A Novel Discovery of Highly Improved Dataset Using Improved UP Growth Algorithm

Prof.Rashmi Deshpande
Department of Information Technology,
Department of Information Technology, SCOE,
. Sadumbare, Pune.
Rashmi2810@gmail.com

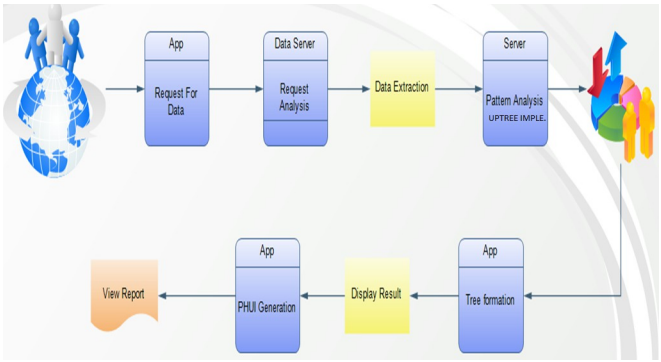
Anuja Palhade
Department of Information Technology,
Department of Information Technology, Savitribai Phule Pune University, Pune, Maharashtra, India.
anujapalhade336@gmail.com

Abstract: ***Efficient discovery of frequent item sets in large datasets is a crucial task of data mining. In recent years, several approaches have been proposed for generating high utility patterns; they arise the problems of producing a large number of candidate item sets for high utility item sets and probably degrade mining performance in terms of speed and space. In phase I of project, the framework of UP-Tree follows three steps: (i). Construction of UP-Tree. (ii).Generate PHUIs from UP-Tree. (iii). Identify high utility item sets using PHUI. The global UP-Tree contains many sub paths. Each path is considered from bottom node of header table. This path is named as conditional pattern base (CPB).***
Recently proposed compact tree structure, UP-Tree, maintains the information of transactions and item sets, facilitate the mining performance and avoid scanning original database repeatedly. In this project, UP-Tree (Utility Pattern Tree) is adopted, which manage them in an efficient data structured way. Applying UP-Tree to the UP-Growth takes more execution time for Phase II. Hence this project presents modified algorithm aiming to reduce the execution time by effectively identifying high utility item sets.

Keywords: Itemset, Utility, UP-tree, UP-Growth, FP-Growth.

1. Introduction

The usefulness of an itemset is characterized as a utility constraint. That is, an itemset is interesting to the user only if it satisfies a given utility constraint. We show that the pruning strategies used in previous itemset mining approaches cannot be applied to utility constraints. In response, we identify several mathematical properties of utility constraints. Then, two novel pruning strategies are designed. Two algorithms for utility based itemset mining are developed by incorporating these pruning strategies. The algorithms are evaluated by applying them to synthetic and real world databases.
In this project, we consider the exceptional prerequisites of indexing to focus the execution of diverse strategies in information stream handling situations. Stream indexing has principle contrasts with methodologies in customary databases. Additionally, we look at information stream indexing models systematically that can give a suitable strategy to stream indexing. Association rules mining are one of the most widely used techniques in data mining and knowledge discovery and has tremendous applications in business, science and other domains. For example, in the business, its applications include retail shelf management, inventory predictions, supply chain management. The main objective of this is to identify frequently occurring patterns of itemsets. It first finds all the itemsets whose co-occurrence frequency are beyond a minimum support threshold, and then generates rules from the frequent itemsets based on a minimum confidence threshold. Traditional model treat all the items in the database equally by only considering if an item is present in a transaction or not. Frequent itemsets may only contribute a small portion of the overall profit, whereas non-frequent itemsets may contribute a large portion of the profit.



Here we have shown the architectural flow of the project view. In reality, a retail business may be interested in identifying its most valuable customers (customers who contribute a major fraction of the profits to the company). A sequence of web pages visited by a user can be defined as a transaction. Since the number of visits to a webpage and the time spent on a particular webpage is different between different users, the total time spent on a page by a user can be viewed as utility. The website designers can catch the interests or behavior patterns of the customers by looking at the utilities of the page combinations and then consider re-organizing the link structure of their website to cater to the preference of users. Frequency is not sufficient to answer questions, such as whether an itemset is highly profitable, or whether an itemset has a strong impact. Utility mining is likely to be useful in a wide range of practical applications.

2. Related Work

As of late, the administration and handling of information streams has turned into a subject of dynamic research in a few fields of software engineering, for example, dispersed frameworks, database frameworks, and information mining. An information stream can be considered a transient, persistently expanding arrangement of information. In information streams' applications, due to web observing,

offering an explanation to the client's questions ought to be time and space effective.

