THE INTERNATIONAL JOURNAL OF SCIENCE & TECHNOLEDGE

Retrieval of High Utility Item Sets for Dynamic Dataset with the Mimic of Cache

N. Naveena Begum
M.E, Department of Computer Science, Arunai Engineering College, Thiruvannamalai, India
N. Parveena Begum
B.E, Department of Computer Science, Arunai Engineering College, Thiruvannamalai, India

Abstract:
Mining high utility item sets from a transferable data base assign to the find of item sets with the high utility such as profits. Even though the number of relevant algorithms has been proposed in recent years, they acquire the problem of generating the huge number of applicant item sets for the high utility item sets. Such a huge number of applicant item sets reduce the mining accomplishment in terms of the execution space and time requirement. The position may become poor when the database incorporates lots of high transactions. Here, we proposed two algorithms, such as utility pattern growth and utility pattern growth+, for mining the high utility item sets with a effective set action for pruning the applicant item sets. The high utility item sets information is keep up in tree based data structure, namely utility pattern tree (UP-Tree) such that applicant item sets can be produced comfortably with the only two data base scrutinize. The act of utility pattern growth and utility pattern growth+ is distinguished with the state of the art algorithms on more types of both the synthetic and real datasets. The Experimental outcomes shows that the algorithm proposed, particularly utility pattern Growth+, and not only decrease the number of applicant comfortably but also exceed other algorithms essentially in terms of runtime, exclusively when databases include lots of the high transactions.

***Key words**: Frequent itemset, high utility itemset, utility minimg, UP-tree*

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 mining, 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 Streaming databases and time series databases, and various application domains, such as bioinformatics, Web click-stream analysis, and mobile environments. Nevertheless, relative importance of each item is not considered in frequent pattern mining. To address this problem, weighted association rule mining was proposed. In this framework, weights of items, such as unit profits of items in transaction databases, are considered. With this concept, even if some items appear infrequently, they might still be found if they have high weights. However, in this framework, the quantities of items are not considered yet. Therefore, it cannot satisfy the requirements of users who are attentive in locate the item sets with huge sale profits, since the profits are composed of unit profits, i.e., weights, and purchased quantities. In this, utility emerges of mining as valuable case in data mining field. Mining high utility item sets from data bases refers to find the item sets with huge profits. Here, the meaning of itemset utility is interestingness, importance, or profitability of an item to customers. The items Utility in a transaction data base have two aspects:
1) The distinct importance items, is called as external utility, 2) the importance of items in transactions, which is called internal utility. Utility of an itemset is defined as the product of its external utility as well as internal. An item set is known as high utility item set if its utility is not less than the user-specified minimum utility threshold; otherwise, its known as low-utility item set. Mining huge utility item sets from data bases is a valuable task has an extra range of applications like website click stream analysis, business promotion in chain hypermarkets, cross marketing in retail stores, online e-commerce management, and mobile commerce environment planning, and even finding important patterns in biomedical applications. Recent years. With the eruptive increase of documents on the Internet, there are various confession applications. For example, the descriptive generation of snippets in the web search can aid users in further analyze and in a Question/Answer organization, in a question based arbitrary is much required to contribute information asked in questions. Another example is brief summaries in news services for news groups, which can help the users to understand the news articles in a group.
However, mining high utility itemsets from databases is not an easy task since downward closure property in frequent item set mining does not hold. In other words, pruning search space for high utility item set mining is complicate because a low-utility superset of an item set may be a huge utility item set. The naive method to address this problem is to enumerate all item sets from data bases by the exhaustion principle. Certainly, this method suffers from the problems of a large search space, especially when

data bases include lots of high transactions or a low minimum utility threshold is set. Hence, how to effectively shorten the finding space and efficiently observe all huge utility item sets with no miss is a crucial challenge in utility mining.

2. Related Work

Extensive studies have been proposed for mining frequent patterns. Among the issues of frequent pattern mining, the most famous are association rule mining and sequential pattern mining. One of the well-known algorithms for mining association rules is Apriori, which is the pioneer for efficiently mining association rules from large databases. Pattern growth-based association rule mining algorithms such as FP-Growth were afterward proposed. It is generally used that FP-Growth achieves the good performance than the Apriori based algorithms therefore it discovers frequent item sets without producing any applicant item set and scans data base just twice.

