An Algorithm of Association Rule Mining Based on Memory Indexing

Yong Huang
School of Management, Dalian University of Technology, Dalian 116024, China
Kuanjiu Zhou
School of Management, Dalian University of Technology, Dalian 116024, China
Huamei Sun
School of Management, Harbin Institute of Technology, Harbin 150001, China
Tiyun Huang
School of Management, Harbin Institute of Technology, Harbin 150001, China
tyhuang@hit.edu.cn

***Abstract*-In the field of data mining, Apriori is a classical algorithm for mining association rules. In this paper, an improved Apriori algorithm is proposed, which is based on the principle of memory indexing. By recording unnecessary scanning transactions and storing relatively frequent transactions, the algorithm decreases the scanning times of the database, and consequently save the time of I/O operation. The experiments result has verified the effectiveness of the improved algorithm.**

Keywords-Data mining, association rules, memory indexing

I. INTRODUCTION AND RELATED WORKS

Apriori algorithm is a classical algorithm of mining association rules, which is focusing on finding the relation of different items in the database, by means of mining specific frequent itemsets. Originally, Apriori algorithm is mainly used for Boolean Association Rules, and later extended to Quantitative Association Rules and Categorical Association Rules.

A variety of algorithm of mining association rules are developed from Apriori algorithm. Wanjun Yu et al. extract frequent itemsets from candidate itemsets[1]. Yuan Jian et al. employ vector calculation to count the frequent itemsets[2]. Lu SF et al. combine the bottom-up and top-down searching strategy to decrease the number of candidate frequent itemsets[3].

Sometimes, the rules needed may involve several different concept levels. Jiawei Han et al. discuss the association rules at multiple concept levels[4]. Raymond et al. propose a association rules mining algorithm with item constraints and give a classification of the constraints[5].

In this article, based on the principle of memory indexing, we propose a improved Aprior algorithm, which has the ability to decrease the scanning times of the database, and increase the effectiveness of I/O operation.

II. ALGORITHM INTRODUCTION

By analyzing the process of Apriori, we know that when the algorithm is conducted to a certain extent, some transaction in the database, not having impact on the calculation results any more, will still be scanned and calculated also. It is obvious that the more item-numbers of one specific transaction, the more frequent the transaction will be scanned and calculated. Therefore, if we can set up a mechanism to decrease the time of I/O operation, the effectiveness of data mining will be enhanced. In light of this idea, we establish a data structure in the memory, which can record unnecessary scanning items and caching the relatively frequent scanned transactions. In addition, a quick indexing table is also built to store some recording states, such as whether a specific transaction should be scanned, and whether it has been cached. As a supplement data structure, the configuration of the designed quick indexing table is shown as follows in table I.

In table I, the item TID is a sign to represent the respective transactions. The transactions in the quick indexing table are stored in a specific sequence. When flag equals zero, it means that transaction need not to be scanned. And when flag is not zero, it means that transaction has been stored. In this situation, the item Cached-address records the beginning physical address of the transaction in the memory, and the non-zero flag also indicates the item count of that transaction. For example, from the second record in table I, the flag value is 3, so we know that the item count of T900 transaction is 3, and we can read 3 units from the memory address 0x0004 to acquire the detail information of that transaction.

Employing the proposed strategy, some records that need not be scanned could be stored into the designed table like table I. With the iterative process, more and more non-need-scanned transaction will be added into the table. Consequently, the count of need-to-be-scanned transactions will become less and less, and the I/O operation time could be significantly saved. The advantages of this strategy are as follows. First of all, it is an approach to virtually delete the records in large database. Without changing or destroying the original structure of the database, the database can be used for other data mining process. Moreover, in the situation of supermarket purchasing, because the customers tend to buy only several commodities in one time, this virtually deleting method is beneficial to eliminate considerable quantities of transaction record in the subsequent mining process. Hence, the core of data operation will be transferred from the main cache to the memory, and in the remained less records, the time of higher multi-dimensional data mining could be largely saved.

Published by Atlantis Press, Paris, France.

III. ALGORITHM DESCRIPTION

The core of the algorithm is applying the record in the memory to decrease the scanning time of database. There exist two main problems: ①Which record should be cached? ②which and when the record should be picked out from the cache? In light of the facts in the above-mentioned section, the transaction with relatively large item count should be stored into the cache. When sub-k itemset is created, the transaction with k item count will be picked out from the cache. If so, the problem is how to determine a threshold, which is named as a parameter *min_length*. And it is regulated that the transaction with its item count larger than *min_length* will be added into the cache. According to the value of *min_length*, in the succeeding scanning process after the first round, the qualified transaction will be affiliated into the cache.

Use D to represent the database, | T | to represent the item count of transaction T in the database, | d | to represent the length of itemset d. The process of the algorithm is described as follows.

Firstly, initialize the quick indexing table, and set up a caching space in the memory. For example, 5% of system memory can be reserved as the cache space, using parameter cache_size to describe the space volume.

Secondly, conduct the first iteration of Apriori, and generate L1. This step can be divided into two separate sessions.

(1) According to the scale of the database, draw a specific part as the sampling, like 1%, to statistic the item count of transaction. Relatively speaking, randomly picking method is more credible than fixing area picking. But considering that the randomly picking will increase additional time to seek the track, we draw a continuous region as the sampling area. Set *T[i]* as the count of transaction with scale i, s as the one itme's memory space of the transaction. Then parameter min_length can be calculated as the largest i which matches the following inequality: (i * T[i] + (i+1) * T[i+1] + … + max * T[max]) * s > cache_size * 0.1% .