Regular example mining has been concentrated on broadly and has numerous helpful applications. Be that as it may, continuous example mining frequently produces an excess of examples to be genuinely efficient or powerful. In numerous applications, it is sufficient to produce and analyze successive examples with a sufficiently decent estimate of the help recurrence rather than in full accuracy. Such a conservative yet "close-enough" regular example base is known as a dense continuous example base. In this paper, we propose and look at a few options for the configuration, representation, and execution of such consolidated regular example bases. A few calculations for figuring such example bases are proposed. Their adequacy at example layering and routines for efficiently figuring them is researched. A methodical execution study is directed on various types of databases, and exhibits the viability and efficiency of our methodology in taking care of incessant example mining in vast databases. We introduce two new calculations for tackling this issue that are in a broad sense not quite the same as the known algorithms. Experimental assessment demonstrates that these calculations beat the known calculations by variables going from three for little issues to more than a request of extent for huge issues. We additionally demonstrate how the best peculiarities of the two proposed calculations can be joined into a mixture calculation, called Apriori hybrid. Scale-up trials demonstrate that Apriori hybrid scales straightly with the quantity of exchanges. Apriori hybrid likewise has excellent scale-up properties as for the exchange size and the quantity of things in the database.

The objective was to deliver a vigorous, efficient, and usable apparatus for that undertaking that can both be utilized by specialists and utilized for evaluation of researching the area. We experimentally confirm the excellent execution of the calculation and its capacity to handle idea drift. The objective was to deliver a vigorous, efficient, and usable apparatus for that assignment that can both be utilized by specialists and utilized for evaluation of research in the area. The proposed strategy approximates the checks of item sets from certain recorded synopsis data without examining the info stream for every item set. Together with an imaginative strategy called progressively approximating to choose parameter-values appropriately for distinctive item sets to be approximated, our system is versatile to streams with diverse conveyances. Our tests demonstrate that our calculation performs much better in enhancing memory use and mining just the latest examples in less time execution with really precise.

3. Procedure/Methodology

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 item sets can be decreased and thus the number of PHUIs can be further reduced.

1. Apriori: uses a generate-and-test approach – generates candidate itemsets and tests if they are frequent in which Generation of candidate itemsets is expensive (in both space and time) and Support counting is expensive Subset checking (computationally expensive), Multiple Database scans (I/O)

2. FP-Growth: allows frequent itemset discovery without candidate itemset generation. Two step approach:

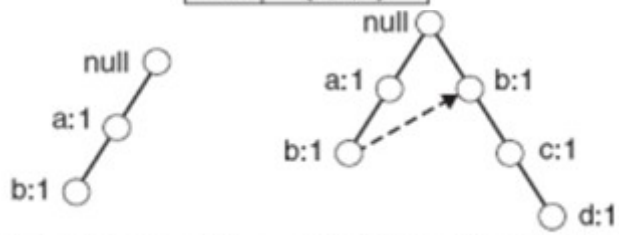
Step 1: Build a compact data structure called the FP-tree Built using 2 passes over the data-set.

Step 2: Extracts frequent itemsets directly from the FP-tree

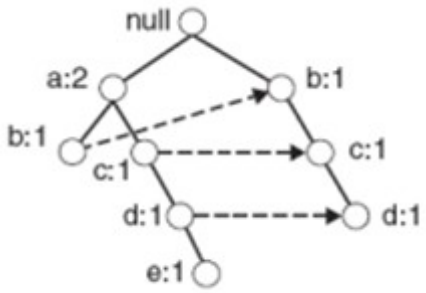
We see stepwise how we are constructing FP tree:

Transaction Data Set

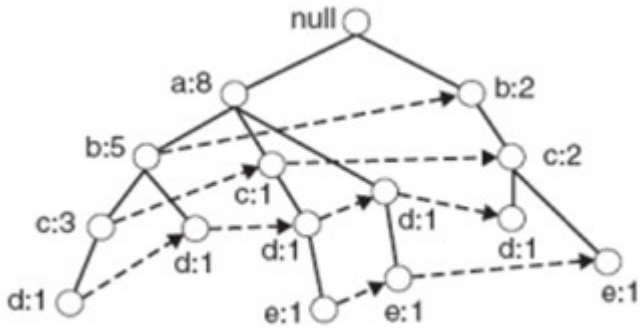
TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}



(i) After reading TID=1 (ii) After reading TID=2



(iii) After reading TID=3



(iv) After reading TID=10

UP-Growth achieves better performance than FP-Growth by using DLU and DLN to decrease overestimated utilities of item sets. 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. We build UP-Growth outperforms UP-Growth although they have tradeoffs on memory usage. The reason is that UP-Growth+ utilizes minimal node utilities for further decreasing overestimated utilities of item sets. Even though it spends time and memory to check and store minimal node utilities, they are more effective especially when there are many longer transactions in databases. The reason is that UP-Growth+ utilizes minimal node utilities for further decreasing overestimated utilities of itemsets. Even though it spends time and memory to check and store minimal node utilities, they are more effective especially when there are many longer transactions in databases.

