International Journal of Future Innovative Science and Engineering Research (IJFISER)
Volume – 3, ISSUE – 2 **ISSN (Online):2454- 1966**

Research Manuscript Title

PATTERN GROWTH ALGORITHM FOR MINING HIGH UTILITY ITEMSETS

T.Gladima Nisia[1], K.Banu Priya[2],V.Gayathri[3]

[1]Assistant Professor, AAA College of Engineering and Technology, Sivakasi

[2,3]UG Student, AAA College of Engineering and Technology, Sivakasi

Corresponding author E-Mail-ID: gladimab@gmail.com

Jun – 2017

www.istpublications.com

International Journal of Future Innovative Science and Engineering Research (IJFISER)
Volume – 3, ISSUE – 2 ISSN (Online):2454- 1966

PATTERN GROWTH ALGORITHM FOR MINING HIGH UTILITY ITEMSETS

T.Gladima Nisia[1], K.Banu Priya[2],V.Gayathri[3]

[1]Assistant Professor, AAA College of Engineering and Technology, Sivakasi

[2,3]UG Student, AAA College of Engineering and Technology, Sivakasi

Corresponding author E-Mail-ID: gladimab@gmail.com

ABSTRACT

Mining high utility itemsets from a transactional database refers to the discovery of itemsets with high utility like profits. Although a number of relevant algorithms have been proposed in recent years, they incur the problem of producing a large number of candidate itemsets for high utility itemsets. In this paper, we propose an algorithm namely Pattern Growth Algorithm (P_Growth Algorithm) for mining high utility itemsets from frequent itemsets with a set of effective strategies for pruning candidate itemsets. The information of high utility item sets is maintained in a Graph based data structure named Pattern Graph (P_Graph) such that candidate item sets can be generated efficiently. Algorithm named Pattern Growth Algorithm and a compact data structure, called Pattern Graph for discovering high utility item sets and maintaining important information related to utility patterns within databases are proposed. The Classification based on Multiple Association Rule (CMAR) is utilized inorder to generate frequent itemsets. Several strategies are proposed for facilitating the mining processes of P-Growth Algorithm by maintaining only essential information in P_Graph. The Pattern Graph is generated for finding High Profit with Support Count.

Keywords: *P-Growth Algorithm, Itemsets, P_Graph, Multiple Association Rule, Support Count.*

1. INTRODUCTION

Data Mining is the process of revealing nontrivial, previously unknown and potentially useful information from large databases. Discovering useful patterns hidden in a database plays an essential role in several data mining tasks, such as frequent pattern, weighted frequent pattern mining, and high utility pattern mining. Among them, frequent pattern mining is a fundamental research topic that has been applied to different kinds of databases, such as transactional databases [1],[14],[17], streaming databases [18],[3], and time series databases [9],[12], and various application domains, such as bio in for-matics [8],[11],[2], Web click-stream analysis [7], mobile environments [15],[14].

Algorithm, named pattern growth (P_Growth) and a compact data structure, called Pattern Graph (P_Graph), for discovering high utility item sets and maintaining important information related to utility patterns within databases are proposed.

High-Utility item sets can be generated from P_Graph efficiently with only two scans of original databases. Several strategies are proposed for facilitating the mining process of P_Growth Algorithm by maintaining only essential information in P_Graph. By these Strategies, overestimated utilities of candidates can be well reduced by discarding utilities of the items that cannot be high utility or are not involved in search space.

Mining high utility item sets from databases refers to finding the item sets with high profits. Here, the meaning of utility item set is interestingness, importance, or profitability of an item to users. Algorithms, named Pattern Growth Algorithm and Graph structure called Pattern_Graph (P_Graph), for discovering high utility item sets and maintaining important information related to utility patterns within databases.

Finding frequent item sets is one of the most investigated fields of data mining. Many algorithms have been proposed to find frequent item sets from a very large database. Transaction Mapping algorithm is also an algorithm to find frequent item sets mining. But this is not a time consuming one. However, there is no implementation on every database with every support threshold. Earlier Decision tree and FP(Frequent Pattern) algorithms are the established algorithm for frequent item sets mining. Both these algorithms doesn't satisfy two constraints which are time and accuracy.

Existing studies [3], [10], [16], [17], [19], [24], [29], [30] applied overestimated methods to facilitate the performance of utility mining. In these methods, potential high utility itemsets (PHUIs) are found first, and then an additional database scan is performed for identifying their utilities. However, existing methods often generate a huge set of PHUIs and their mining performance is degraded consequently. This situation may become worse when databases contain many long transactions or low thresholds are set. The huge number of PHUIs forms a challenging problem to the mining performance since the more PHUIs the algorithm generates, the higher processing time it consumes.

To address this issue, we propose algorithms as well as a compact data structure for efficiently discovering high utility itemsets from transactional databases. Major contributions of this work are P_Graph and Pattern Growth Algorithm.