In the framework of frequent item set mining, the valuable of items are not considered to users. Thus, the topic called weighted association rule mining was brought to attention was proposed the concept of weighted items and weighted association rules . However, since the framework of weighted association rules does not have downward closure property, mining performance cannot be improved. To address this problem, we proposed the concept of weighted downward closure property. By using transaction weight, weighted support can not only reflect the importance of an itemset but also maintain the downward closure property during the mining process. There are also many studies that have developed different weighting functions for weighted pattern mining. Although weighted association rule mining considers the importance of items, in some applications, such as transaction databases, items' quantities in transactions are not taken into considerations yet. Thus, the issue of high utility itemset mining is raised and many studies have addressed this problem. Liu et al. proposed an algorithm named Two-Phase which is mainly composed of two mining phases. In phase I, it employs an Apriori-based level-wise method to enumerate HTWUIs. Candidate itemsets with length k are generated from length k-1 HTWUIs and their TWUs are computed by scanning the database once in each pass. After the above steps, the complete set of HTWUIs is collected in phase I. In phase II, HTWUIs that are high utility itemsets are identified with an additional database scan.

3. Proposed Algorithm

The framework of the proposed methods consists of three steps: 1) Scan the database twice to construct a global UP Tree
with the first two strategies; 2) recursively generate PHUIs from global UP-Tree and local UP-Trees by UP-Growth with the third and fourth strategies or by UP-Growth+ with the last two strategies identify actual high utility itemsets from the set of PHUIs . Note that we use a new term "potential high utility itemsets" to distinguish the patterns found by our methods from HTWUIs since our methods are not based on traditional TWU model. By our effective strategies, the set of PHUIs will become much smaller than the set of HTWUIs.

3.1. The Proposed Data Structure: UP-Tree

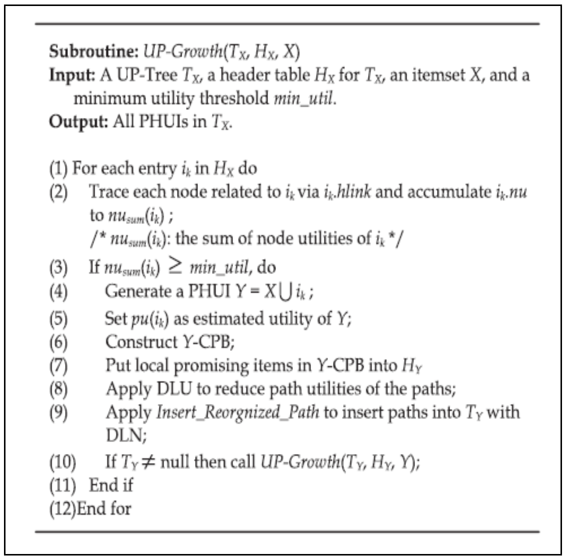
To facilitate the mining enforcement and prevent scanning real data base often, we use the condensed tree structure, named UP-Tree, to preserve the details of transactions and huge utility item sets. Two strategies are applied to minimize the overestimated utilities stored in the nodes of global UP-Tree. In following sections, the elements of UP-Tree are first defined. Next, the two strategies are introduced. Finally, how to construct an UP-Tree with the two strategies is illustrated in detail by a running example.

3.1.1. The Elements in UP-Tree

In an UP-Tree, each node N consists of N.nu, N.name, N.count, N.hlink, N.parent, and the set of child nodes. N.name is a item name of nodes. N.count is the node's support count. N.nu is the node's node utility, i.e., overestimated utility of the node. N.parent records the parent node of N. N.hlink is the node which link the points to a node which item name is a same as the N.name. A table named header table is assigned to further the traversal of UP-Tree. In header table, each entry records an item name, an overestimated utility, and the link. The link points to the least arrival of the node, which have the same entry item in the UP-Tree. By considering the links in header table and the nodes table in UP-Tree, the nodes having the same name can be traversed efficiently. In following sections, two strategies for decreasing the overestimated utility of each item while construction of the global UP-Tree is initiated.