In contrast, UP-Growth performs better only when min_util is small. This is because when number of candidates of the two algorithms is similar, UP-Growth+ carries more computations and is thus slower. Finally, high utility itemsets are efficiently identified from the set of PHUIs which is much smaller than HTWUIs generated by IHUP. By the reasons mentioned above, the proposed algorithms UP-Growth and UP-Growth+ achieve better performance than IHUP algorithm.til is small. This is because when number of candidates of the two algorithms is similar, UP-Growth+ carries more computations and is thus slower. Finally, high utility itemsets are efficiently identified from the set of PHUIs which is much smaller than HTWUIs generated by IHUP. By the reasons mentioned above, the proposed algorithms UP-Growth and UP-Growth+ achieve better performance than IHUP algorithm.

3.1 Algorithm

To address this issue, we propose two novel algorithms as well as a compact data structure for efficiently discovering high utility item sets from transactional databases. Major contributions of this work are summarized as follows:

1. Two algorithms, named utility pattern growth (UP Growth) and UP-Growth+ , and a compact tree structure, called utility pattern tree (UP-Tree), 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 UP-Tree efficiently with only two scans of original databases.
2. Several strategies are proposed for facilitating the mining processes of UP-Growth and UP-Growth by maintaining only essential information in UP-Tree. 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 the search space. The proposed strategies can not only decrease the overestimated utilities of PHUIs but also greatly reduce the number of candidates.

Fp-Growth Algorithm

Fp-Growth approach is based on divide and conquers strategy for producing the frequent item sets. FP growth is mainly used for mining frequent item sets without candidate generation. Major steps in FP-growth is-

Step1- It firstly compresses the database showing frequent item set in to FP-tree. FP-tree is built using 2 passes over the dataset.

Step2: It divides the FP-tree in to a set of conditional database and mines each database separately, thus extract frequent item sets from FP-tree directly. It consist of one root labeled as null, a set of item prefix sub trees as the children of the root, and a frequent .item header table. Each node in the item prefix sub tree consists of three fields: item-name, count and node link where---item-name registers which item the node represents; count registers the number of transactions represented by the portion of path reaching this node, node link links to the next node in the FP- tree. Each item in the header table consists of two fields---item name and head of node link, which points to the first node in the FP-tree carrying the item name.

```
Input: Constructed FP tree
Output: Complete set of frequent pattern
Method: Call FP growth(FP-tree, null)
Procedure FP-growth(tree, a)
{
        1) if tree contain single path P then
        2) for each combination generate B U a with support =minimum
           support of nodes in B
        3) else for each header ai in the header of tree do{
        4) generate pattern B=ai U a with support=ai.support;
        5) construct B.s conditional pattern base and then B.s conditional
           FP-tree Tree B
        6) if Tree B=null
        7) Then call FP-growth(Tree B,B);
}
```

3.2 Mathematical Model

We first define the notations for utility mining in streaming data environments and then address the problem statement of this work. A data stream DS is composed of a continuous set of transactions denoted by {Tid1, Tid2... Tidn ...}. A transaction Tidk is denoted as {itemsetk, tk}, where itemsetk is the items appeared in the transaction Tidk and tk is the time when itemsetk appeared in DS. Itemsetk is composed of a set of items and their purchased numbers, which is denoted by {(x1, q1), (x2, q2)... (xp, qp)}, where xr is the item and qr is the purchased number of xr in Tidk. I = {i1, i2, ...,im} is the set of items. An itemset is a subset of I. Every item in the data stream has its own unit profit. The unit profit of an item x, which is recorded in a utility table, is denoted by p(x).

