An Efficient Algorithm for Identification of Most Valuable Itemsets from WebTransaction Log Data

Litty Tressa George*, Asha Rose Thomas**
*(Computer Science and Engineering, Adi Shankara Institute of Engineering and Technology, Kalady, India)
** (Computer Science and Engineering, Adi Shankara Institute of Engineering and Technology, Kalady, India)

Abstract:
Web Utility mining has recently been a blooming topic in the field of data mining and so is the web mining, an important research topic in database technologies. Thus, the web utility mining is effective in not only discovering the frequent temporal web transactions & generating high utility itemsets, but also identifying the profit of webpages. For enhancing the web utility mining, this study proposes a mixed approach to the techniques of web mining, temporal high utility itemsets & on-shelf utility mining algorithms, to provide web designers and decision makers more useful and meaningful web information. In the two phases of the algorithm, we came out with the more efficient and modern techniques of web & utility mining in order to yield excellent results on web transactional databases. Mining most valuable itemsets from a transactional dataset refers to the identification of the itemsets with high utility value as profits. Although there are various algorithms for identifying high utility itemsets, this improved algorithm is focused on online shopping transaction data. The other similar algorithms proposed so far arise a problem that is they all generate large set of candidate itemsets for Most Valuable Itemsets and also require large number database scans. Generation of large number of itemsets decreases the performance of mining with respect to execution time and space requirement. This situation may worse when database contains a large number of transactions. In the proposed system, information of valuable itemsets are recorded in tree based data structure called Utility Pattern Tree which is a compact representation of items in transaction database. By the creation of Utility Pattern Tree, candidate itemsets are generated with only two scans of the database. Recommended algorithms not only reduce a number of candidate itemsets but also work efficiently when database has lots of long transactions.

Keywords — Utility Mining, Itemset utility, Valuable itemsets, Most valuable itemsets.

INTRODUCTION
Extracting or “mining” knowledge from large amounts of data stored in databases, data warehouses or any other information repositories is called data mining. A dataset can be defined as any named group of records upon which datamining is performed. Groups of items that appear together in any transaction datasets can be called as Itemsets. Frequent itemsets are set of items that appear in a data set frequently. For example, a set of items, like milk and bread, appears frequently together in a transaction data set. Finding such frequent patterns plays an essential role in data classification, mining associations, correlations, identifying many interesting relationships among data. Thus, frequent pattern mining has turn out to be an imperative data mining task and a focused theme in data mining research. Frequent pattern mining searches for recurring relationships in a given data set. The problem with frequent pattern mining was that, individual importance of each pattern is not considered. Infrequent itemset mining which is a type of frequent pattern mining, where the patterns are itemsets, only the occurrence of items are considered. Unit profits as well as purchased quantities of the items were not taken...
into consideration. Individual importance of each item is not considered infrequent itemset mining. Therefore, it cannot satisfy the requirements of users who are interested in discovering the item sets with high sales profits, since the profits in these unit profits or weights, or even purchased quantities. For example, each item in a supermarket has different price or profit and multiple copies of an item can be sold in a transaction. Hence, the most profitable item sets cannot be found in those frameworks since profit of an item set can be calculated by multiplying unit profit of each item in the item set by the quantities in transactions including the item set. To find the most valuable item sets, both the importance and quantity of each item have to be reflected. In view of this, utility mining arises as a main topic in data mining field.

I. STATE OF ART

A. Frequent Pattern Mining

Frequent pattern mining is concerned with the mining of most frequently appearing patterns within a dataset. Herethe problem is to discover the complete set of patterns satisfying a minimum support in the transaction database. The entire dataset is pruned on the basis of downward closure property to identify the infrequent patterns. The downward closure property states that if a pattern is infrequent, then all of its super patterns must also be infrequent. The Apriori algorithm [2] was the first solution to mine frequent patterns. It is a breadth first search algorithm. The drawback was that it suffers from a level-wise candidate generation and test problem and also it needs several database scans. That is for the first database scan, the Apriori discovers all the one-element item sets and on the basis of that produces the candidates for two-element frequent item sets. In the second database scan, Apriori identifies all of the two-element frequent item sets, and based on that, generates the candidates for three-element frequent item sets and so on. Thus if the size of the largest frequent item set is 'n', then Apriori needs 'n' database scans. In order to overcome this limitation, later FP growth algorithm was proposed. It was a depth first search algorithm. It needed only two database scans for generating frequent patterns without any candidate generation.

