INTRODUCTION

Data mining \cite{1} is the process of extracting unknown but potential and useful information from enormous, incomplete, noisy, fuzzy, and random practical application data. The information from the data mining is effective and practical. The core modules of data mining technology include statistics, artificial intelligence, and machine learning. Data mining is the most popular area in database research, and association rule is an important topic of KDD (Knowledge Discovery in Database) research that American R. Agrawa started firstly. The discovery of association rules in two steps: firstly, iterate and identify all the frequent item set with the requirement support is not lower than minimum support set by users; secondly, build credibility no lower than users-setting minimum confidence from the frequent item set to identify all the frequent item sets. This is the core of Association Rule discovery algorithm.

ASSOCIATION ALGORITHM ANALYSIS: BASIC CONCEPTS AND ALGORITHMS

Many business enterprises accumulate large quantities of data from their day-to-day operations. For example, huge amounts of customer purchase data are collected daily at the checkout counters of grocery stores. Table 1 illustrates an example of such data, commonly known as market basket transactions.
Each row in this table corresponds to a transaction, which contains a unique identifier labeled TID and a set of items bought by a given customer. Retailers are interested in analyzing the data to learn about the purchasing behavior of their customers. Such valuable information can be used to support a variety of business-related applications such as marketing promotions, inventory management, and customer relationship management. Association analysis is useful for discovering interesting relationships hidden in large datasets. The uncovered relationships can be represented in the form of association rules or sets of frequent items. For example, the following rule can be extracted from the data set shown in Table 1.

Table 1. An example of market basket transactions.

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{Bread, Milk}</td>
</tr>
<tr>
<td>2</td>
<td>{Butter}</td>
</tr>
<tr>
<td>3</td>
<td>{Beer, Diaper}</td>
</tr>
<tr>
<td>4</td>
<td>{Bread, Milk, Butter}</td>
</tr>
<tr>
<td>5</td>
<td>{Bread}</td>
</tr>
</tbody>
</table>

The rule suggests that a strong relationship exists between the sale of butter, bread and milk meaning that if butter and bread are bought, while customers also buy milk. Retailers can use this type of rules to help them identify new opportunities for cross selling their products to the customers.

Besides market basket data, association analysis is also applicable to other application domains such as bioinformatics, medical diagnosis, web mining, and scientific data analysis. In the analysis of Earth science data, for example, the association patterns may reveal interesting connections among the ocean, land, and atmospheric processes. Such information may help Earth scientists develop a better understanding of how the different elements of the Earth system interact with each other. Even though the techniques presented here are generally applicable to a wider variety of data sets, for illustrative purposes, our discussion will focus mainly on market basket data. There are two key issues that need to be addressed when applying association analysis to market basket data. First, discovering patterns from a large transaction data set can be computationally expensive. Second, some of the discovered patterns are potentially spurious because they may happen simply by chance.

2.1 Basic concepts

Following the original definition proposed by Agrawal et al.,\[2\] the problem of association rule mining is defined as:

Let \( I = \{i_1, i_2, \ldots, i_n \} \) be a set of n binary attributes called items.

Let \( D = \{t_1, t_2, \ldots, t_m \} \) be a set of transactions called the database.

Each transaction in \( D \) has a unique transaction ID and contains a subset of the items in \( I \).

A rule is defined as an implication of the form:

\[
X \Rightarrow Y \quad \text{where} \quad X \subseteq I \land X \cap Y = \phi
\]

Every rule is composed of two different set of items, also known as item sets, \( X \) and \( Y \), where \( X \) is called antecedent or left-hand-side (LHS) and \( Y \) consequent or right-hand-side (RHS).

To illustrate the concepts, we use a small example from the supermarket domain. The set of items is \( I=\{\text{Bread, Milk, Diapers, Beer, Cola}\} \) and in the Table 2, the A binary 0/1 representation of market basket data, is shown a small database containing the items, where, in each entry, the value 1 means the presence of the item in the corresponding transaction, and the value 0 represents the absence of an item in a that transaction.

Table 2. A binary 0/1 representation of market basket data.

<table>
<thead>
<tr>
<th>TID</th>
<th>Milk</th>
<th>Bread</th>
<th>Butter</th>
<th>Beer</th>
<th>Diaper</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

An example rule for the supermarket could be \{butter, bread\} \Rightarrow \{milk\} meaning that if butter and bread are bought, customers also buy milk.

In order to select interesting rules from the set of all possible rules, constraints on various measures of significance and interest are used. The best-known constraints are minimum thresholds on support and confidence.

Let \( X \) be an item-set, \( X \Rightarrow Y \) be an association rule and \( T \) be a set of transactions of a given database.

