Association Rules Mining Including Weak-Support Modes Using Novel Measures

JIAN HU, XIANG –YANG LI
School of Management
Harbin Institute of Technology
Harbin, Heilongjiang Province, 150001
P. R. China
jianhu-hit@163.com

Abstract: -In association rules mining application, some rules can provide a lot of useful knowledge for us, though these rules have the lower Support, called weak-support mode in this paper. However, in existing Support-Confidence framework, the rules with lower Support will be lost. Thus, this paper puts forward a new association rules mining technique, which sets up the lower support threshold value to ensure the weak-support rules to be mined and applies Csupport measure to recognise weak-support mode. Then, a new measure, called as N-confidence, is used to restrict mining size in generation frequent sets, which can strain away the weak-support rules without correlation. Furthermore, this paper puts forward a new interesting measure to distinguish from the association rules interesting degree. In order to enhance mining efficiency, a novel algorithm, namely FT-Miner, is presented to discover association rules in a forest by using two new data structures, including UFP-Tree and FP-Forest. The experimentation shows that the algorithm not only mines useful and weak-support rules, but also has better capability than classical association rules mining algorithms.

Key-Words: - Data mining; Association rules; Weak-support mode; UFP-Tree; FP-Forest

1 Introduction

Association rules mining is an important problem in the data mining area and has a wide range of applications. It was introduced by Agrawal et al. [1] in the context of transaction databases. A transaction database is a database containing a set of transactions and each transaction is associated with a transaction id. Let $D=\{t_1,t_2,\cdots,t_n\}$ be a transaction database and $I=\{i_1,i_2,\cdots,i_n\}$ be the set of items in *D*, where $t_i(i\in[1,n])$ is a transaction and $t_i\subseteq I$. An association rule can be depicted as $X\Rightarrow Y$, where $X\subset I$, $Y\subset I$, $X\cap Y=\varnothing$ .The Support of the rule $X\Rightarrow Y$ in *D* is the ratio of the number of items containing *X* and *Y* in the transaction sets to the number of items in the transaction, which formula is $\text{Support}(X\Rightarrow Y)=\left|\{t:X\cup Y\subseteq t\in D\}\right|/\left|D\right|$. And the Confidence of the rule $X\Rightarrow Y$ in the transaction sets is the ratio of the number of transactions including *X* and *Y* to the number of those including *X*. It is written as Confidence ($X\Rightarrow Y$), that is to say, $\text{Confidence}(X\Rightarrow Y)=\left|\{t:X\cup Y\subseteq t\in D\}\right|/\left|t:X\subseteq t,t\in D\right|$. Association rules mining process can be divided into two steps:

- Finding all item sets having the highest Supports;
- Generating association rules from frequent item sets.

In existing association rules mining, most algorithms depend on Support-Confidence structure. However, there are some shortcomings to be solved. In application, Support is usually set up the bigger value to avoid the combination explosion problem. So many potential and valuable rules are cut down in frequent item sets pruning process. But, these weak Support rules sometimes present us some interesting knowledge, which may be more novel than the rules with the higher support. Such as, the sale of superior goods in shop is specially paid attention by managers, even though their Support is lower. In addition, the formula Confidence is $Conf(X\Rightarrow Y)=\frac{Sup(X\Rightarrow Y)}{Sup(X)}=\frac{P(XY)}{P(X)}$, which only thinks relation *X* and *Y* and ignore the situation of appearance of *Y*. When $\frac{P(XY)}{P(X)}$=1, it denotes that *X* and *Y* is independent. But the confidence of rule is enough big, which can make the rule establishment. In the condition, the rules are inveracious, which can not be used by user.

In order to solve above-mentioned questions, the best method is setting up a lower Support threshold

value to ensure to get all interesting rules and apply other measures to limit the rules generating. On the basis, this paper gives a new association rules mining structure, which defines Csupport measure to ensure rules containing weak Support mode and puts forward N-confidence to restrict the number of rules. At the same time, a new interesting measure is presented to get the rules which are required by users. In addition, a new algorithm based on frequent growth method and dividing and conquering strategy, FT-Miner, is designed to mine association rules on the basis of the new two data structure UFP-Tree and FP-Forest to enhance mining efficiency.

2 Related Work

In existing association rules algorithms, Apriori[1] and FP-Growth[2] are the most influential association rule mining algorithms.

Apriori algorithm adopts layer by layer search iteration method to mine association rules, which is frequent k-1 itemset. L_{k-1} is used to search K itemset L_k. However, it needs to scan database when L_k is found. So, Apriori algorithm will generate a large number of candidate sets. If there are 10^4 frequent 1-itemsets, then more than 10^7 candidate 2-itemsets are produced. Furthermore, Apriori algorithm needs frequently scan database. 2^m-2 possible subsets will be tested for m-frequent itemsets. In order to enhance mining efficiency, some researchers continuously put forward improved Apriori algorithm, such as AprioriTid[3], DIC[4], Partition[5], DHP[6]. However, most improved algorithms are based on Apriori structure, and many candidate sets are still brought.

