Discovery of Weighted Association Rules Mining

Preetham Kumar
Selection Grade Lecturer, Department of Information and Communication Technology
Manipal Institute of Technology, Manipal, Karnataka
e-mail : preetham.kumar@manipal.edu

Ananthanarayana V S
Professor, Department of Information Technology
National Institute of Technology Karnataka, Surathkal, Karnataka, India
e-mail: ananthvs1967@gmail.com

***Abstract*—Mining of association rules for basket databases, has been investigated by [1] [3] [4], [9], [12], etc. Most of these works focus on mining binary association rules, i.e, most of the association rules mining algorithms to discover frequent itemsets do not consider the quantity in which items have been purchased. This paper discusses an efficient method for discovering a weighted association rules from a large volumes of data in a single scan of the database. The data structure used here is called Weighted Tree. We found that this algorithm is more efficient than Cai's Algorithm.**

Keywords- Association , Attribute Node , Confidence, TID Node, Quantity, Weighted Minimum Support.

I. INTRODUCTION

An *association rule* is an implication of the form $X \Rightarrow l_j$ where X is a set of some items in A, and l_j is a single item in A that is not present in X. The LH.S and R.H.S of the rule $X \Rightarrow l_j$, are called respectively *antecedent* and *consequent* of the rule. It can be seen that the consequent of the rule contains a single item.

The more general rule in which the consequent of the rule can contain any number of items from A is described in [Agrawal et al 1994]. It states that, for a given transaction database D, an association rule is an expression of the form $X \Rightarrow Y$, where X and Y are the sets of items in *A* and $X \cap Y = \phi$.

The rule $X \Rightarrow Y$ holds in the database D with *confidence* c if c% of transactions in D that contain X also contain Y. This is taken to be the conditional probability, P(Y|X). confidence $(X \Rightarrow Y) = P(Y|X) = \dfrac{\text{support}(X \cup Y)}{\text{support}(X)} = c.$

The rule $X \Rightarrow Y$ has *support* s if s% of transactions in D contain $X \cup Y$. This is taken to be the probability, $P(X \cup Y)$. i.e $\text{support}(X \Rightarrow Y) = P(X \cup Y) = s$.

Let D be the transaction database. An itemset $X \subseteq A$ is said to be a *frequent* (*large)* [1] itemset in D with respect to the user specified minimum support *s*, if the support value of X is greater than or equal to *s.* Otherwise it is called an *infrequent* itemset. Any subset of a frequent itemset is frequent. This property of the frequent itemset is called the downward closure property.

The user specified minimum support is the minimum support value defined by the user, for any itemset to be frequent in the database.

If X is a frequent itemset containing k items (k>0), then this set is called *frequent k – itemset*. The association rules that have a confidence greater than or equal to the user minimum confidence are called strong association rules.

The discovery of association rules is a two step process. They are i) Find all frequent itemsets with respect to the user specified minimum support. ii) Use the frequent itemsets to generate the desired rules.

Discovering strong association rules from the frequent itemsets obtained in step (i), is a straight forward process [1] and is described as follows.

For every large itemset l, find all non empty subsets of l. For every such subset a, output a rule of the form $a \Rightarrow (l - a)$ if the ratio of support (*l*) to support (a) is at least minimum confidence. i.e $\dfrac{s\{l\}}{s\{a\}} \geq c$, where *s* and *c* are user defined minimum support and confidence respectively.

The general idea is that if $l = \{A, B, C, D\}$ and $a = \{A, B\}$ are frequent itemsets, then we can determine if the rule $AB \Rightarrow CD$ holds by checking the following inequality,

$$\frac{s\{A, B, C, D\}}{s\{A, B\}} \geq c.$$

Most of the algorithms of association rules assume that items have equal weights, and few algorithms propose a weighted concept [2], [7], [8] [13], [14].

Researchers defined a weighted support, which is calculated by multiplying a support of pattern with a weight. The weight is according to particular items, such as under promotion or more profitable.

Cai et al. introduced two new algorithms in the year 1998 to find weighted frequent itemsets. The items are given weights to reflect their importance to the user. The weights may correspond to special promotions on some products, or the profitability of different items. Therefore, it is mandatory to attach weight field to every item in the database. The first step of these algorithms is to search for the maximum size of the large itemsets. This requires a scan of the database. Further, these algorithms are based on candidate generation and pruning techniques, in addition to the application of k-support bound property. Therefore multiple scans of the database are required to find all weighted frequent itemsets.