3.2. The Proposed Mining Method: UP-Growth

After constructing a global UP-Tree, a basic method for generating PHUIs is to mine UP-Tree by FP-Growth. However too many candidates will be generated. Thus, we propose an algorithm UP-Growth by pushing two more strategies into the framework of FP-Growth. By the strategies, overestimated utilities of itemsets can be decreased and thus the number of PHUIs can be further reduced. In following sections, we first propose the two strategies and then describe the process of UP-Growth in detail by an example.



```
Subroutine: UP-Growth($T_X$, $H_X$, $X$)
Input: A UP-Tree $T_X$, a header table $H_X$ for $T_X$, an itemset $X$, and a
    minimum utility threshold min_util.
Output: All PHUIs in $T_X$.

(1) For each entry $i_k$ in $H_X$ do
(2)     Trace each node related to $i_k$ via $i_k$.hlink and accumulate $i_k$.nu
        to $nu_{sum}(i_k)$ ;
        /* $nu_{sum}(i_k)$: the sum of node utilities of $i_k$ */
(3)     If $nu_{sum}(i_k) \geq min\_util$, do
(4)         Generate a PHUI $Y = X \bigcup i_k$ ;
(5)         Set $pu(i_k)$ as estimated utility of $Y$;
(6)         Construct $Y$-CPB;
(7)         Put local promising items in $Y$-CPB into $H_Y$
(8)         Apply DLU to reduce path utilities of the paths;
(9)         Apply Insert_Reorgnized_Path to insert paths into $T_Y$ with
            DLN;
(10)        If $T_Y \neq$ null then call UP-Growth($T_Y$, $H_Y$, $Y$);
(11)    End if
(12)End for
```

Fig 1:The subrotuine of UP-Growth

3.3. An Improved Mining Method: UP-Growth$^+$

UP-Growth achieves better performance than FP-Growth by using DLU and DLN to decrease overestimated utilities of itemsets. However, the overestimated utilities can be closer to their actual utilities by eliminating the estimated utilities that are closer to actual utilities of unpromising items and descendant nodes. In this section, we propose an improved method, named UP-Growth+, for reducing overestimated utilities more effectively.

In UP-Growth, minimum item utility table is used to reduce the overestimated utilities. In UP-Growth+, minimal node utilities in each path are used to make the estimated pruning values closer to real utility values of the pruned items in database.

3.4. Efficiently Identify High Utility Itemsets

After finding all PHUIs, the third step is to identify huge utility item sets and the utilities from the set of PHUIs by scanning original database once. This step is called phaseII. However, in previous studies, two problems in this phase occur: 1) number of HTWUIs is too large; and (2) scanning original database is very time consuming. In our framework, overestimated utilities of PHUIs are smaller than or equal to TWUs of HTWUIs since they are reduced by the proposed strategies. Thus, the number of PHUIs is much smaller than that of HTWUIs.Therefore, in phase II, our method is much efficient than the previous methods.

4. System Design

Twitter API's can accessed only by the applications. Below we gave detail procedure for creating an API call from a Twitter App using OAuth:

- Application is also called as consumers. All applications are needed to record itself with the Twitter4. Within this process the application is distributed a consumer key, secret which the app must use by the authenticate person itself to Twitter.
- The app uses the consumer key, secret to produce a constant Twitter link to which a consumer is directed for authentication. The customer authorizes the app by authenticating them self to the Twitter. Twitter verifies the user's identity and issues an OAuth verifier also called a Pin.
- The customer provides this Pin to the app. The app uses a Pin to give REQ an "Access Token" and "Access Secret" unique to the user.
- Using the "Access Token" and "Access Secret", the application authenticates the customer in Twitter and distributes API calls at half of the user. The "Access Token" and "Access Secret" for a customer cannot modify and can be cached by the app for the future REQ. Therefore, this process needs only to perform once, and then it can be accomplished using the procedure GetUserAccessKeySecret.