F-Growth is proposed by J. W. Han et al, which compresses data items into FP-tree, then calls FP-growth method to divide FP-tree into some conditional pattern base. Then frequent itemsets are mined in conditional pattern base by depth-first method. F-Growth algorithm does not generate candidate itemsets, and only scans two times database. So its efficiency is higher 10 times than Apriori algorithm, but it needs excessive CPU consumption and storage cost. Some improved algorithms of F-Growth includes FP-Growth*[7], AFOPT [8], COFI [9], and so on.

To solve the existing problem of Support-Confidence structure, some scholars presented improved algorithms. Liu et al[10] put forward MSApriori algorithm to discover the association rules for significant rare data by improved Apriori algorithm that assumed the uniform frequency pattern of the data. Xiong et al [11] made a research on the mining of strong correlation item pairs by using Pearson's correlation coefficients as measuring criteria and expressed Taper algorithm. Whereas, when the items and trade numbers are great, the algorithm still needs a mass of calculation. Some researchers put forward improving algorithms on the basis of Xiong. In order to avoid costly testing of a large number of candidate pairs, Zeng you H.[12] proposed an efficient algorithm, called Tcp, based on the well-known FP-tree data structure. Tie yun. Q [13] mined negative correlation item pairs, and a new algorithm called MNI was introduced to use the lower bound of Phi correlation coefficient to generate all candidate negative correlation items. Omiecinski[14] illustrated two new correlation measuring, including all-confidence and bond. Both have the downward closure property.

In addition, other scholars also described their own viewpoints recently. Lee et al [15] brought forward two new association algorithms including all-confidence and bond by means of pattern-growth technology, which were considered as Omiecinski's concept extension. Kim et al [16] put forward the concept of confidence-closed correlated patterns and illustrated the mining algorithm named as CCMine.

3 Weak-support Mode and Novel Measures

3.1 Weak-support Mode

In the real application, there are relatively infrequent data in data set. However, the existing association rule mining algorithms get some rules based on Support, which are inadequate to measure the importance of a rule composed by the data items only according to their Support. Some data occurs infrequently, but they appear simultaneously with the specific data in high proportion. So, the data is significant for some users despite the low Support, thus the special characteristic is worthy to be discovered. The existing association rule algorithms can not mine the rules in which the Support is low and can not satisfy the Support.

Example 1. Suppose data items a, b and c belonging to the data set A, where each item of a, b and c has Support of 10%, 30% and 15% respectively and the user sets the minimum support threshold value to 20%. So, a and c are not the frequent item since they do not satisfy the minimum Support. However, the itemset {a, b, c} may have Support of 9%, and it has 90% possibility that a may appear together with b and c. In the existing mining algorithms, the itemset{a, b, c} will be lost because it does not satisfy the minimum Support.

In this paper, we suggest weak-support mode,

which can be defined as:

Definition 1 Weak-support mode is one itemset which frequency in the database is too low. But it appears with the specific data in high proportion of its frequency.

In order to discover weak-support mode rules in searching frequent itemset, we present a strategy which adopts a lower Support threshold to ensure weak-support mode not to be lost. At the same time, a new measure, Csupport, is put forward to prevent generating unnecessary rules.

Given an item set $X=\{i_1,i_2,\cdots,i_n\}$, the Csupport can be defined as:

$$Csupport(i_1,i_2,\cdots,i_k)=\frac{\min\{Sup(i_1),Sup(i_2),\cdots Sup(i_n)\}}{\max\{Sup(i_1),Sup(i_2),\cdots Sup(i_n)\}} \quad (1)$$

Theorem 1 If the support of i_k ($i_k \in X$) is lower, and $Csupport(i_1,i_2,\cdots,i_k)$ is greater than the threshold value h_c which is set up by user, the itemset X is a weak support mode.

3.2 N-confidence

By Csupport criteria, some weak-support modes are included in frequent itemsets, but in weak-support mode, many itemsets have not correlation and unnecessary for user. So, this paper proposes a new measure, N-confidence, to restrict the useless rules generating.

Lemma 1. Confidence has the anti-monotonicity.

$$Conf\{i_1,i_2\}\Rightarrow\{i_3,i_4,\cdots,i_k\}\le Conf\{i_1,i_2,i_3\}\Rightarrow\{i_4,i_5,\cdots,i_k\} \quad (2)$$

Lemma 2. For a frequent item set $X=\{i_1,i_2,\cdots,i_k\}$, $Sup(i_j)=\max\{Sup(i_1),Sup(i_2),Sup(i_3),\cdots,Sup(i_k)\}$, then the rule $i_j\Rightarrow i_1,i,i_3,\cdots,i_k$ has the minimum confidence in all rules, which is

$$\frac{Sup(\{i_1,i_2,\cdots,i_k\})}{Max\{Sup(i_1),Sup(i_2),\cdots,Sup(i_k)\}} \quad (3)$$

called as N-confidence.