Support: The support value of \( X \) with respect to \( T \) is defined as the proportion of transactions in the database which contains the item-set \( X \). In formula: \( \text{sup}(X) \).

Confidence: The confidence value of a rule, \( X \Rightarrow Y \), with respect to a set of transactions \( T \), is the proportion of the transactions that contains \( X \) which also contains \( Y \).

Confidence is defined as:

\[
\text{conf}(X \Rightarrow Y) = \frac{\text{sup}(X \cup Y)}{\text{sup}(X)}
\]

For example, the rule \{butter, bread\} \Rightarrow \{milk\} has a confidence of 0.2/0.2=1.0 in the database, which means that for 100% of the transactions containing butter and bread the rule is correct (100% of the times a customer buys butter and bread, and milk is bought as well).
2.2 The algorithms

Association rule research can be divided into two sub-tasks:[2]

1. Find out all the frequent item set in data set D according to the minimum support.
2. Generate association rule according to frequent item set and minimum confidence.

The first task is to find all the frequent item set in D. It’s the main problem and standard judgment of association rule research. Most studies are focused on this.

Apriori algorithm[3,4] is one kind of hierarchical association rule research. Most studies are focused on this.

The main problem and standard judgment of association rule research. Most studies are focused on this.

Apriori algorithm[3,4] is one kind of hierarchical association rule research. Most studies are focused on this. It’s the main problem and standard judgment of association rule research. Most studies are focused on this.

The first task is to find all the frequent item set in D. It’s the main problem and standard judgment of association rule research. Most studies are focused on this. Apriori algorithm[3,4] is one kind of hierarchical association rule research. Most studies are focused on this.

AprioriTid Algorithm[5] is the classic association rule algorithm which belongs to association rule research, meanwhile, it’s a common algorithm with good combination property. Apriori algorithm needs multi-steps processing to search frequent item set: the first step is to find out 1 frequent item set, and then circularly process until there is no generation of frequent item set. When it comes the K cycle, use apriori-gen function to generate candidate 1-dimension frequent item set. Then C stores K-dimension candidate item set k+1, then the number of frequent item set k must greater than k.

It can be proved from Property 1 that in Lk+1 the different k+1 item of k subset must belong to set frequent item set k.

Property 3: The support of any piece of transaction in frequent item set Lk support at least k pieces of k-1 item set in Lk-1; Property 4: The non-frequent item in AprioriTid transaction data Ck can be dismissed when calculating Lk-1.

According to proof by contradiction, if non-frequent item cannot be dismissed, then it supports Lk+1, which stands against Nature 1.

3 OPTIMIZED APRIORITID ALGORITHM

3.1 The feature of optimized AprioriTid Algorithm as follows

Property 1: All the nonvoid subsets in any frequent item set are frequent item set, and the superset of non-frequent item set is non-frequent item set.[5]

Property 2: If frequent item set k can generate item set k+1, then the number of frequent item set k must greater than k.

It can be proved from Property 1 that in Lk+1 the different k+1 item of k subset must belong to set frequent item set k.

Property 3: The support of any piece of transaction in frequent item set Lk support at least k pieces of k-1 item set in Lk-1.

Property 4: The non-frequent item in AprioriTid transaction data Ck can be dismissed when calculating Lk-1.

According to proof by contradiction, if non-frequent item cannot be dismissed, then it supports Lk+1, which stands against Nature 1.

3.2 The optimization idea of AprioriTid

(1) The improved algorithm based on transaction item set compression

In the k-th step, during the traversing of candidate item set Ck in Ck-1,c∈Ck . If the number of potential large item set is less than or equal to 1 in any transaction record, then delete the transaction record directly.

Authentication: in k-th step, c∈Ck , during the process of Ck traversing Ck-1 and generating Ck, should judgment of Ck-1’s each pieces of transaction record whether included in or not in c-[k-1] and c-[k],
therefore, each piece of transaction record contains at least two or more items.

(2) The improved algorithm based on candidate item set compression [8]

Before association rule digging, algorithm needs to preprocess the original data and build data dictionary to make attributive classification and simple number equivalent. In this paper, the algorithm ranks data set by dictionary ascending order, in order to reduce the candidate item set, when generate the K candidate itemsets just keep the support is greater than the minimum support degree of itemsets, while K ≥ 2 generate candidate itemsets k Tid item set in the table with the set instead of the item set (just like this k \{c \in L_{k-1}; c \in t\} >) and this method could reduce the data storage. After getting the frequent item set L_k, we should count for each item. If the item’s number is less than the minsup, we should remove it and use a new frequent itemsets L_k to replace the original L_k frequent itemsets, which reduces the next time the number of candidate itemsets generation and also the storage space and time.