978-1-4244-5586-7/10/$26.00 ©2010 IEEE

Lu et al, proposed an algorithm in the year 2001, called Mixed Weighted Association rules which uses the concept of weighted support, and with this algorithm, it is possible to find vertical and horizontal association rules.
Zhang et al in the year 2003, assigned a weight by highlighting the novelty of data, based on the concept of weighted support, and Yun (2007) proposed a weighted confidence on mining interesting patterns.

Liewean Cheng et al 2009, proposed two algorithms based on the concept that the greater the difference among items in an association rule, the higher the weight. The purpose is to discover cross section relationship among items and then extract the unknown patterns.

From literature review on association of data items based on weights, it is evident that researchers do not confirm which algorithm has the best performance. Moreover, none of the algorithms are based on the quantity in which the items have been bought. In a large database it is possible that, even if the itemset appears in very few transactions, it may be purchased in a large quantity for every transaction in which it is present, and may lead to very high profit. Consider for example a sample database given in Table 1.1, in which every element of each transaction represents quantity of the respective attribute or item.

TABLE 1: SAMPLE DATABASE

TID\Attributes	A	B	C	D
1	10	1	0	0
2	0	1	3	0
3	4	0	4	0
4	5	1	5	0
5	0	5	0	10

If user defined minimum support is 2 transactions, then an item D is not frequent, and will not appear in the set of frequent itemsets, even though it is bought in large quantity, and leads to more profit than other frequent items. Therefore the quantity in which the items are bought is the most important component, without which loss of information may occur.

II. RELATED AREA AND MOTIVATION

There are many interesting algorithms for finding frequent itemsets based on user defined minimum support, and a few algorithms are based on weighted concept. Some of the weight based algorithms are [2], [8],[10], [12], [13]. In most of the papers the weighted support is calculated by multiplying a support of pattern with a given weight. The weight is according to particular items which are under promotion or more profitable. The definition of weight defined in the discovery of frequent itemsets using weighted Tree approach is different, and is based on the quantity in which the items have been bought. In this case weight of a k-itemset I is defined as follows.

$$Weight(I) = \sum_{j}\sum_{i} qij \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1)$$

where i = 1, 2,….k and j =1, 2, .. n and q_{ij} represents a quantity of an item i ∈ I , in a j^{th} transaction. This definition is modified in the paper[11] . i.e

$$Weight(I) = \frac{\sum_{j}\sum_{i} qij}{n}$$, where i = 1, 2,….k and j =1, 2, .. n and q_{ij} represents a quantity of an item i ∈ I , in a j^{th} transaction……..……..……..(2)

Both of the above algorithms work based on weighted minimum support. If an itemset satisfies user defined weighted minimum support, then it is considered as a frequent itemset.

There is one possible problem in these above mentioned methods if we take the definition of weight as defined in equations (1) and (2). Even if some items appear in a small quantity, when the number of items in an itemset is large, the total quantity may be large, and the weight of that set is greater than the weighted minimum support. Depending on the application requirements, this may or may not be desirable. It may be desirable if the user considers a total quantity, which leads to profitability, as interesting. It may not be desirable, if an itemset with many light weighted items should not be considered interesting. Further, in the weighted case, the downward closure property does not hold good. It is not necessary that all subset of a frequent itemset are frequent, necessary that all the subsets of a frequent itemset are frequent, since the meaning of frequent itemset is modified to handle weighted support. Hence downward closure property need not be true in a weighted case.
For example, consider the sample database given in Table 1. If we assume w_min_sup =2 and applying weight definition given in equation (2), then the items A , B, C and D are frequent 1-itemsets. Since,
Weight(A) = 19/3= 6.33,
Weight(B) = 8/4 =2,
Weight(C)=12/3=4,
Weight(D) =10/1 =10
Further, we see that 2-itemset{A, B} is also frequent as Weight ({A, B})=17/2= 8.5. We observe from this that, even though item B appears in smaller quantity, an itemset {A,B} is frequent. This issue has motivated us to propose the following method to discover frequent itemsets based on weight in a single scan of the database. Further, we found that this algorithm is efficient than Cai's algorithm for finding weighted association rules.

Problem Statement

Given a transaction database D, it is required to discover weighted association rules using tree based data structures named Weighted Tree in a single scan of the database.
General Algorithm to achieve this
Input: Database D
Output: Weighted Association Rules
Steps:

1. Construct Weighted Tree with every elements of the transaction in D.
2. Discover all frequent itemsets based on weighted minimum support.
3. Apply Association Rules Mining Algorithm to discover weighted association rules based on user defined minimum confidence. This step is a straight forward step and is as given in [Han et al 2004].