2. Methodologies

The Proposed strategies can not only decrease the overestimated utilities of PHUIs (Potential High Utility Itemsets) but greatly reduce the number of candidates. Different types of both real and synthetic data sets are used in a series of experiments to the performance of the proposed algorithm with state-of-the-art utility mining algorithms. Experimental results show that P_Growth outperform substantially in term of execution time, especially when databases contain lots of long transactions or low minimum utility thresholds are set.

The Pattern Growth algorithm is the most established algorithm for Utility item sets mining (UIM). Pattern Growth outperforms all implementations. Several implementations of the Pattern Growth algorithm have been reported and evaluated. Figure 1 shows the system design of the proposed system.

1. The algorithm, named Pattern Growth (P_Growth) and a compact Graph structure, called Pattern Graph for discovering high utility item sets and maintaining important information related to utility patterns within databases are proposed.

2. High-Utility item sets can be generated from Pattern Growth Algorithm efficiently with only two scans of original databases. Several strategies are proposed for facilitating the mining process P_Graph by maintaining only essential information in P_Graph.
3. By these Strategies, overestimated utilities of candidates can be well reduced by discarding utilities of the items that cannot be high utility or are not involved in search space.

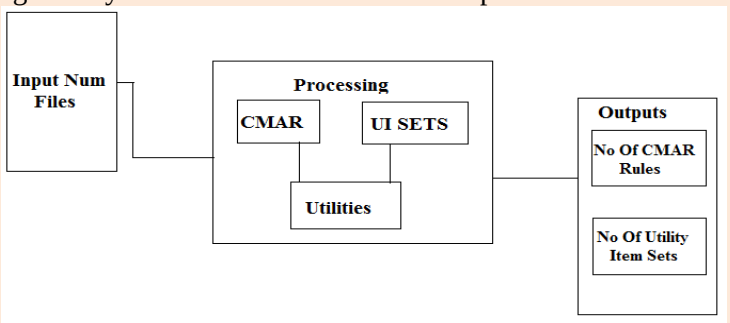

Figure 1: System design of the proposed system

In the first phase input number of files, contains the information of the purchase items. Each and every item in the text file assign a unique value. The input items contain both support and confident items. Then the both mixed items processed in the CMAR processing. CMAR categorize and classify the items. The processing phase contains both the CMAR and Utility item sets. There are three rule generated in CMAR,

i) Scan the training data sets
ii) Sort the rules
iii) Prune the rules

Find the utilities by using the Number of CMAR rules. Then finally calculating the utility itemsets in the output phase.

2.1 Classification based on Multiple Association Rules

The input file contains the information of the candidate purchase items. Classification based on Multiple Association Rules (CMAR). They adopted a variant FP-growth(Frequent Pattern) method to find complete set of CMARs, which is more efficient than Apriori-like method. CMAR consists of two phases: rule generation and classification. In the first phase, rule generation, it finds the complete set of rules in the form of R: P → c, where P is a frequent pattern passing support threshold, and c is a class label passing confidence threshold. It further prunes some rules and selects a subset of high quality rules for classification. In the second phase, classification, it extracts a subset of rules matching new data object and predicts the class label based on the subset of rules. The experimental results on performance study shows that CMAR is faster and has better classification accuracy than other classification methods.

First phase: Rule Generation

1. It first scans the training data set and finds all the frequent itemset. Then, sorts the attribute values in support descending order, F-list (Frequent list), and scans the training data again to construct a FP-tree. Third, based on the F-list generates the subset of CMARs without overlap. Fourth, prunes the FP-tree according to class label distribution.

2. Sort the rules in CR-tree. CR-tree is a compact tree structure. It explores potential sharing among rules and thus saves space since the rules that have common frequent items share the part of path. CR-tree can be used as an index of rules. Once a CR-tree is built, rule retrieval becomes efficient.

3. Prune the rules. Use general and high confidence rules to prune more specific and lower confidence ones. Select only positively corrected rules. Select a subset of rules based on database coverage.

Second phase: Classification based on Multiple Rules

CMAR divides the rules into groups according to class labels. Compare the strength of the groups by measuring the combined effect of each group. Theoretically, it is hard to verify the effect of measures on strength of groups of rules, CMAR adopted weighted X measure.

ALGORITHM

Selecting Rules Based On Database Coverage by CMAR

//Input: A set of rules and a coverage threshold

//Output: A subset of rules for classification Method.

Sort rules in the rank descending order.

For each data object in the training data set, set its cover-count to 0.

While both the training data set and rule set are not empty, for each rule R in rank descending order, find all data objects matching rule R. If R can correctly classify at least one object then select R and increase the cover-count of those objects matching R by a data object is removed if its cover-count passes coverage threshold.