From above lemmas, it is easy to know that $\frac{Sup(\{i_1,i_2,\cdots,i_k\})}{Max\{Sup(i_1),Sup(i_2),\cdots,Sup(i_k)\}}\le\frac{Min\{Sup(i_1),Sup(i_2),\cdots,Sup(i_k)\}}{Max\{Sup(i_1),Sup(i_2),\cdots,Sup(i_k)\}}$.

So, $N\text{-}confidence\le\frac{Min\{Sup(i_1),Sup(i_2),\cdots,Sup(i_k)\}}{Max\{Sup(i_1),Sup(i_2),\cdots,Sup(i_k)\}}$.

This paper applies *N-confidence* as a measure to delete the rules without correlation in weak support mode. If *N-confidence* is greater than the given threshold h_n, then the rule is valid.

Example 2. In a shopping basket, the Support of some very superior brandy is 0.15%, the Support of some high-class cigarette is 0.2% and the support of X= {very superior brandy, high-class cigarette} is 0.1%. In existing association rules mining algorithms, X will be lost because of their low Support. *Csupport* is equal to 0.75. Suppose h_c=0.5 and h_n=0.3, then {very superior brandy, high-class cigarette} is weak support mode. By the formula (3), *N-confidence*(X) is equal to 0.5. So, {very superior brandy, high-class cigarette} is useful, which will be held to generate rules.

3.3 A New Interesting Measure

The Interesting of rules is an integrative measure with reasonable, novel and comprehensible characters. At present, some scholars presented different interesting. Han[17] put forward Interesting with correlation, which value range is [0, +∞]. It denotes that the rule is positive correlation when the value interesting is greater than 1. For the upper bound +∞ of interesting, it is difficult to reflect the real difference of Interesting, such as Interesting ($X\Rightarrow Y$)=3 and Interesting ($X\Rightarrow Y$)=5; G. Piatetsky[18] gave other Interesting with correlation, but it has the above question; P. Smyth et al [19]presented J-measure Interesting, which simplified the rules and thought the similarity of rule antecedent X and consequent Y probability distribution, but ignored the infection $P(Y)$; Klemettinen[20] proposed rule template Interesting which needed user to indicate the type of rules, then the rules meeting the requirement can be got.

According to above analysis, existing Interesting has some shortcomings. So this paper puts forward improved Interesting, which is defined as followed.

$$Interesting(X\Rightarrow Y)=\begin{cases}\frac{Con(X\Rightarrow Y)-Sup(Y)}{Con(X\Rightarrow Y)} \\ Con(X\Rightarrow Y)\ge Sup(Y)\end{cases}=\begin{cases}\frac{P(XY)-P(X)P(Y)}{P(XY)} \\ P(XY)\ge P(X)P(Y)\end{cases} \quad (4)$$

In application, we usually concern the positive rules, so the value range of new Interesting is limited to positive real number. According to (4), which value belongs to [0, 1).

Theorem 2 If Interesting($X\Rightarrow Y$)=0, that is $P(XY)$=P(X)P(Y), it shows that there is no interaction of the appearance of item X and item Y, which are independent.

Theorem 3 The value of Interesting ($X\Rightarrow Y$) is the more closed to 1, it suggests that the appearance of X item has more important action than the appearance of Y item. Contrarily, X item has smaller effect than Y item.

4 Algorithm Describing

To generate candidate itemsets, we should be able to construct the candidate itemset that contains weak-support mode. However, due to adopt the lower Support threshold value, classical Apriori and FP-Growth algorithms can not be used to search candidate itemsets. So, this paper designs two new data structures, called UFP-Tree and FP-Forest, to

avoid too bigger data structure. Then, a new algorithm, called FT-Miner, is described in this paper. A transaction database is stored by many UFP-Trees, those UFP-Trees form FP-Forest. The FT-Miner algorithm traverses FP-Forest in top-down depth-first order.

4.1 Data Structure

In this chapter, we take for an example of the data in Table 1. UFP-Tree is a mutation of FP-Tree. Suppose the Support threshold value is 0.2. Fig.1 shows UFP-Tree construction process.

Table 1 Transaction database

Tid	Item set	Tid	Item set
100	{I_1, I_2, I_5}	600	{ I_2, I_3}
200	{I_2, I_4}	700	{ I_1, I_3}
300	{ I_2, I_3}	800	{I_1, I_2, I_3, I_5}
400	{I_1, I_2, I_4}	900	{I_1, I_2, I_3}
500	{ I_1, I_3}		

At first, frequency 1-itemsets, named L = {I_5, I_4, I_1, I_3, I_2} are got by one-time scanning database, which are arranged according to Support ascending order, as can be seen in Table 2.

Table2 Frequent 1-itemset

Frequent 1-itemset
I_5: 2
I_4: 2
I_1: 6
I_3: 6
I_2: 7

It is different with other frequency item sets mining algorithm, that UFP-Tree doesn't use item head table. The root of the UFP-Tree is an item in L which can identify this tree, not null, and seeks data with top-down method. So the length of L is the number of UFP-Tree. Then, database is scanned second time, frequent item set tables are obtained for every transaction, which are defined as F. For transaction $T100$, frequent item set table which is arranged according to support ascending order, F_1= {I_5, I_1, I_2}, and the first item set I_5 is treated as root of UFP-Tree, defined this tree as T_{I5}, the other item sets I_1, I_2 are insert T_{I5}. Also, F_2= {I_4, I_2} for transaction $T200$, the first item set I_4 is the root T_{I4}, the item set I_2 is inserted in T_{I4}. In the same way, multi-trees structure is built when every transaction is operated, which is shown in Fig.1.

