A BDD-based Method for Mining Association Rules

6AA－2

Li JIANG, Mary INABA, Hiroshi IMAI

Dept.of Information Science, Faculty of Science, University of Tokyo

1 Introduction

Mining for association rules in large databases is an important data mining problem. Many algorithms have been proposed since the problem was first introduced in 1993. Basically, the procedure of mining association rules can be divided into two steps: first find out all frequently appearing sets of items (large itemsets), then calculate association rules using the large itemsets. Once all the large itemsets are obtained, the generation of association rules is quite simple.

In Jiang, Inaba, Imai[3], we first proposed a new approach for generating large itemsets by using Binary Decision Diagram (BDD). Extensive experiments using BEM-II[4] package are conducted to evaluate large itemset generation performance, and the results show that BDD represents and manipulates the large itemsets efficiently. In this paper, we will first review some of our previous work, then show how to improve the basic BDD algorithm[3] to cope with the actual large database. At last, we discuss the possibility of applying BDD to sampling method. Also the relation between the prime implicants of BDD and the candidate sets will be carefully examined.

2 Our Previous Work

2.1 Problem Description

Suppose that we are given a large database of sales transactions. Each transaction consists of items purchased by a customer in a shopping. As an example, let $E = \{1, 2, 3, 4\}$ represent the set of items that are being sold in a supermarket. Suppose that 40% of the customers bought 1,2, and 3. And among these customers, 70% of those who bought 1 and 2 also bought 3. We call the number 40% and 70% support and confidence, respectively. Then we get the association rule:

$$\{1, 2\} \Longrightarrow \{3\}; s40\%, c70\%$$

What we want to do here is to find out all the association rules, which meet the given minimum support and minimum confidence requirement.

2.2 Binary Decision Diagrams

Binary Decision Diagram (BDD) is a simple and efficient way to represent Boolean functions. With BDD, symbolic Boolean functions can be constructed, manipulated by simple and efficient graph algorithms. Here we give a brief explanation about BDD. We use the same example mentioned above, and suppose that $F = \{\emptyset, \{1\}, \{2\}, \{3\}, \{4\}, \{1, 2\}, \{2, 3\}\}$ is the itemsets which meet the min-support requirement. The Reduced Ordered BDD (ROBDD) which represents the set F is shown in Figure 1. For details, see Minato[4].

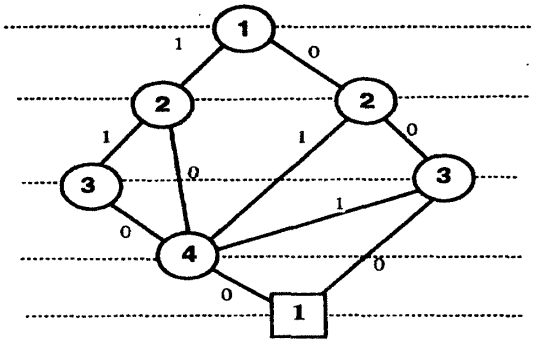

Figure 1: ROBDD

2.3 BDD Approach

According to the BDD approach, first, all the items are assigned with a different symbol each. Then we represent each transaction by a logical expression consisting of these symbols. In fact, the logical expression indicates all the subsets of the set containing the items appearing in that transaction. With these expressions, we can calculate all the large itemsets by the mixture of arithmetical and logical operations while constructing the BDD.

3 BDD Partition Algorithm

One problem the method mentioned above faces is that, when constructing BDD, intermediate BDD results cost much more memory space than the actual amount used by the final shared-BDD. And the more the symbols or the transactions, the more memory needed. In order to reduce the memory consumption, we divide the transactions into several blocks. Within each block, its sub-BDD is built separately. After that, we just enumerate all the combination of the sub-BDD results and calculate the large itemsets which meet the minimum support requirement. To evaluate the efficiency of this algorithm, we generate synthetic database consisting of 100 kinds of items and 1000 transactions and conduct several experiments on

BDD を用いたデータマイニング
姜 力、稲葉真理、今井浩
東京大学 理学部 情報科学

Table 1: BDD Partition Algorithm

Number of blocks	time (s)	total size of intermediate BDD	size of final BDD
1	432	4391550	4836
2	657	3156363	4836
4	432	2303922	4836

a Sun Ultra 170E workstation with 256MB memory. Table 1 shows the changes of the time, size of intermediate BDD and the size of final result by partitioning. From the result, we can see that the memory space needed for constructing BDD decreases rapidly as the number of the blocks increases.