B. Weighted Frequent Pattern Mining

Weighted frequent pattern mining deals with binary databases where frequency of each item in each transaction can be either 1 or 0. W. Wang [5] et al in proposed weighted association rule mining algorithm WAR. In WAR, we discover first frequent item sets and the weighted association rules for each frequent item set are generated. The weight of a pattern ‘p’ is defined as the proportion of the sum of all its items’ weight value to the length of ‘p’. The foremost challenge in this area is the weighted frequency of a pattern does not satisfy the downward closure property. Thus the mining performance cannot be improved. In order to address this problem, Cai et al. [4] first proposed the concept of a weighted downward closure property. By means of the transaction weight, weighted support can not only reveal the importance of an item set but also retains the downward closure property in the course of the mining process. 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. Also the non-binary occurrence of items in transactions is not considered. Thus, the matter of high utility item set mining came into scene.

C. High Utility Pattern Mining

High utility pattern mining focus on mining the highly utilized or most valuable patterns (it can be item sets) from a dataset. Identification of only frequent patterns in a database cannot achieve the requirement of identifying the most valuable item sets that add to the majority of the total
profits in a retail business. This gives the inspiration to develop a mining model to determine those itemsets contributing to the majority of the profit. To find the most valuable itemsets, both the importance and quantity of each item have to be reflected. Identification of the item sets with high utilities is called as High Utility Item set Mining. Here, the meaning of item set utility is interestingness, importance, profitability or any other user relevant feature exhibited by the items of the dataset under consideration.

1) Two Phase Algorithm: Liu et al. proposed an algorithm named Two Phase [6] which is mainly consists of two mining phases. In phase I, it includes an Apriori-based level-wise method which is a breadth first search strategy, to enumerate High Transaction Weighted Utility Itemsets (HTWUI). Candidate itemsets which are of length k are generated from length k-1 HTWUIs, and their TWUs are calculated by scanning the database once in each pass. After these steps, the whole set of HTWUIs is collected in phase I. In phase II, HTWUIs that are high utility itemsets are recognized with an additional database scan. The authors have well-defined the transaction-weighted utilization (twu) and by that they proved it is possible to sustain the downward closure property. In the initial database scan, the algorithm discovers all the one-element transaction-weighted utilization itemsets, and based on that result, it produces the candidates for two element transaction-weighted utilization itemsets. In the second database scan, it discovers all the two-element transaction-weighted utilization itemsets, and based on that result, it creates the candidates for three-element transaction-weighted utilization itemsets, and so on. At the final scan, the Two-Phase algorithm discovers the real high utility itemsets from the high transaction-weighted utilization itemsets. This algorithm suffers from the difficulty of the level-wise candidate generation and test methodology. Although two-phase algorithm decreases search space by using TWDC property, it still produces too many candidates to obtain HTWUIs and needs multiple database scans.

2) Incremental High Utility Pattern (IHUP) Algorithm: To efficiently produce HTWUIs in phase I and avoid scanning database multiple times, Ahmed et al. [9] proposed a tree-based algorithm, called IHUP, for mining high utility itemsets. It includes an IHUP-Tree to maintain the information of high utility itemsets and transactions. Every node in IHUP-Tree consists of an item name, a support count, and a TWU value. The structure of the algorithm consists of three parts: (1) The construction of IHUP-Tree, (2) the generation of HTWUIs and (3) identification of high utility itemsets. In step 1, items in the transaction are reorganized in a fixed order like lexicographic order, support descending order or TWU descending order. Then, the reorganized transactions are put into the IHUP-Tree. In step 2, HTWUIs are produced from the IHUP-Tree by applying the FP-Growth algorithm. Thus, HTWUIs in phase I can be retrieved more capably without producing candidates for HTWUIs. In step 3, high utility itemsets and their utilities are recognized from the set of HTWUIs by scanning the original database once. Even though IHUP finds HTWUIs without producing any candidates for HTWUIs and achieves a better performance than Two-Phase, it still results in too many HTWUIs in phase I since the overestimated utility calculated by TWU is too long. IHUP and Two-Phase produce the similar number of HTWUIs in phase I, since they use Transaction-Weighted Utilization mining (TWU) model to overestimate the utilities of the itemsets. However, this model may underestimate too many low utility itemsets as HTWUIs and produce too many candidate itemsets in phase I. Such a huge number of HTWUIs reduces the mining performance in phase I in terms of execution time and memory consumption. Besides, the number of HTWUIs in phase I also decrease the performance of the algorithms in phase II since the more HTWUIs are generated in phase I, the more execution time is required for recognizing high utility itemsets in phase II.
3) Utility Pattern Growth (UP) Algorithm: The framework of the UP Growth method proposed by V.S.Tseng [12] consists of three parts: (1) construction of UP Tree, (2) generation of potential high utility itemsets from the UP-Tree by UP-Growth, and (3) identification of high utility itemsets from the set of potential high utility itemsets. In this algorithm, a new term called potential high utility itemsets (PHUIs) is used to distinguish the discovered patterns found by up growth approach from the HTWUIssince our approach is not based on the traditional framework of transaction-weighted utilization mining model. UP Growth efficiently generates PHUIs from the global UPTree with two strategies, namely DGU (Discarding Global Unpromising items) to eliminate the low utility items and their utilities from the transaction utilities and DGN (Decreasing Global Node utilities) node utilities which are nearer to UP-Tree root node are effectively reduced. Even though it successfully generates the PHUI's by the above two strategies, the problem is that there will be more no of PHUI's.