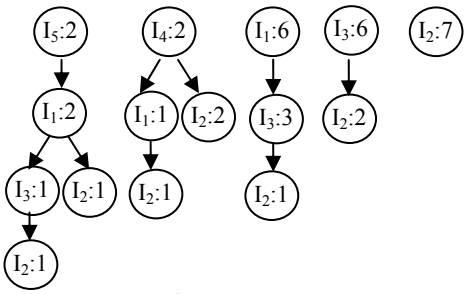

Fig.1 UFP-Trees

FP-Forest is constructed by UFP-Trees for better managing data. Fig.2 shows FP-Forest construction. At the same time, one-dimension array A_{Ti} is applied to store the counts of other nodes of UFP-Tree.

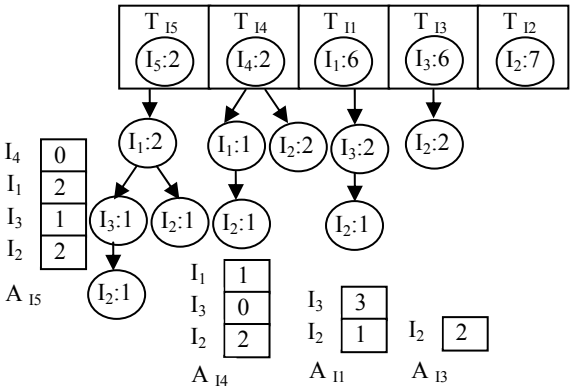

Fig. 2 FP-Forest

The pseudocode of UFP-trees and FP-forest building algorithm are given in Fig.3.

FP-forest algorithm

Input: transaction database D; Min-Support;

Output: FP-forest

Scan D, get frequent 1-itemsets L;

Build n Trees T_{Ii} using the items in L as root Node and n one-dimension A_i;

Scan D for second time

For every transaction $t \in D$

{Get frequent item table F according to the order of L;

Find the tree $T_{F,}$ which root is the first item of F;

Insert other node of F into T_F, A_i stores the count of node in tree T_F}

Fig. 3 FP-forest algorithm

4.2 Algorithm Process

FT-miner adopts frequent growth based on depth-first searching method, dividing and conquering strategy to respectively treat every UFP-Tree. According to study the structure of UFP-Tree, a type tree, named single frequent branch, is found that it can get frequent item sets by easy enumeration.

Definition 2 When traversing an UFP-tree in top-down, if the count of a node of UFP-tree is less than the minimum threshold value of support and the count of father node is greater than or equal to the minimum threshold value of support, and every ancestor node only have one child node, then the UFP-tree is defined as single frequent branch.

Due to use multi-trees to store data, it is possible that the items in a UFP-tree do not exist in other UFP-trees. So right shift combination operation is designed to avoid some items loss in frequent item sets. When a UFP-Tree has been mined, then right shift combination operation is used to aggregate branches of tree into other UFP-Trees.

Definition 3 Right shift combination operation. When an UFP-tree has been mined, the branch of UFP-tree is searched. The first node is found of branch. Then the branch is inserted some UFP-tree which root node is same with the first node of branch.

The association rules mining procedure to apply the new measures is described in Fig.4.

FT-Miner algorithm

Input: F-Forest structure; Csupp; N-conf; Inter

Output: Association rules

```
For each UFP-Tree T_Ii in FP-Forest
  {
   Scan UFP-Tree T_Ii;
   If T_Ii is single frequent branch
     Get combination of root node and T_Ii' nodes
     Calculate the Csupport and N-confidence
     If Csupport> Csupp && N-confidence> N-conf
       Calculate the Interesting;
       If Interesting> Inter
         Output association rules
   Else find frequent 1-itemsets L;
       If the length L=1
         Get combination of frequent 1-itemsets and root node as frequent itemsets;
         Get subsets of frequent itemsets;
         Calculate Csupport and N-confidence
         If Csupport> Csupp && N-confidence> N-conf
           Calculate the Interesting;
           If Interesting> Inter
             Output association rules;
       Else F-Forest (L)
             FT-Miner (new F-Forest);
             T_Ii right shift combination operation;
   }
```

Fig. 4 FT-Miner algorithm

Table3 Parameter and function in FT-Miner algorithm

Csupp	The threshold value of Csupport
N-conf	The threshold value of N-confidence
Inter	The threshold value of Interesting
F-Forest	A function to build F-Forest data structure

FT-miner adopts frequent growth based on depth-first searching method, dividing and conquering strategy to respectively treat every UFP-Tree. This paper takes the case of T_{I5} to display association rules mining algorithm. Therein, Csupp is equal to 0.8; N-conf is equal to 0.25; Inter is equal to 0.5. The mining process is as following.

