CHAPTER 1

Introduction

Association rule mining is one of the most basic and important topics in Data Mining area. Its goal is to discover the relationship of set of items from large database of customer transactions in supermarket management problem [1]. A typical example of such an association rule is the statement that customers who purchase diaper and milk also purchase beer. To discover the association rule, we have to find out the frequent itemsets first. An itemset is a set collection of items, such as {diaper, beer, milk}, and the term "frequent" means that the itemset appears no less than a given number, minimum support, of times. After discovering frequent itemset, the association rule can be generated. Figure 1.1 gives a brief example of frequent itemset mining, while the given minimum support is three.

Algorithms for the frequent itemset mining have been investigated for a long time, such as Apriori [2], FP-growth [3], Eclat [4], Tree-Projection [5], H-Mine [6], DHP [7], and so on. The essence of frequent itemset mining is to discover frequent subsets from a set of itemset. In this thesis, we propose a novel mining task to discover frequent superset, rather than frequent subset. We want to find patterns that are superset of a certain number of transactions. For the clarity of description, in the following, we use the term "frequent subset" to denote the traditional frequent itemset, and "frequent superset" to this new problem.

Database	
TID	Items
1	Beer, Bread, Milk
2	Beer, Diaper, Milk, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Bread, Diaper, Milk

Frequent Itemset	Support
Bread	3
Beer	4
Diaper	4
Milk	5
Bread, Milk	3
Beer, Diaper	3
Beer, Milk	4
Diaper, Milk	4
Beer, Diaper, Milk	3

Figure 1.1: An example for the discovery of frequent itemset.

Definition 1.1 Let $D=\{T_1, T_2, \ldots, T_k\}$ be a transaction database with k transactions, X be an itemset, and minimum support is h. If $T_{i_j} \subseteq X$, where $1 \leqq i_j \leqq k$, for each $j=1, 2, \ldots, m$, m is referred to as the support of itemset X. If $m \geqq h$, we say that X is a frequent superset.

Database	
TID	Items
100	1, 3, 4
200	3, 5
300	1, 5
400	5
500	2, 4

Figure 1.2: An example database.

Example 1.1 Figure 1.2 is an example of a transaction database D, the support of itemset {1, 3, 5} is 3, because there are three transactions, {3, 5}, {1, 5}, and {5} which are subsets of {1, 3, 5}. Given the value of minimum support 3, the frequent supersets are {1, 2, 3, 4, 5}, {1, 2, 3, 5}, {1, 2, 4, 5}, {1, 3, 4, 5}, {2, 3, 4, 5}, and {1, 3, 5}.

Intuitively, the basic approach to solve the problem of frequent superset mining is just a little modification of the original Apriori algorithm for frequent subset mining. The original Apriori property states that all nonempty subsets of a frequent subset must also be frequent [8]. In other words, if an itemset is not frequent, then its superset is not a frequent itemset. Therefore, in the Apriori algorithm, the level-wise generation of frequent itemsets is employed.

However, the Apriori property cannot be directly applied to the problem of frequent superset mining. Because if an itemset X is not a frequent superset, it is possible that the itemset which is a superset of X is frequent. Therefore, we cannot use k-itemset to prune and explore (k+1)-itemset in the same way as the original Apriori algorithm does. Although the level-wise characteristic of Apriori can reduce the number of scan database, but the search space can not been lessened. This method is used as the Baseline approach for performance comparison among the proposed approaches.

Certainly, we can utilize the property that if an itemset X is not a frequent superset, then the itemset that is a subset of X is not frequent either. Based on this property, we can develop the algorithm that (k+1)-itemset are used to prune and explore k-itemset. However, this algorithm starting the exploration from a long pattern is inefficient. For example, suppose we have totally 100 items, the first step is starting from a 100-itemset, and then generates 99-itemsets with the amount of 100.

In order to solve this problem, we propose three algorithms to discover such frequent supersets. These are Apriori-based, Eclat-based, and data complement technique (DCT), presented in section 3.3, 3.4, and 3.5, respectively. The Apriori-based algorithm, named

Apriori-C, is a breadth-first search and counting occurrence method [9]. It can really adopt the Apriori property and reduce both the number of scan database and the size of search space. The Eclat-based algorithm, named Eclat-C, is a depth-first search and transaction-ID intersection method [9]. It is beneficial for mining frequent superset when the patterns are long. The last algorithm, DCT, can utilize the original algorithms of frequent itemset mining as a black box to discover the frequent superset.

In our experiments, we assess the performance of these algorithms based on the following four parameters, minimum support, number of transactions, number of items, and the average size of transactions. In DCT algorithm, we choose the Apriori, Eclat, and FP-Growth as the black box to compare with the proposed Apriori-C and Eclat-C algorithms. In addition, we evaluate the effect of the number of candidate itemsets in each level for the Apriori-based algorithms, Baseline method and Apriori-C.