4.1. Data Collection

A Twitter user's Tweets are also known as status messages. A Tweet can be at most 140 characters in length. Tweets can be published using a wide range of mobile and desktop clients and through the use of Twitter API. A special kind of Tweet is the retweet, which is created when one user reposts the Tweet of another user.

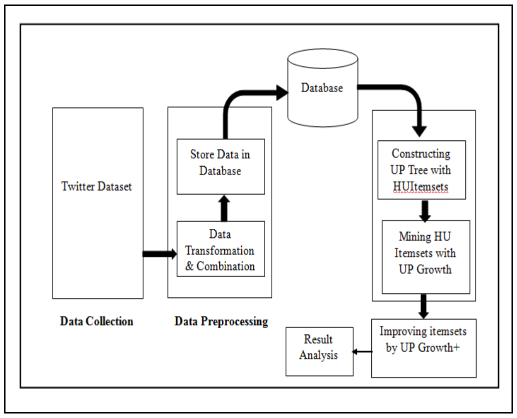

Database
Twitter Dataset
Store Data in Database
Data Transformation & Combination
Constructing UP Tree with HUItemsets
Mining HU Itemsets with UP Growth
Improving itemsets by UP Growth+
Result Analysis
Data Collection
Data Preprocessing

Fig 2. System Architecture

4.2. Data Pre-Processing

The next important step in our project is data pre-processing. Thus, data is to be pre-processed so as to remove the noisy and unwanted data. Pre-processing means removing the other unwanted parameters from the dataset. Data pre-processing includes following processes:

- *Data cleansing:* It is also known as data cleaning, it is a case in which the noise data, irrelevant data is removed from the collection. Data cleaning routines perform to fill in the missed value, and smooth out noise while authorize outliers, and then correct discrepancy in the data.
- *Data transformation:* It is also called as data consolidation; it is a case in which the choose data is modified into forms suitable for the mining steps. Data transformation can involve the following:

Smoothing: It is a works to eliminate all the noise from data. Such techniques are binning, regression, and clustering.
Aggregation: This step used in the creating a data cube for identify of the data at multiple granularities.
Generalization: Here low-level data are revoking by higher-level idea through the use of idea hierarchies.

4.3. Constructing UP Tree

In this module we are constructing Compact tree structure, named UP-Tree (Utility Pattern Tree) is used to avoid scanning original database repeatedly and to facilitate the mining performance, to keep up the details of huge utility item sets and transactions. For reducing the size here the tree is compacting (closely packed together).

4.3.1. The Elements in UP-Tree

In a UP-Tree, each node N Consists of N.count, N.name, N.hlink, N.parent and set of child nodes. N.name is the node's item name. The node's support count is N.count. The node's node utility is N.nu i.e., the utility of the node overestimated. N.parent records the parent node of N. N.hlink is a node link which the points to a node item name is same such as N.name. The table named header table is assigned to further the UP-Tree traversal. Each entry records an item name in header table, link and an overestimated utility. The last occurrence of the node is pointed by the link node, which have the same item entry in UP-Tree.

4.3.2. Inserting elements in UP Tree

With two scans of the original database the construction of a global UP-Tree can be done. TU of each transaction is computed in the first scan. TWU of each unique item is also gathering at the same time. Promising and unpromising are two nodes of a node. More profits are given by selecting promising nodes and others as unpromising nodes. High utility itemsets are only the supersets of the itemsets and the less quality is given by subsets of the item. Transactions are inserted into a UP-Tree during the second scan of database.

4.4. Implementing UP Growth

Mining UP-Tree by UP-Growth: by two scans of a conditional pattern base conditional UP-Tree can be constructed. For the first scan, by summing the path utility for each item in the conditional pattern base local unpromising and promising items are learned. During the second scan of the conditional pattern base reduce overestimated utilities DLU is applied. From the path and its path utility items and their estimated utilities are eliminated when a path is retrieved. In the conditional pattern base by the descending order of path utility of the items the path is reorganized. During inserting reorganized paths into a conditional UP-Tree DLN is applied.