(1) Scan UFP-Tree T_{I5} and find frequent items $\{I_1, I_2\}$, then arrange them according to support ascending order to form frequent 1-itemset $L=\{I_1:2, I_2:2\}$ of a new F-Forest, which is shown as Fig.5.

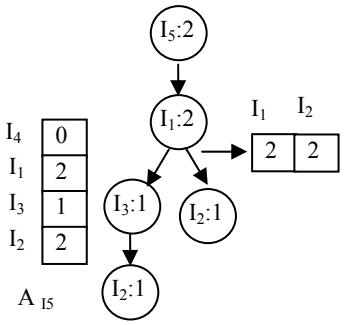

Fig.5 Find frequent 1-itemsets

(2) Build a new F-Forest with two UFP-trees as Fig.6, namely T_{I5I1} and T_{I5I2}. Search two frequent branches $\{I_1, I_2:1\}$ and $\{I_1, I_2:1\}$, and insert them into T_{I5I1}.

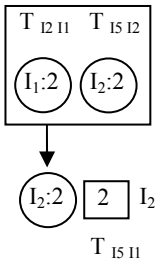

Fig. 6 Building new FP-Forest on T_{I5}

(3) Then, perform FT-Miner operator for new tree F-Forest and get association rules in Table 4. From Table 4, we can know that the supports of association rules are low, but Csupport of these rules is satisfied with Csupp. So, these rules are weak-support mode, which can be discovered by FT-Miner algorithm.

(4)Perform right shift combination operation for T_{I5}. The root of UFP-tree only has one children $\{I_1:2\}$ and insert it into UFP-tree T_{I1}, as can be seen in Fig.7.

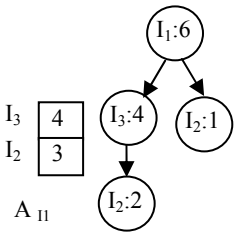

Fig. 7 T_{i5} right shift

Table 4 Association rules

Frequent Itemset	Possible Association Rules	Csupport	N-confidence	Interesting	Association Rules
$\{I_5, I_1: 2\}$	$I_5 \Rightarrow I_1$	1.0	0.33	0.78	$I_5 \Rightarrow I_1$
	$I_5 \Rightarrow I_1 \wedge I_2$			0.56	
$\{I_5, I_1, I_2: 2\}$	$I_1 \wedge I_5 \Rightarrow I_2$	1.0	0.29	0.22	$I_5 \Rightarrow I_1 \wedge I_2$
	$I_5 \wedge I_2 \Rightarrow I_1$			0.22	
$\{I_5, I_2: 2\}$	$I_5 \Rightarrow I_2$	1.0	0.29	0.22	No

4.3 Time Complexity Analysis

The worst situation of time complexity for FT-Miner algorithm is that all the UFP-Trees have not single frequent branch. In search a UFP-Tree T, it firstly needs scan array A and find frequent 1-itemsets L. So the time cost is $O(A)+O(LlogL)$. Then building the new FP-Forest, if there are n leaf nodes and F_i is frequent item table including i leaf node, the time cost is $\sum_{i=1}^{n} O(F_i)$. In addition, for right shift combination operation, supposing T has m sub-trees, and the time cost is $\sum_{j=1}^{m} Cost(j,T^j)$. Therein, $Cost(j,T^j)$ is time cost of sub-tree j combination into UFP-Tree T^j which rood node is the same with the sub-tree j' rood node. So time cost is $\sum_{k=1}^{l}(O(A+L\cdot\log L+\sum_{i=1}^{n}O(F_i)+\sum_{j=1}^{m}Cost(j,T^j)+\sum_{j=1}^{m}Cost(j,T^j))$ for FT-Miner algorithm.

5 Experimental Study

In order to test FT-miner algorithm capability, experiment adopts the real data. Mushroom, Connect and Pumsb* data sets come from UCI[21]. T30I1.0D60k data set is generated by IBM synthetic data generator. Table 5 shows the characteristic of data. The platform of experiment is an IBM PC with 512 MB RAM and 3.0 GHZ CPU. We use java language implementation association rules mining algorithm.

Table 5 Data sets

Data Set Name	Data	
	Item number	Record number
Mushroom	119	8124
Connect	129	67557
Pumsb*	7117	49046
T30I1.0D50K	100	60000

5.1 Algorithm Performance Comparing

5.1.1 Running Time Comparing

Running time is an important parameter of algorithm capability, so this paper selects two classical algorithms Apriori and F-Growth* to compare with FT-Miner algorithm. Due to above-mentioned three algorithms based on different measures, this paper only compares the running time of generating frequent itemsets with the threshold value changing of Support. Therein, Csupp is equal to 0.8; N-conf is equal to 0.25.

The experimental results in Fig 8-10 show that the FT-Miner algorithm has better performance than Apriori algorithm and FP-growth* algorithm. Fig.11 shows FT-Miner has better performance than Apriori and FP-growth* with data rapid increase.