3.3 Improved AprioriTid Algorithm pseudocode

The improved algorithm is as follow:

1. \( C_1 = D \)
2. \( L_1 = \text{getFreq1Itemset}(C_1) \)
3. \( L'_1 = L_1 \)
4. for(\( k = 2; (L'_{k-1} \neq \Phi) \& \&(C_{k-1} \neq \Phi); k++ \}){
5. \( C_k = \text{Apriori_gen}(L_{k-1}) \)
6. \( C_k = \Phi \)
7. if(transaction t ≤ k)
8.     continue
9. else if(c \in C_k)\{
10.     if(c \in t)\{
11.         C_i = C_i \cup c
12.         c.sup++
13.     }\)
14. }\)
15. if(c \in C_k \& \&(C_i.sup \geq \text{min sup})
16. \{ \( C_i = C_i \cup (< \text{tid}, C_i >) \)\}
17. \( L_k = \{c \in C_i | c.sup \geq \text{min sup}\} \)
18. \( L_k = \text{getFreqKItemSet}(L_k) \)
19. }

4 EXPERIMENT AND THE RESULT

To verify the efficient of the improved algorithm, this experiment compares the frequent item set generating-time of improved algorithm (named NewAprioriTid) and AprioriTid Algorithm under different support.

To better verify NewAprioriTid algorithm efficiency so we use the UCI standard test data set – Mushroom data set. UCI standard test data is the popular data set in data mining, so it can be got freely from the Internet and has much authority. Experimental environment is Windows 7 operating system, 2.50 GHz CPU, 4.0 GB Memory, and the JAVA language as the development language.

4.1 Comparison between the original algorithm and the improved algorithm and analysis

(1) Fix data centralized transaction number and compare the original algorithm and the improved algorithm under different supporting threshold value. Choose 4,000 data randomly in transactional databases, and then get the different running time in Table 3.

<table>
<thead>
<tr>
<th>Support</th>
<th>0.5</th>
<th>0.55</th>
<th>0.6</th>
<th>0.65</th>
<th>0.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>The run time of AprioriTid (ms)</td>
<td>340</td>
<td>330</td>
<td>304</td>
<td>283</td>
<td>261</td>
</tr>
<tr>
<td>The run time of NewAprioriTid (ms)</td>
<td>313</td>
<td>290</td>
<td>270</td>
<td>242</td>
<td>214</td>
</tr>
<tr>
<td>Improvement efficiency</td>
<td>8%</td>
<td>12%</td>
<td>11%</td>
<td>14%</td>
<td>18%</td>
</tr>
</tbody>
</table>

To compare the running efficiency clearly this paper takes Figure1 as description.

![Figure 1. The run time of 4000 transactions.](image-url)
improved algorithm takes reducing 10%-20%. With the increase of threshold value, the time-saving efficiency of improved one becomes more apparent. In the beginning of the program operation, the lower supporting threshold value makes condition-satisfying frequent item set and association rule become more, so running time is relatively long. With the increasing of support, rangeability of running time becomes less. This shows the superiority of improved algorithm. Due to the high support at the beginning of program, it restrains the generation of large number of candidate item set and frequent item set which satisfy lower support and saves a lot of time to make the rangeability bigger.

(2) When comparing the efficiency under different transaction, take minimum support 0.5, and then randomly take 5000,6000,7000,8000 data to compare and analysis. The results are as follow in Table 4.

<table>
<thead>
<tr>
<th>The number of transaction set</th>
<th>5000</th>
<th>6000</th>
<th>7000</th>
<th>8000</th>
</tr>
</thead>
<tbody>
<tr>
<td>The run time of AprioriTid (ms)</td>
<td>263</td>
<td>291</td>
<td>326</td>
<td>354</td>
</tr>
<tr>
<td>The run time of NewAprioriTid (ms)</td>
<td>227</td>
<td>258</td>
<td>283</td>
<td>314</td>
</tr>
</tbody>
</table>

For better comparison, the data are described as Figure 2.

![Figure 2. The efficiency of different transaction.](image)

It can be seen that the running time of algorithm and improved one increase with the bigger transaction set with the fixed support 0.5 condition. It shows that the general tendencies of the two algorithms are the same. Meanwhile, the improved algorithm has advantages in time increasing rate

5  CONCLUSION

Association rule digging problem can be divided into two sub-tasks: find frequent item set and generate association rule. Find frequent item set is the core of association rule digging. This paper proposed an improved AprioriTid algorithm based on the transaction set compression and candidate item set compression. On the one hand, it compresses the size of data set efficiently and reduces the times of scanning data set; on the other hand, it reduces the candidate item set and improves the algorithm. The experimental result has shown that the improved AprioriTid Algorithm is better than the original one.

REFERENCES