4.5. Implementing UP Growth+

After introducing the modification of global UP-Tree, now we address the processes and two improved strategies of UP Growth+, named DNU and DNN. When a local UP Tree is being constructed, minimal node utilities can also be acquired by the same steps of global UP-Tree. In the mining process, when a path is retrieved, minimal node utility of each node in the path is also retrieved.

5. Discussion

- *Strategy 1.* DNU. Discarding local unpromising items and their estimated Node Utilities from the paths and path utilities of conditional pattern bases.
- *Strategy 2.* DNN. Decreasing local Node utilities for the nodes of local UP-Tree by estimated utilities of descendant nodes.

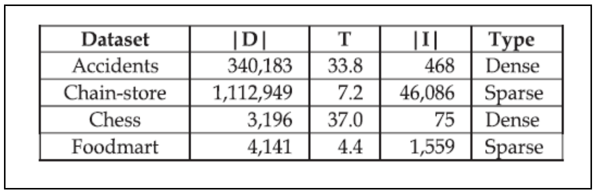
By the above two strategies, the subroutines Insert_Reorganized_Path and UP-Growth in can be modified to the two new subroutines for UP-Growth+. Insert Reorganized Path mnu is an improved version of Insert Reorganized Path with following modifications. When a new node Nix is created in Line 1, the element, minimal node utility, is added into Nix and set Nix: mnu=∞ initially. Then, Nix: mnu is checked by inserting the procedure "If Nix: mnu > mnu (ix; pj), set Nix: mnu to mnu(ix; pj)" between Lines 2 and 3 of the subroutine Insert_Reorgnized_Path.

6. Results

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.

6.1. Performance Comparison on Different Data Sets

In this part, we show the performance comparison on three real data sets: dense data set Chess and sparse data sets Chain-store and Food mart. First, we show the results on real dense data set Chess. we can observe that the performance of proposed methods substantially out performs that of previous methods. The runtime of IHUPT&FPG is the worst, followed by UPT&FPG, UPT& UPG, and UPT&UPG+ is the best. The main reason is the performance of IHUPT&FPG and UPT&FPG is decided by the number of generated candidates. Runtime of the methods is just proportional to their number of
candidates, that is, the more candidates the method
produces, the greater its execution time.



Dataset	|D|	T	|I|	Type
Accidents	340,183	33.8	468	Dense
Chain-store	1,112,949	7.2	46,086	Sparse
Chess	3,196	37.0	75	Dense
Foodmart	4,141	4.4	1,559	Sparse

Table 1

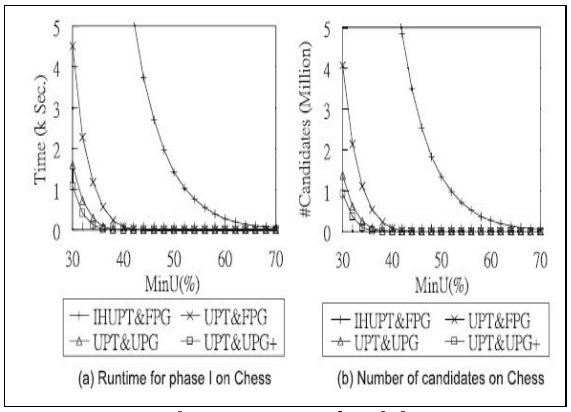


Fig 3: Characteristics of real data sets
Fig 4: Performance comparison graph

Experimental results in this section show that the proposed methods outperform the state-of-the-art algorithms almost in all cases on both real and synthetic 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 of IHUP-Tree since DGU and DGN effectively decrease overestimated utilities while creating of the global UP-Tree
Second, UP-growth and UP-Growth+ generate much fewer candidates than FP-growth since DLU, DLN, DNU, and DNN are applied during the construction of local UP Trees. By the proposed algorithms with the strategies, generations of candidates in phase I can be more efficient since lots of useless candidates are pruned.