5.1.2 Weak Support Rules Number Comparing

In the performance evaluation of generating rules number, we limit the lower Support threshold value range to test the performance discovering weak-support rules number, which compares FT-Miner algorithm with Apriori and MSApriori on 10,000 transaction records randomly selecting from T30I1.0D50K. Therein, β is equal to 0.25 in MSApriori; Cspp is equal to 0.85 and N-conf is equal to 0.25 in FT-Miner; Confidence is 0.6 in Apriori. From Table 6, we can know that Apriori algorithm can only discover one weak-support mode association rule, and MSApriori also mines fewer weak-support mode association rules than FT-Miner algorithm. So, FT-Miner algorithm has better performance than MSApriori and Apriori algorithms on mining weak-support mode rules.

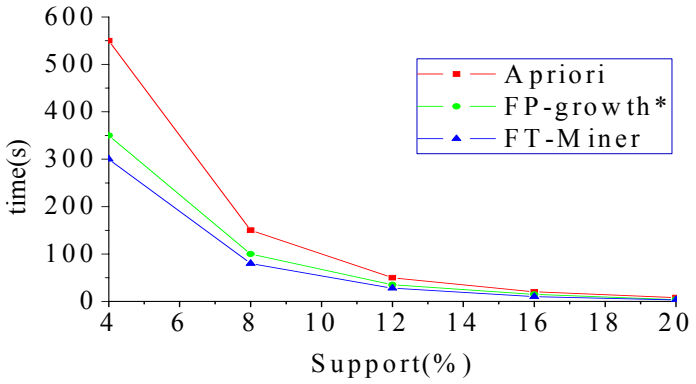

Fig.8 Running time comparing on mushroom set

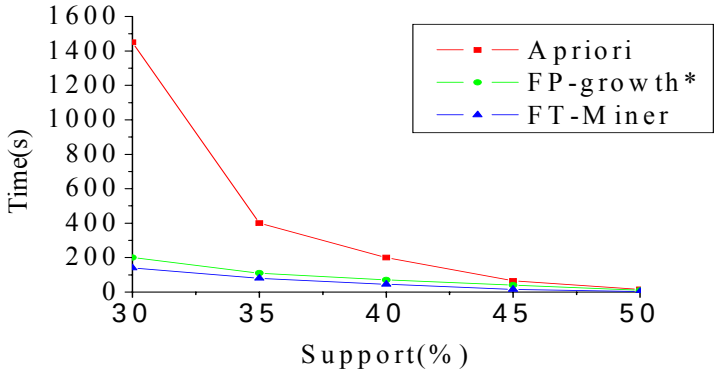

Fig.9 Running time comparing on pumsb* set

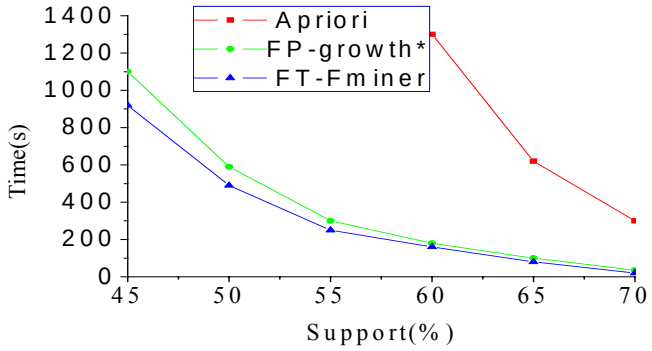

Fig.10 Running time comparing on connect set

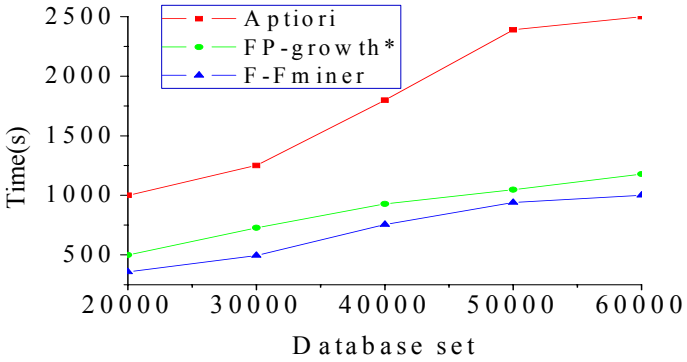

Fig.11 Running time comparing on T30I1.0D60K

Table 6 Comparing rules number with MSApriori and Apriori alorithms

Algorithm	Support（%）			
	1.2-1.9	2 -2.9	3-3.9	4-4.9
FT-Miner	8	14	7	29
MSApriori	6	22	3	15
Apriori	0	0	1	0

5.2 N-confidence Measure Performance

In order to test the action of N-confidence measure on algorithm efficiency, this paper examines the running time of FT-Miner algorithms in the condition of the given Csupp. Therein, the threshold of Support is equal to 0.15; Inter is equal to 0.6. From Fig.12, we can know that the running time of algorithm is decreasing with the N-conf increasing.