2.2 Construction of P_Graph

This Data Structure contains atmost n childs. Initially at first level each node represent the separate items. If suppose customer buys a set of item sets at level two the single items are combine according to the customer choice and from a item sets in a single node.

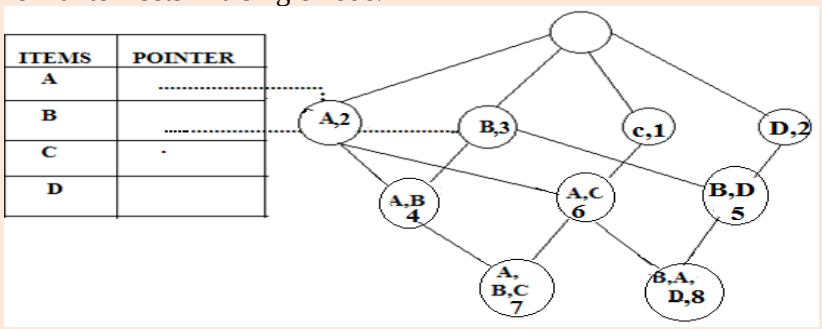

Figure 3: P_Graph with atmost n child

Association Rule

Association rules are used to show the relationships between data items. These uncovered relationships are not inherent in the data, as with functional dependencies, and they do not represent any sort of causality or correlation. Instead, association rules detect common usage of items. The strength of association rule is measured using support and confidence.

Support

Support shows the frequency of the patterns in the rule. It is the percentage of transactions that contain both A and B,

i.e) Support = Probability(A and B)

Support = (# of transactions involving A and B) / (total number of transactions).

Confidence

Confidence is the strength of implication of a rule. It is the percentage of transactions that contain B if they contain A.

i.e) Confidence = Probability (B if A) = P(B/A), Confidence = (# of transactions involving A and B) / (total number of transactions that have A).

TRANS_ID	PEN	INK	MILK	BREAD
T1	1	0	0	1
T2	0	0	1	1
T3	1	1	0	0
T4	1	0	0	0

Figure 2: Transaction table

In the above transaction table the value represents (i.e., 1=present 0=not present) Mining for associations among items in a large database of sales transaction is an important database mining function. For example, the information that a customer who purchases a keyboard also tends to buy a mouse at the same time is represented in association rule.

below: Keyboard ⇒Mouse [support = 6%, confidence = 70%]

Based on the types of values, the association rules can be classified into two categories: Boolean Association Rules and Quantitative Association Rules

• Boolean Association Rule: Keyboard →Mouse [support = 6%, confidence = 70%]

• Quantitative Association Rule: (Age = 26…30) → (Cars =1, 2) [Supt 3%, conf = 36%]

P_Graph Algorithm

Input: Purchased items.

Output: Set of transactional item sets.

initially start at level 0

for each item in the itemsets

create a single item as a separate node

end-for

combine the single item and merge at a separate node.

combine the itemsets and merge at a separate node

The itemsets may go to Level n (Level n –Node with n item sets).

//here, node n represent "Patterns" of item with support count.

2.3 Mining P_Graph using Pattern Growth Algorithm

Good pattern generation method:

Generating candidate patterns as less as possible.

Good database processing method

Sorting, aggregation and classification of data set according to the intrinsic principle of pattern mining.

The P_Graph generates the set of itemsets. Here if a customer buys a same item they are classified under the same category. Then using the Pattern_Growth Algorithm to find the high utility itemsets. Initially, assign a value to the given itemsets, then these value are applied to find the profit of item sets.

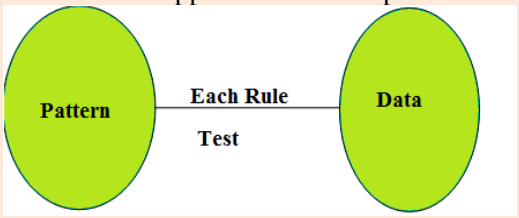

Figure 4: Profit Data Generation

When applying Association rule the support and the confidence items are classified with the given constrain (i.e., Confidence item = 50% Support item = 30%). Then find profit for the generated itemsets. Finally we get the profit of more number of item sets. If a node is visited once it is marked as read. This process is followed by each and every node in the graph.

The profit for High Utility Item sets can be found using,

Utility of Item Set = (Support Count * [$\sum_{i=1}^{n}(p_i)$])

where p_i = Profit of Item

Adding the profit of the items in the item sets and multiply the value with their Support Count. From which the high utility itemsets are generated (for example, user specified Threshold value=70).

Pattern Algorithm - P_Growth(T_y,H_y,Y)

//Input: P_Growth graph T_x, a header table H_x for T_x, an itemsets X, and threshold, min_util.

//Output: All PHUIs in T_x.

For each entry i_k in H_x do.

Trace each node related to i_k via i_k hlink and accumulate i_k.