The most important step in the above problem is discoveryof frequent itemsets based on user defined weighted minimum support

III. PROPOSED METHOD

Our definition of weight is based on the quantity in which the items have been bought and is defined as follows. The weight of an itemset I is the ratio of the sum of the quantities of all the items in I for every transaction which contains I, to the number of transactions in which I is present.

i.e., $Weight(I) = \frac{\sum_j \sum_i qij}{n}$ where i = 1, 2,….k and j =1, 2, .. n and q_{ij} represent the quantity of an item i ∈ I , in a j^{th} transaction……..…….(3)

We construct an abstraction of the database in a memory, and then we use it to find the frequent itemsets based on weight.

The association rules involving items which are frequent, based on weight in *antecedent* and *consequent* part of a rule are called weighted association rules. These rules are said to be strong if it satisfies user defined minimum confidence. i.e if *l* = {A, B, C, D} and a = {A, B} are frequent itemsets, then we can determine if the rule AB⇒CD holds by checking the inequality (4),

$$\frac{Weight\{A,B,C,D\}}{Weight\{A,B\}} \geq c \quad \text{……..……(4)}$$

A k-itemset I is said to be frequent, if it satisfies the following conditions.

(i) Weight of individual items in I with respect to the set of transactions which contain I, is greater than, or equal to, w_min_sup and

(ii) Total weight of I is greater than, or equal to, w_min_sup . If one of the above conditions fails then I is not a frequent itemset.

Weighted Tree Algorithm

Structure of a node in Weighted Tree : The Weighted tree has two different nodes, and is shown in Figure 1.1.

(i) The first type of node shown in Figure 1(a), labeled attribute contains an attribute or item name, and two pointers, one pointing to the nodes containing transaction ids and weights, and the other is a child pointer pointing to the next attribute. This node represents the head of that particular branch.

(ii) The second type of node shown in Figure1 (b) has 2 parts. The first part labeled TID, represents a transaction number or id and the second part of which is labeled weight, indicates quantity purchased in that transaction. This node has only one pointer pointing to the next object having this particular attribute.

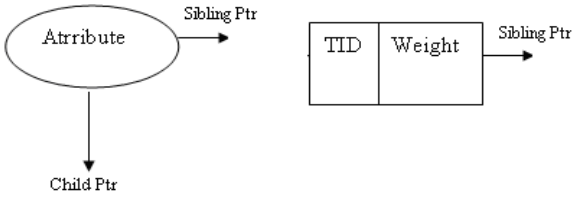

Figure 1: (a) Attribute Node (b) TID Node in a Weighted Tree

Algorithm for Discovery of Weighted Association Rules Based on Weights

The Weighted tree algorithm involves 4 steps. They are

(i) Construction of Weighted Tree
(ii) Removal of infrequent attributes of Weighted Tree
(iii) Discovery of frequent itemsets based on weights
(iv) Discovery of weighted Association Rules

Input: *The database D*
Output: *Weighted Tree*

(i) Construction of Weighted Tree

Method:

for each item or attribute with quantity q in a transaction t ∈ D do
begin
create a node labeled with q and add these nodes to the respective attribute node.
end

(ii) Algorithm for Reducing the Weighted Tree

Input *: w_min_sup = user specified weighted minimum support*
Output: *Weighted Tree without infrequent attributes.*
Method:

for each attribute' a' in Weighted tree do
begin
if sum(quantities of all of nodes of a)/n < w_min_sup then
remove a's branch from the tree
end

(iii) Algorithm to discover all frequent itemsets based on weight

F, is the set of all frequent 1-itemsets or attributes, P, is the set of all non empty subsets of F excluding the sets containing one attribute, and f is set of attributes and is an element of P.

F_w is the set of all frequent attributes or one itemset

Input: *A reduced Weighted tree,*
w_min_sup = user specified weighted minimum support

Output: *Set of all frequent itemsets, F_w .*
Method:

```
for each f in P do
begin
      T={TIDs of first attribute in f }
            for each m in f other than first attribute do
     begin
                  T=T ∩ { TIDs in which  m is present}
    end
     if  T is non empty then
          for each attribute 'a'  in f
           begin
            if(sum (quantities of 'a' in every transaction t
                  in  T )) /|T| ≥ w_min_sup
                then   flag=1;
           end
        if ( flag==1) then
             if ( sum (quantities of elements of T w.r.t to
                    f )) /|T|  ≥ w_min_sup then
                  F_w = F_w ⋃ f
end
```

(iv) Discovery of Weighted Association Rules

Input: *F_w, a set of frequent itemsets obtained based on weight.*
c, a user defined minimum confidence.
Output: *Set of all Weighted Association rules.*
Method:

```
for every items set f in F_w
begin
 for every subset s in f
      begin
          if( weight(f)/weight(s)) ≥ c
           output a rule of the form s ⇒ (f-s)
       end
end
```

Illustration

Consider for example, a sample database given in Table 1. A Weighted Tree for this database is shown in Figure 2.