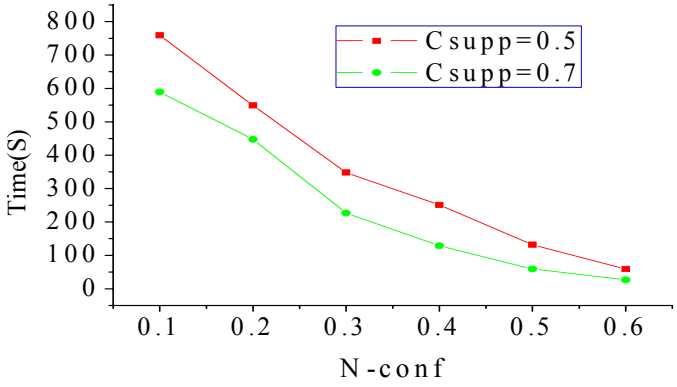

Fig.12 N-confidence measure impact for running time on connect set

5.3 New Interesting Measure Performance

To test the action of Interesting measure on algorithm efficiency, this paper examines the running time of FT-Miner algorithms in the condition of the given Csupp. Therein, the threshold of Support is equal to 0.2; From Fig.13, we can know that rules number is decreasing with the increasing of Inter, which induces the running time of algorithm reduction.

Generating rules number is also an important parameter of algorithm capability, so this paper compares it among Apriori, FP-growth*, FT-Miner and FT-Miner without Interesting on T30I1.0D60K dataset. Therein, Csupp is equal to 0.7, N-conf is equal to 0.3 and Interesting is equal to 0.6. The experimental results in Table 7 show that the FT-Miner algorithm with Interesting has better performance in Generating rules number.

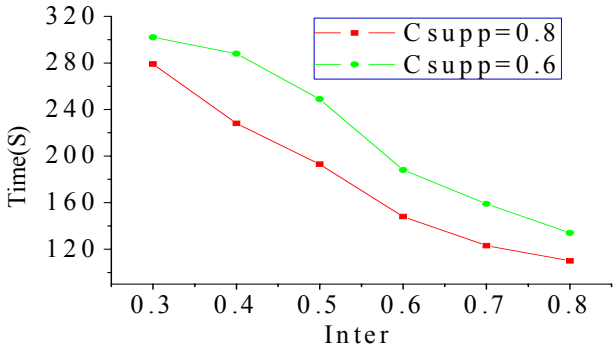

Fig.13 Interesting measure impact for running time on mushroom set

Table 7 Generation rule number comparing

Rule Number Comparing	Algorithm Name			
	Apriori	FP-growth*	FT-Miner without Interesting	FT-Miner
Rule number	3507	2410	1929	841

6 Conclusion

In this paper, we have proposed and designed a technique through which we can discover the association rules to contain low Support data items, defined as weak-support mode. The technique adopts some new measures, including Csupport, N-confidence and a new Interesting, that enables us to identify the strong correlation of the significant low Support data items with the specific data co-occurring in relatively high proportion. At the same time, two new data structures are put forward, UFP-Tree and FP-Forest, for mine association rules including weak-support mode rules. It is easy to realize large-scale database storage, and not like a single FP-Tree. Furthermore we have designed a fast rules mining algorithm, FT-Miner, with frequent growth based on depth-first searching method and dividing and conquering strategy. According to the experiments on real large data sets, FT-Miner algorithm has higher efficiency than other algorithms. So the method in this paper is very appropriate to mine association rules including weak-support mode.

Acknowledgment

This work is partially supported by the National Natural Science Foundation of China (Grant No. 70571019), the Ph.D. Programs Foundation of Education Ministry of China (No. 20060213004) and the Research Center of Technology, Policy and Management, Harbin Institute of Technology

Appendix

Proof of lemma 1.

$$Conf(\{i_1,i_2\} \Rightarrow \{i_3,i_4,\cdots,i_k\}) = \frac{Sup(\{i_1,i_2,i_3,\cdots,i_k\})}{Sup(\{i_1,i_2\})} = \frac{P(\{i_1,i_2,i_3,\cdots,i_k\})}{P(\{i_1,i_2\})}$$

$$Conf(\{i_1,i_2,i_3\} \Rightarrow \{i_4,i_5,\cdots,i_k\}) = \frac{Sup(\{i_1,i_2,i_3,\cdots,i_k\})}{Sup(\{i_1,i_2,i_3\})} = \frac{P(\{i_1,i_2,i_3,\cdots,i_k\})}{P(\{i_1,i_2,i_3\})}$$

$$\therefore P(\{i_1,i_2,i_3\}) \le P(\{i_1,i_2\})$$

$$\therefore \frac{Min\{Sup(i_1), Sup(i_2), \cdots, Sup(i_k)\}}{Max\{Sup(i_1), Sup(i_2), \cdots, Sup(i_k)\}} \le \frac{P(\{i_1, i_2, i_3, \cdots, i_k\})}{P(\{i_1, i_2)} \le \frac{P(\{i_1, i_2, i_3, \cdots, i_k\})}{P(\{i_1, i_2, i_3)}$$

$$\Rightarrow Conf\{i_1, i_2\} \Rightarrow \{i_3, i_4, \cdots, i_k\} \le Conf\{i_1, i_2, i_3\} \Rightarrow \{i_4, i_5, \cdots, i_k\}$$

Proof of lemma 2.