Find the support count, S of each node

If (S > min_util)

Add ik to PHUI

End-if

End-for

If $T_y \neq$ null then call P_Growth(T_y,H_y,Y).

End if.

End for.

3. Experimental Evaluation

Performance of the proposed algorithms is evaluated in this section. The experiments were performed on a 2.80 GHz Intel Pentium D Processor with 3.5 GB memory. The operating system is Microsoft Windows 7. The algorithms are implemented in Java language. Both real and synthetic data sets are used in the experiments. Synthetic data sets were generated from the data generator in [1]. The system shows increased performance by reduction in the number of candidates and also increment in the system evaluation speed.

4. CONCLUSION

The frequent item set is evaluated. To generate frequent item sets support and confidence is given as threshold and support count of an item set is an aggregated of all local support count of item set. To generate frequent item set, a threshold value has to be specified which is known as minimum support count. The item set is set to be frequent if it satisfies support >= minimum support. Then the candidate item set is generated using support count in iterative passes. If no more items satisfy support count the process is stopped and frequent item set is generated. From the generated frequent itemsets the P_ Graph is constructed. Each and every product we assign a update value and also how many candidates purchased these items (ie., Setting minimum threshold) to find the utility for all set of items. Then setting a constrain value to find the maximum utility value for the item sets. In future, we would extend the concepts proposed in this work to discover other patterns like utility item with negative profit.

REFERENCES

[1]. Agrawal.R and Srikant.R, (2013) "Fast Algorithms for Mining Association Rules," Proc. 20th Int'l Conf. Very Large Data Bases (VLDB), pp. 487-499, 1994.1784 IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 25, NO. 8.

[2]. AhmedC.F, TanbeerS.K, JeongB.S, and LeeY.K, (2009)"Efficient Tree Structures for High Utility Pattern Mining in Incremental Databases," IEEE Trans. Knowledge and Data Eng., vol. 21, no. 12, pp. 1708-1721.

[3]. Chang K.V, (1995) "Mining Weighted Sequential Patterns in a Sequence Database with a Time-Interval Weight," Knowledge-Based Systems, vol. 24, no. 1, pp. 1-9, 2011. R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proc. 11th Int'l Conf. Data Eng., pp. 3-14.

[4]. ChenM.S, Park J.S, and P.S. Yu, (1998)"Efficient Data Mining for Path Traversal Patterns," IEEE Trans. Knowledge and Data Eng., vol. 10, no. 2, pp. 209-221.

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

[6]. Chan.R, Yang.Q, and Shen.Y, (2003) "Mining High Utility Itemsets," Proc. IEEE Third Int'l Conf. Data Mining, pp. 19-26.

[7]. Erwin, A., Gopalan, R.P.: N.R. Achuthan,(2007)CTU-Mine: An Efficient High Utility Itemset Mining Algorithm Using the Pattern Growth Approach. In: IEEE CIT 2007. Aizu Waka- matsu, Japan.

[8]. Erwin.A, Gopalan R.P, and Achuthan N.R, (2008) "Efficient Mining of High Utility Itemsets from Large Data Sets," Proc. 12th Pacific-Asia Conf. Advances in Knowledge Discovery and Data Mining (PAKDD), pp. 554-561.

[9]. Han.J and Fu.Y, (1995) "Discovery of Multiple-Level Association Rules from Large Databases," Proc. 21th Int'l Conf. Very Large Data Bases, pp. 420-431.

[10]. Han.J, Dong.G, and Yin.Y, (1999) "Efficient Mining of Partial Periodic Patterns in Time Series Database," Proc. Int'l Conf. on Data Eng., pp. 106-115.

[11]. Han.J, Pei.J, and Yin.Y, (2000) "Mining Frequent Patterns without Candidate Generation," Proc. ACM-SIGMOD Int'l Conf. Manage- ment of Data, pp. 1-12.

[12]. LeeS.C, Paik.J, Ok.J, Song.I, and KimU.M, (2007) "Efficient Mining of User Behaviors by Temporal Mobile Access Patterns," Int'l J. Computer Science Security, vol. 7, no. 2, pp. 285-291.

[13]. Vincent S. Tseng, Bai-En Shie, Cheng-Wei Wu, and Philip S. Yu, Fellow, (2013) "Efficient Algorithms for Mining High Utility Itemsets from Transactional Database", IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 25, NO. 8.

[14]. Yun.U and Leggett.J.J,(2006) "WIP: Mining Weighted Interesting Patterns with a Strong Weight and/or Support Affinity," Proc. SIAM Int'l Conf. Data Mining (SDM '06), pp. 623-627.

[15]. ZakiM.J,(2000) "Scalable Algorithms for Association Mining," IEEE Trans. Knowledge and Data Eng., vol. 12, no. 3, pp. 372-390.