II. METHODOLOGY

In this paper, we are considering one web transaction as one visit to the website which may include ordering a single product (item) or multiple (different) products (otherwise itemsets i.e., group of items) of specified quantities. Usually, a single webpage may be allotted for a single product. A web transaction is said to occur not just by hitting a web page of a company’s product details or spending some time on it; but by ordering or paying for a product and thus making some profit to the company. So our work will be focusing on analyzing which item or itemsets are providing maximum profit value i.e., the most valuable ones. The framework of the intended system consist of an algorithm UP-Growth+ which is an improved version of the state of the art algorithm UP-Growth, used for mining high utility itemsets. The entire algorithm consists of three steps: 1) Scanning of the database twice to construct a global UPTree with the two strategies DGU and DGN. 2) Recursive generation of valuable itemsets from global UPTree and local UP-Trees by UP-Growth with the strategies DNU and DNN. 3) Identifying the most valuable itemsets from the set of valuable itemsets. Thus the entire process can be described as two phases. Both phases consists of two strategies each.

A. Problem Statement

Given a transaction dataset D and a client specified minimum utility threshold min_util, the task of mining the most valuable itemsets from D is to discover the complete set of the items whose utilities are larger than or equal to minimum utility threshold value without any miss, and analyzing the importance (value) of each individual item, so that the most valuable itemsets among them, can be retrieved.

B. Preliminary

An itemset X is a set of k distinct items \{i_1, i_2, ..., i_k\}. An itemset with length k is called a k-itemset. A transaction database D = \{T_1, T_2, ..., T_n\} contains a set of transactions, and each transaction has a unique identifier d, called TID. Each item ip in transaction Td is associated with a quantity q(ip, Td), that is, the purchased quantity of ip in Td. Consider the following small sample set of items retrieved from the web transactional data of an online shopping site. Each item is followed by its count, i.e., number of occurrences in that corresponding transaction. So, one complete transaction includes different products in different web pages of different quantities ordered by a single customer in one single visit to the site.

<table>
<thead>
<tr>
<th>T_id</th>
<th>Transaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>(A,1) (C,1) (D,1)</td>
</tr>
<tr>
<td>T2</td>
<td>(A,2) (E,6) (C,2) (G,5)</td>
</tr>
<tr>
<td>T3</td>
<td>(A,1) (B,2) (C,1) (D,6) (E,1) (F,5)</td>
</tr>
<tr>
<td>T4</td>
<td>(B,4) (C,3) (D,3) (E,1)</td>
</tr>
<tr>
<td>T5</td>
<td>(B,2),(C,2) (E,1) (G,2)</td>
</tr>
</tbody>
</table>
Definition 1: Utility of an item ip in a transaction Td is denoted as \( u(ip, Td) \) and is defined as \( pr(ip) \times q(ip, Td) \).

Definition 2: Utility of an itemset X in Td is denoted as \( u(X, Td) \) and defined as \( \sum_{ip \subseteq X \subseteq Td} u(ip, Td) \).

Definition 3: Utility of an itemset X in D is the sum of the utilities of all occurrences of item X in the entire dataset D and is denoted as \( u(X) \) and defined as \( \sum_{X \subseteq Td \wedge Td \subseteq D} u(X, Td) \).

Definition 4: Transaction Utility of a transaction Td is denoted as \( TU(Td) \) and is defined as the sum total of the utilities of all items in that transaction Td.

Definition 5: Transaction Weighted Utility of an itemset X is the sum of the transaction utilities of all the transactions containing X, in the dataset and is denoted as \( TWU(X) \).

Definition 6: An item or itemset is called a Valuable Itemset if its utility is not less than a user-specified minimum utility threshold or else they are called less-utility or less valuable itemset and is denoted as VI.

Definition 7: An item or itemset is called the Most Valuable Itemset if it is having a utility value greatest of all utility values of the highly Valuable Itemsets and is denoted as MVI.

C. Proposed Framework

The framework of proposed method consists of two phases.

Phase 1: Scan the database twice to construct a global UP-Tree with the first two strategies.