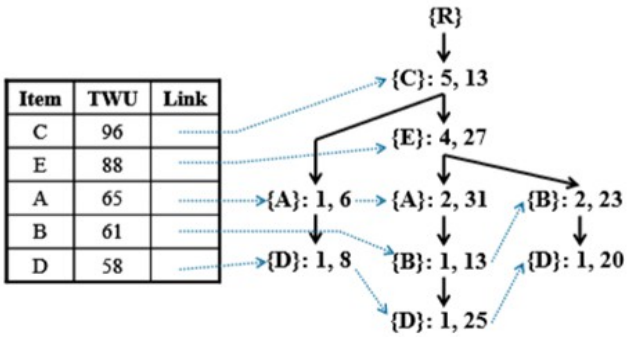
- $u(i_p, T_q)$, *utility*, the quantitative measure of utility for item i_p in transaction T_q, is defined as $o(ip,Tq)\times s(ip)$. For example, $u(A, T_8) = 3 \times 3$, in Table 1.
- $u(X, T_q)$, *utility of an itemset X in transaction* T_q, is defined as $\sum_{ip\in X} u(ip,Tq)$, where $X = \{i_1, i_2, \ldots, i_k\}$ is a k-itemset, $X \subseteq T_q$ and $1 \le k \le m$.
- $u(X)$, *utility of an itemset X*, is defined as $\sum_{T_q\in D\wedge X\subseteq T_q} u(X,Tq)$.

4. Expected Result

The Irrelevant features, along with redundant features, severely affect the accuracy of the learning machines. Thus, feature subset selection should be able to identify and remove as much of the irrelevant and redundant information as possible. Moreover, "good feature subsets contain features

highly correlated with (predictive of) the class, yet uncorrelated with (not predictive of) each other. Keeping these in mind, we develop a novel algorithm which can efficiently and effectively deal with both irrelevant and redundant features, and obtain a good feature subset. Different types of both real and synthetic data sets are used in a series of experiments to compare the performance of the proposed algorithms with the state-of-the-art utility mining algorithms. Experimental results show that UP-Growth and UP-Growth+ outperform other algorithms substantially in terms of execution time, especially when databases contain lots of long transactions or low minimum utility thresholds are set.

In this section, we describe the details of UP-Growth for efficiently generating PHUIs from the global UP-Tree with two strategies, namely DLU (Discarding local unpromising items) and DLN (Decreasing local node utilities). Although strategies DGU and DGN described in previous section can effectively reduce the number of candidates in phase I, they are applied during the construction of the global UP-Tree and cannot be applied during the construction of the local UP-Tree. The reason is that the individual items and their utilities are not maintained in the conditional pattern base. We cannot know the utility values of the unpromising items in the conditional pattern base. To overcome this problem, a naïve approach is to maintain the utilities of the items in the conditional pattern base. However, this approach may be impractical since it consumes lots of memory usages. Instead of maintaining exact utility values of the items in the conditional pattern base, we maintain a minimum item utility table, abbreviated as MIUT, to maintain the minimum item utility for all global promising items.



A UP-Tree by applying strategies DGU and DGN.

Minimum item utility table

Item	A	B	C	D	E
Minimum item utility	5	4	1	2	3

5. Conclusion

Thus here we propose high utility mining approach rather than using the traditional approach based on frequency. We proposed a novel framework, namely Generation of maximal high Utility Itemsets from Data streams (GUIDE), which efficiently mines maximal high utility itemsets from large datasets. The proposed compact data structure UP-Tree is cooperated for storing essential information in data streams. We first discover compact forms of high utility itemsets from data streams. GUIDE is an effective framework which meets the needs of data stream mining. The tree structure maintains essential information for the mining processes. GUIDE generates patterns which are not only high utility but also maximal from the datasets.

References

[1] Adinarayana reddy B, O SrinivasaRao,MHM Krishna Prasad, "*An Improved UP-Growth High Utility Itemset Mining*" ,International Journal of Computer Applications (0975 – 8887) Volume 58– No.2, November 2012

[2] P.Abinaya, Dr.J.Sutha" *Effective Feature Selection for High Dimensional Data using Fast Algorithm*" Dept. of CSE, Sethu Institute of Technology, Affiliated to Anna University, Kariapatti, Tamilnadu, India. International Journal of Advanced Research in Vol. 2 Issue Special 1 Jan-March 2014

[3] R. Agrawal and R. Srikant, "*Fast Algorithms for Mining Association Rules*," in Proceedings of the 20th International Conference on Very Large Data Bases, pp. 487-499, September 2014.

[4] Arati W. Borakar "*Utility Mining Algorithm for High Utility Item sets from Transactional Databases*", IOSR Journal of Computer Engineering (IOSR-JCE) Issue 2, Ver. V (Mar-Apr. 2014), Department of CSE, Pune University, India.

[5] Bifet, G. Holmes, B. Pfahringer, and R. Gavaldà, "*Mining Frequent Closed Graphs on Evolving Data Streams*" in Proc. of the 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 2011), pp. 591-599, San Diego, CA, USA, August, 2011.

Author Profile

Anuja Palhade received her B.E. degree in Information Technology from Hi tech Institute of Technology in Dr.Babasaheb Ambedkar Marathwada University, Aurangabad in 2010. Now pursuing the M.E degree From Siddhant College of engineering, in Savitribai Phule University of pune.