7. Conclusion

We have proposed two efficient algorithms named UP-Growth+ and UP-Growth for mining the high level utility item sets from the transaction data bases. A data structure named UP-Tree was proposed for maintaining the information of high utility itemsets. PHUIs can be efficiently generated from UP-Tree with only two database scans. Moreover, we developed several strategies to decrease overestimated utility and enhance the performance of utility mining. In the experiments, both real and synthetic data sets were used to perform a thorough performance evaluation. Results show that the strategies considerably improved performance by reducing both the search space and the number of candidates. Moreover, the proposed algorithms, especially UP-Growth+, outperform the state of- the-art algorithms substantially especially when data bases include lots of deep transactions or a low minimum, utility threshold is used.

8. References

1. S. Tseng, Shie, and S. Yu, Fellow, IEEE,"Efficient Algorithms For Mining Huge Utility Item sets From sale Data bases," IEEE transaction on knowledge and data engineering, vol.25. No.8, Aug 2013.
2. C. Ahmed, B.-S. Jeong, and Y.-K. Lee, "Efficient Tree Structures for the Huge Pattern of Utility Mining in the evolving Data bases," IEEE Trans. Knowledge and Data Eng., vol. 21, no. 12, pp. 1708-1721, Dec. 2009.
3. M.-S. Chen, J.-S. Park, "Efficient Data Mining for Path Traversal Patterns," IEEE Trans. Knowledge and Data Engineering. pp. 209-221, Mar. 1998.
4. H.Y. Huang, Y.J. Liu, and S. Lee, "Fast and Memory Efficient Mining of Huge Utility Item sets in Database Stream," Proc. IEEE Eighth Int'l Conf. on Data Mining, pp. 881-886, 2008.
5. Jian-Ping Mei, Lihui Chen. "A Fuzzy approach for multitype relational data clustering,"IEEE transactions on Fuzzy systems, vol 20, No.2, April 2012.
6. Asmaa Benghabrit, Brahim Ouhbi,Hicham Behja, "Text clustering using statistical and semantic data,"IEEE trans. Pattern analysis,vol 22,no.8,2013.
7. Jian-Ping Mei and Lihui Chen, "An enhanced fuzzy c-means clustering using relational information, "proc.eigth international conference on fuzzy systems and knowledge discovery, vol 39, 2011.
8. Yves Lechevalliar, Filipe M. de Melo, "A relational fuzzy C-means clustering algorithm based on multiple dissimilarity Matrices", IEEE transaction on fuzzy systems.vol 19, 2010.
9. Zhang Pei-ying, LI Cun-he, "Automatic text summarization based on sentences clustering and extraction", IEEE Tran, pattern analysis, vol 22, no.34, 2009
10. Naoki Haga, Katsuhiro Honda and Akira Notsu, "Linear Fuzzy Clustering of Relational Data Based on Extended Fuzzy c-Medoids", IEEE international conference on Fuzzy systems, 20
11. W. Wang, J. Yang, and P. Yu, "Efficient Mining of Weighted Association Rules (WAR)," Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD '00), pp. 270-274, 2000.
12. H.J. Hamilton L. Geng, "A Unified Framework for Utility-Based Measures for Mining Item sets," Proc. ACM SIGKDD Second Workshop Utility-Based Data Mining, pp. 28-37, Aug. 2006.
13. Y.S. Lee, S.J. Yen, "Mining Huge Utility Quantitative Association Rules." Proc. Ninth Int'l Conf. Data Warehousing and Knowledge Discovery (DaWaK), pp. 283-292, Sept. 2007.
14. S.J. Yen, Y.S. Lee, C.K. Wang, C.W. Wu, and L.-Y. Ouyang, "The Studies of Mining Frequent Patterns Based on Frequent Pattern Tree," Proc. 13th Pacific-Asia Conf. Advances in Knowledge Discovery and Data Mining (PAKDD), vol. 5476, pp. 232-241, 2009.
15. C.-H. Yun and M.-S. Chen, "Using Pattern-Join and Purchase-Combination for Mining Web Transaction Patterns in an Electronic Commerce Environment," Proc. IEEE 24th Ann. Int'l Computer Software and Application Conf., pp. 99-104, Oct. 2000