Phase 2: Recursively generate potential highly Valuable Itemsets (abbreviated as VI’s) from global UP local UP-Trees by UP Growth+ with the last two strategies.

The Proposed Data Structure: UP-Tree

To facilitate the mining performance and avoid scanning original database repeatedly, we will use a compact tree structure, named UP-Tree (Utility Pattern Tree), to maintain the information of transactions and high utility itemsets.

Elements of UP-Tree

In an UP-Tree, each node N consists of N. name: the node’s item name, N.count: the node’s support

<table>
<thead>
<tr>
<th>Item</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profit</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
count, \( N.nu \): the node’s node utility, \( N.parent \): the parent node of \( N \), \( N.link \): a node link which points to a node whose item name is the same as \( N.name \). Two strategies are applied to minimize the overestimated utilities stored in the nodes of global UP-Tree. In following subsections, the elements of UP-Tree are first defined. Next, the two strategies are introduced.

**Strategy 1: Discarding Global Unpromising Items – DGU**

TU of each transaction is computed. Then the TWU of each single item is also accumulated. On the basis of TWU global unpromising items are then discarded. Thus the unpromising items are removed from the transaction and their corresponding utility values are also eliminated from the initial TU of the transaction. The remaining promising items in the transaction are sorted in the descending order of TWU and is classified under the name Re-organized Transactions. By reorganizing the transactions, not only less information is needed to be recorded in UP-Tree, but also smaller overestimated utilities for itemsets are regenerated.

**Strategy 2: Decreasing Global Node Utilities – DGN**

Re-organized Transactions are then inserted into UP-Tree in a particular manner. Initially, the first reorganized transaction (e.g. \( T'1 \)) is retrieved. Then the first node representing the first item in that transaction is created with \( Na.item= \{A\} \) and \( Na.count=1 \) and \( Na.nu \) is calculated as RTU \( \{ T'1 \} \) minus the sum of the utilities of the rest items that are behind \( \{C\} \) in \( T'1 \). All the reorganized transactions are then inserted in the same way.

**Phase 2**

**Strategy 3: Discarding local unpromising items and their estimated Node Utilities from the paths** and path utilities of conditional pattern bases – DNU

Here the MNU’s of the nodes are calculated and is recorded in a table named Minimum Node Utility table. Then conditional pattern bases CPB’s of each nodes are generated. By scanning the CPB once the path utility of each local item is calculated. Then DNU is applied. That is the local unpromising items are found out and their MNU’s are discarded from the path utilities of their associated paths and the path utilities are recalculated and the items in each path are reorganized by descending order of path utility of local items.

**D. Experiment and result**

1) **Practical environment.**

In this section, the input dataset and its type, practical results and environment is described.

**Input.**

A transaction dataset and profit table corresponding to the items in the dataset are used for the experiment.

**Hardware Requirements:**

1) Operating System: windows XP/Win7

2) Processor: Pentium IV or advanced

3) RAM: 2 GB

4) HDD: 160 GB

**Software Requirements:**

1) Programming Language: Java

2) Framework: Net beans 6.8 or more

3) Development Kit: JDK 1.6 or more

4) Database: My SQL

**Output**

All Maximum Valuable Itemsets in the input dataset.
2) Data Collection

Any web transaction dataset with the itemset details including their count can be collected from any of the web transaction sites. It should be then tabulated or orderly arranged along with corresponding counts. Datasets can also be collected according to the requirement from the FIMI Repository [13].

3) Result Analysis

![Figure 2: PHUI generation comparison](graphic)

From the figure 2 which is a graph that illustrates the PHUI’s generated (in our method it is called as Valuable Itemsets) by the state of the art algorithm UP growth Vs. the UP growth plus algorithm. From the graph it’s clear that the UP growth plus effectively reduce s the number of valuable itemsets and thus makes it easy to mine the Most Valuable Itemsets from these.

III. CONCLUSION

To present a new scheme for high utility itemset mining from web transactional data, aiming to be with high-performance in terms of performance, scalability and time. This method is very much useful where continuous updating goes on appearing in a database. If the data is continuously added to the original transaction database, then the database size becomes larger and mining the entire process would take high computation time, hence this scheme will mine only the updated portion of the database. It will use previous mining results to avoid unnecessary calculations. High-utility item sets can be generated from UP-Tree efficiently with only two scans of original databases. This can not only decrease the overestimated utilities of PHUIs but also greatly reduce the number of candidates. This scheme overtake other individual algorithms substantially in term s of execution time, especially when databases contain lots of long transactions or low minimum utility thresholds are set, by the use of a two efficient strategies DNU and DNN. Results show that the methods significantly improved performance by reducing both the search space and the variety of candidates.

REFERENCES


