) Transactions on Knowledge and Data Engineering. 2009 Dec; 21(12):1708–21.

9. Tseng VS, Wu CW, Shie BE, Yu PS. UP-growth: an efficient algorithm for high utility item set mining. In Proceeding of the 16th Association for Computing Machinery's Special Interest Group on Knowledge Discovery and Data Mining (ACM SIGKDD) International Conference on Knowledge Discovery and Data Mining (KDD). 2010 Jul. p. 253–62.
10. Erwin A, Gopalan RP, Achuthan NR. Efficient mining high utility item sets from large datasets. Springer Berlin Heidelberg; 2008 May. p. 554–61.
11. Tseng VS, Shie BE, Wu CW, Yu PS. Efficient algorithms for mining high utility item sets from transactional databases. Institute of Electrical and Electronics Engineers (IEEE) Transactions on Knowledge and Data Engineering. 2013 Aug; 25(8):1772–86.
12. Vo B, Coenen F, Bac L. A new method for mining frequent weighted item sets based on wit-trees. An International Journal of Expert Systems with Applications. 2013 Mar; 40(4):1256–64.
13. Yun U, Lee G, Ryu K. Mining maximal frequent patterns by considering weight conditions over data streams. Knowledge-Based Systems. 2014 Jan; 55:49–65.
14. Lee G, Yun U, Ryu K. Sliding window based weighted maximal frequent pattern mining over data streams. Expert Systems with Applications. 2014 Feb; 41(2):694–708.