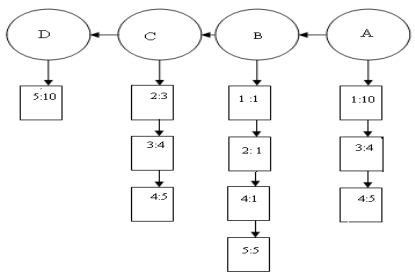

Figure 2: Weighted Tree for Table 1

If w_min_sup = 2 then applying above algorithm,
we get Fw ={A, B,C, D}
Now we apply step3 of the algorithm for the above set.
P={{A,B}, {A, C}, {A,D}, {B,C}, {B,D}, {C,D}, {A,B,C}, {A,C,D},{B,C,D},{A,B,D}, {A, B,C, D} }

Consider a set {A, B}, which appears in transaction 4. i.e T={4}. Also, sum of quantities/|T|=(6)/1=6. Even though {A,B} satisfies w_min_sup, weight of B corresponding to the number of transactions in T is less than w_min_sup. i.e weight(B) in T = 1/1 =1 < w_min_sup. Hence {A,B} is not frequent. By similar arguments we can prove that {B,C} is also not frequent.
Consider a set {A, C} , which appears in transactions 3 and 4. i.e T={3,4} . Also, sum of quantities/|T|=(9+9)/2=9. Hence it is frequent. Similarly, {B,D} is found to be frequent.
By similar arguments, we found that the sets {A, D}, {C,D}, {A,C,D},{B,C,D},{A,B,D}, and {A, B,C, D} are infrequent sets. Hence Fw = { {A}, {B},{C}, {D},{ A, C}, {B,D}}.

IV. EXPERIMENTAL ANALYSIS

The weight definition of our algorithm is different from all the existing algorithms for mining weighted association rules. For experimental purposes we compared our algorithm with Cai's algorithm [Cai et al 1998] for discovering weighted association rules.

In Cai's algorithm every item has to appear with a weight. In order to achieve this, using our definition of weight, the weights for all the items of the datasets are calculated. These weights are then appended to the respective items of datasets for running Cai's algorithm

For the experimental analysis, simulation of buying patterns of the customers in retail patterns is generated, and in the data set which we have used, every element of the transaction is considered as quantity of the corresponding attribute in the database. The data sets used here contain transactions 1K, 2K, 3K, 4K and 5K with 20 items. The weighted minimum support used is equal to 2.

The time and space required in both the cases is given Table 2 and 3 respectively. The corresponding graphs are given in Figures 3 and 4.

TABLE 1: TIME REQUIRED FOR CAI'S AND WEIGHTED TREE

Data Sets Used in K	Weighted Tree in Seconds	Cai's Algorithm in Seconds
1K	43	55
2K	76	83
3K	92	97
4K	113	123
5K	141	153

TABLE 2: SPACE UTILIZED FOR CAI'S AND WEIGHTED TREE

Data Sets Used in K	Weighted Tree in Bytes(X 10^5)	Cai's Algorithm in Bytes(X 10^5)
1K	0.805	0.753
2K	2.352	1.9

3K	3.458	2.56
4K	4.211	3.72
5K	5.154	4.28

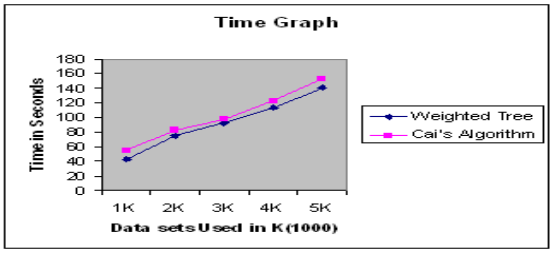

Figure 3: Time Graph of Cai's and Weighted Tree

Even though the Weighted Tree algorithm requires around 17% more space than Cai' algorithm, it is time efficient. It requires around 10% less time than Cai's algorithm. This may be due to Computation of the k-support bound for every set of items to check whether it is a subset of k-frequent itemset, Requirement of multiple scans of the database, Computation of candidates itemset.