Although we can see that consumption of memory decreases with the above method, there are still some problems. Dividing the database into blocks can save memory. However, as the number of blocks grows up, the combination of all the sub-BDD results which we should check may blow up. And the more blocks, the more time consumed in the computation. After all, all the results, including the sub-BDD, should be stored on the memory during the calculation. So the memory space is quite critical for the processing.

An different method using partition is introduced in Savasere[5]. The main idea is to first divide the transactions into blocks of reasonable size, which can be processed within the existing memory. Next, just fetching one block of transactions and apply the existing data mining algorithm, then save the large itemsets found in the local block and make room for the next block by pruning the current data out of memory. Finally, we get all the "local large itemset" from each block. We take this as the candidate set of potentially large itemsets and pass the whole database again to count each candidate and discover the real one.

In this approach, BDD-algorithm is practical when mining the local large itemsets within each block.

4 Sampling

Sampling[6] is a powerful method for data mining. Instead of completely processing all the data, one can just take a small amount of data by a random sampling and analyze the sample. Usually, we use a lowered support to find the potential large itemset in the sample in order to avoid missing a real one. Then use the rest of the data to see whether the regularities found in the sample hold throughout the whole database. Moreover, it is not sufficient to just rely fully on the candidates found in the sample, *border set*, which consisting of those most likely to be large, needs to be checked to guarantee that we never miss a rule.

Here we know the fact that all the subset of a large itemset should be large, so the inclusion relation of the large itemset is monotone and all the subsets are independent. Therefore, the generation of border set is correspond to the computation of the prime implicant of the original BDD.

We take the same example as we used in previous section to illustrate this briefly. Also assume that $F = \{\emptyset, \{1\}, \{2\}, \{3\}, \{4\}, \{1,2\}, \{2,3\}\}$ is the large itemsets. Here we define function $f(x)$ as follows:

$$f(x) = \begin{cases} 1 & x = \chi_S \text{ from some } S \in F \\ 0 & \text{otherwise} \end{cases}$$

(χ_S is the characteristic vector of S)

Obviously, $f(x)$ is a monotone negative function whereas $\bar{f}(x)$ is a monotone positive function. The prime implicants of $\bar{f}(x)$ corresponds to set $B = \{\{1,3\}, \{1,4\}, \{2,4\}, \{3,4\}\}$. Here we choose the minimal itemsets from B to construct the border set. Also, the prime implicants of $f(\bar{x})$ corresponds to $\{\{1,2\}, \{2,3\}, \{4\}\}$, which are the maximal sets among large itemsets.

Algorithms to compute border set, maximal large itemsets from F, and to compute F from border set or maximal large itemsets via BDD and their complexity are discussed in Hayase[1] (see also Hayase, Imai[2]), and we can apply them to our data mining procedure.

5 Conclusions

In this paper, we have presented new approach for mining association rules using BDD. By partitioning the transactions table into a number of blocks, the BDD-based method can be efficient coping with large database. Besides, we also discussed the efficiency of BDD when used in sampling method. When sampling, the border set can be obtained by computing the prime implicants of the original BDD. As for future work, we are considering to seek for additional benefit by using BDD, such as deriving negative association rules, the possibility to compute the final association rules from the itemsets by logical operations.

References

[1] K. Hayase: On the Complexity of Constructing OBDDs of a Monotone Function and of the Set of Its Prime Implicants. *Master's Thesis*, Department of Information Science, University of Tokyo, March 1996.

[2] K. Hayase and H. Imai: OBDDs of a Monotone Function and of Its Prime Implicants. *Proc. ISAAC'96*, Lecture Notes in Computer Science, Vol.1178, 1996, pp.136–145.

[3] L. Jiang, M. Inaba and H. Imai: An Application of BDD to Mining Association Rules. *Proc. DEWS'97*, 1997, pp.61–66.

[4] S. Minato: *Binary Decision Diagrams and Applications for VLSI CAD*, . Kluwer Academic, 1996.

[5] A. Savasere, E. Omiecinski, and A. Navathe: An efficient algorithm for mining association rules in large databases. *Proc. 21th VLDB Conf.*,1995, pp.432–444.

[6] H, Toivonen: Sampling Large Databases for Association Rules. *Proc. 22th VLDB Conf.*,1996, pp.134–145.