The formula (2) denotes that the Confidence of will not increase when the item in the left of rule move to its right. So the rule with minimum Confidence only has one item in its left for some frequent item sets.

References:

[1] R. Agrawal, T. Imielinski and A. N.Swami, Mining Association Rules between Sets of Items in Large Databases, *In: Proceedings of the 1993 ACM SIGMOD Conference*, 1993, pp.207–216.

[2] J. W. Han, J. Pei, Y. Yin, Mining frequent patterns without candidate generation, *In: Proceeding of 2000 ACM-SIGMOD International Conference on Management of Data*, 2000, pp.1–12.

[3] R. Agrawal, R. Srikant, Fast Algorithms for Mining Association Rules, *Proceeding of the VLDB Conference*, 1994, pp.487–499

[4] S. Brin, R. Motwani, J. D. Ullman, S. Tsur, Dynamic Itemset Counting and Implication Rules for Market Basket Analysis, In: Proceedings of the ACM SIGMOD Conference, Tucson, Arizona, 1997, pp.255-264.

[5] Ashoka Savasere, Edward Omiecinski, Shamkant B Navathe, An Efficient Algorithm for Mining Association Rules in Large Databases, *In: 21st International Conference on Very Large Databases*, 1995, pp.432–444.

[6] J. S. Park, M. A. Chen, P. S. Yu, An Effective Hash-based Algorithm for Mining Association Rules, *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, 1995, pp. 175-186

[7] G. Grahne, J. Zhu. Efficiently Using Prefix-trees in Mining Frequent Itemsets. *In Proceeding of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations, Melbourne,* 2003.

[8] Gui Mei Liu, Hong Jun Lu et al, AFOPT: An Efficient Implementation of Pattern Growth Approach. *In: Proceeding of IEEE ICDM Workshop on Frequent Itemset Mining. Implementations*, 2003

[9] Osmar R. Zaiane, Mohammed, El-Hajj, COFI-tree Mining: A New Approach to Pattern Growth with Reduced Candidacy Generation, *In: the first Workshop on Frequent Itemset Mining Implementations, held with IEEE ICDM* ,2003

[10] B. Liu, W. Hsu, Y. Ma. Mining Association Rules with Multiple Minimum Supports. *In: Proceedings of the ACM SIGKDD*, 1999

[11] H. Xiong, S. Shekhar, P-N. Tan, and V. Kumar, Exploiting a Support-based Upper Bound of Pearson's Correlation Coefficient for Efficiently Identifying Strongly Correlated Pairs. *In: Proc. of ACM SIGKDD'04*, 2004, pp. 334–343

[12] He Zeng-You, Xu Xiao-Fei, and Deng Sehng-chun, A FP-Tree Based Approach for Mining All Strongly Correlated Pairs without Candidate Generation. *Proc. of CIS'05, LNAI 3801*, 2005, pp.735-740

[13] Qian Tie-Yun, Feng Xiao-Nian, and Wang Yuan-Zehn, Mining Negative Correlation Rules beyond Support-Confidence Framework, *Computer Science*, vol.32, 2005, pp. 124–128

[14] E. Omiecinski, Alternative interest measures for mining associations, *IEEE Trans. Knowledge and Data Engineering*, vol.15, 2003, pp. 57–69

[15] Y. K. Lee, W. Y. Kim, Y. D. Cai, and J. Han, CoMine: efficient mining of correlated patterns, *In: Proc. ICDM'03*, 2003, pp. 581–584.

[16] W. Y. Kim, Y. K. Lee, and J. Han, CCMine: Efficient Mining of Confidence-Closed Correlated Patterns, *In: Proc. of PAKDD'04*, 2004, pp. 569–579.

[17] J. W. Han, M. Kambr. Han J and Kamber M. Data Mining: Concepts and Techniques, *Los Altos, CA, USA: Morgan Kaufmann Publishers*, 2001, pp.149-183.

[18] G. Piatetsky-Shapiro. Discovery Analysis and Presentation of Strong Rules, *In: Knowledge Discovery in Databases, edited by G. Piatetsky-Shapiro and W. Frawley*, 1991, pp. 229-248.

[19] P. Smyth, R. M. Goodman. Rule Induction Using Information Theory. Knowledge Discovery in Database, *AAAI/MIT Press*, 1991

[20] M. Klentettinen, H .Mannila, PRonkinen, et al. Finding Interesting Rules from Large Sets of Discovered Association Rules, *In: Proceedings of the3th International Conference on Information and Knowledge Management*, Gaithersburg, Maryland, USA,1994, PP. 401-407.

[21] C. J. Merz, P. Merphy. UCI Repository of Machine Learning Databases, 1996. (Http://www.ics.uci.edu/)