Therefore the proposed algorithm excels over Cai's algorithm in the following ways:

Scans database only once., Scanning of database is not required to find the maximum size of the frequent itemsets,It is a non candidate generation method, Weights are calculated randomly.

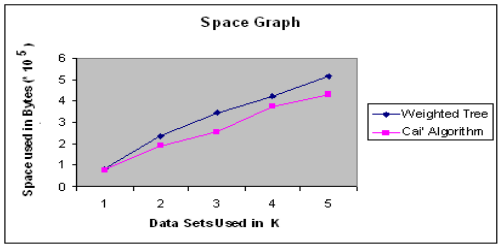

Figure 4: Space Graph of Cai's and Weighted Tree

V. THEORETICAL ANALYSIS

Time complexity

In a transaction database D, if there are |D| transactions, then |D| transactions have to be read by the algorithm to construct a Weighted Tree. Therefore, this tree can be constructed in O (|D|) steps. If there are m attributes, then reduction of the Weighted Tree is in O (m). The major step in the discovery of frequent itemsets based on user defined weighted minimum support is the process of finding power set P. If there are n frequent 1-item attribute sets, then the number of possible subsets obtained from this is 2^n. Hence this step is in $O(2^n)$.

Space complexity

If the average length of the transaction t is equal to m attributes, then there will be m*(|D+1) nodes in the tree. Hence this step is in O(m*|D|). If there are k infrequent items then the Reduced Weighted Tree will contain (m-k) attribute lists with at most(|D|+1)* (m-k) nodes. This step is in O(|D|*(m-k)).

VI. CONCLUSION

The novel method for discovering frequent itemsets based on weights is discussed. This method is also found to be efficient than Cai's algorithm. The process of finding all subsets requires exponential time. Therefore, a suitable method has to be thought of in this direction and is considered as future enhancement.

REFERENCES

[1] Agrawal R and Srikant R (1994) "Fast algorithms for Mining association rules". *In Proceedings of the 20th VLDB Conference*, pages 487-499, 1994.

[2] Cai, C.H., Fu, Ada W.C., Cheng, C.H. and Kwong, W.W.(1998), " Mining Association Rules with Weighted Items", Database Engineering and Applications Symposium, 1998, In Proceedings of IDEAS '98.

[3] Cheung D, V.T. Ng, A. Fu, and Y. Fu(1996). "Efficient mining of association rules in distributed databases" *In IEEE Transactions on Knowledgeand Data Engineering,* pages 1-23.

[4] Han. J, M. Kamber, and J. Chiang(1997) "Mining multidimensional association rules using data cubes"Technical report, Database Systems Research Laboratory, School of Science, Simon Fraser University.

[5] Han, J., Kamber, M.(2001) "Data Mining: Concepts and Techniques", Mogan Kaufmann Publishers.

[6] Han. J and M. Kamber(2004), "Data Mining Concepts and Techniques": San Franscisco, CA:. *Morgan Kaufmann Publishers*.

[7] Liewean Cheng, Su-Chuan Chen, and Jashen Chen (2009) "Applying Weighted Association Rules with the Consideration of Product Item Relevancy", 978-1-4244-3662-0/09, 2009 IEEE

[8] Lu, S., Hu, H., and Li, F.(2001), "Mining weighted association rules", *IntelligentData Analysis 5 (2001),* pp.211-225.

[9] Park J.S , Chen M-S, and Yu P.S. (1995) An effective hash-based algorithm for mining association rules.*In Proceedings of ACM SIGMOD,* pages 175-186.

[10] Preetham Kumar and Ananthanarayana V.S(2008) " Discovery of frequent itemsets using Weighted Tree method" IJCSNS, Vol. 8 No. 8 pp. 195-200

[11] Preetham Kumar and Ananthanarayana V.S(2009) "Discovery of Frequent Itemsets Based on Minimum Quantity and Support" Computer Science Journal of Malaysia, Vol 3, Issue 3, June 2009.

[12] Savasere, A., Omiecinsky, E., Navathe, S (1995): "An Efficient Algorithm for Mining Association Rules in Large Databases", *Proc. of International conference. on VLDB*, Zurich, Switzerland, 432-444.

[13] Yun, U(2007)., "Efficient mining of weighted interesting patterns with a strong weight and/or support affinity", *Information Science* 177 (2007) pp.3477-3499.

[14] Zhang, S., Zhang, C., Yan X., "Post-mining: maintenance of association rules by weighting", *Information System* 28 (2003) pp.691-707.