(2) In the remaining scanning, the transaction which item count is larger than min_length will be added into the quick indexing table, using flag to represent the item count. Then save the itemset of transaction item ID in the cache, and add the item with length equals 1 into the quick indexing table, where flag = 0. In addition, in the subsequent scanning process, calculate continuously the value of T[i], and when the scan is completed, obtain the value of min_length by calculating the inequality (i * T[i] + (i+1) * T[i+1] + … + max * T[max]) * s > cache_size.

Thirdly, conduct the kth iteration (k>1). In the process of these interations, only the value of TID column in the database is necessary to be scanned and read. Then, seek the TID read in the quick indexing table. If founded, it means that the detail content of the transaction is not necessary to be read from database. And if the value of flag is zero, it means that the scale of the transaction is less than k, and it is not necessary to read the content of its item; while if the value is non-zero, it means that the content of that transaction has been stored into cache, and it could be got out according to the parameter cached_addr.Moreover, if the scale of the transaction equals k, it should be screened out from cache, set flag as 0, and delete the value of cached_addr. If flag is designated as 0, the corresponding transaction content wouldn't be scanned and read in the next iteration. In the iteration, linking and trimming approach is similar to classical Aprior. In other words, the function of apriori_gen( ) and has_infrequent_subset( ) could be used in the process.

Fourthly, continuously repeat the fourth step until the end of the Aprior.

IV. ALGORITHM APPLICATION EXAMPLES

The proposed algorithm is applied into a failure analysis scenario of an enterprise to verify its effectiveness.

A. Problem description

An enterprise is intended to analyze the reason and features of their equipment failures in mass data collected from equipment repairing process. The analysis results can provide a support for improvement of product quality of the producing department, and for purchasing strategy of the purchasing department. The specific objectives are, in light of the maintaining data from the repairing department, analyzing the cause of the equipment, using the association rules to mine the regular pattern of the fault, and helping the design department to improve the product design to enhance the quality of the designed product.

In the experiment, we use the real data from a computer main board enterprise in Taiwan, and mine the data complied with the criterions of CRISP-DM model. The data mining process is conducted and realized in the platform of VC++ 6.0, and Microsoft SQL Server 2000 in Windows XP operation system.

B. Data mining process

In the stage of data preparation, the data set sample provides some information about the current and past maintaining record of the accessories. In the stage of data selection, the data from several relevant tables, like fault analysis fact table, main board information table, fault components dimension table, and accessory information table, are collected and analyzed to get the basic data needed to be used in the mining process. In this paper, we have selected the repairing record in one region from 2002 to 2005. After the screen process, 224523 records are selected. Table II shows the attributes of each record. For the limited space, the fields from 11 to 22 are abbreviated, which are fault 2 to fault 7. Table III shows the correspondence between original relation database attributes and the transformed transaction database after the stage of data transforming. According to table III, the transaction database after the data transforming is listed in table IV. In this case, the designated minimal support is

12.5% and the minimal confidence is 50%. Table V shows the mined association rules partially.

C. Analysis of data mining results

In light of the association rules in table V, we give the following analysis and summarization.

Rule 1 to rule 5 indicate that: ①it is subject to happen faults in the season of winter and spring in the area of north china and west north china. It's probably related to the dry and cold whether in these seasons of those areas. ②In the area of south china, the fault rate is high in summer. The reason may be the hot weather. The results reveals that fault rate has a significant relationship with the rough weather and remind the enterprise to analyze deeply about the defect of product design, especially in the season of autumn and spring.

Rule 6 and rule 7 indicate that one type of accessories is easy to be fault in winter, and another type in summer. The results suggest the enterprise to improve the temperature performance of these two accessories.

Because of the limited space, we will no give more discussion about the other rules.

V. CONCLUSIONS

In this paper, an improved Aprior algorithm is proposed to enhance the speed of data mining. By designing a special data structure to record the transactions that need not to be scanned, and cache the transactions that are frequently scanned, the algorithm manage to decrease the data scanning times to save the time of database I/O operation. In the experiment, the real repairing and maintaining data of a main board company is mined and analyzed using our proposed algorithm. The results provide the company with many valuable information and rules to improve their product, and also verify the effectiveness of the proposed algorithm.

ACKNOWLEDGMENT

Supported by NSFC(National Natural Science Foundation of China): 70971032

REFERENCE

[1] Wanjun Yu, Xiaochun Wang, Fangyi Wang.The research of improved Apriori algorithm for mining association rules. In Proceedings of the 11th IEEE International Conference on Communication Technology Proceedings, pp 513-516,2008.

[2] Yuan Jian, Wang Wenhai.An Improved Apriori Algorithm for Mining Association Rules.2008,29(5):448-451.

[3] Lu SF, Lu ZD. Fast Mining maximum frequent itemsets. Journal of Software, 2001, 12(2):293-297.

[4] Jiawei Han, Yongjian Fu. Discovery of multiple-level association rules from large databases. In Proceedings of the 21st International Conference on Very Large Data Bases, 1995: 420-431.

[5] Ng. Raymond, V.S. Laks , J. Han, Mah. Teresa. Exploratory mining via constrained frequent set queries. In Proceedings of the ACM SIGMOD International Conference on Management of Data. 1999: 556-558.

TABLE I. STRUCTURE OF THE INDEX_TABLE

TID	Flag	Cached-address
T400	3	0x0001
T900	3	0x0004
T200	0	NULL
T300	0	NULL
T500	0	NULL
T600	0	NULL
T700	0	NULL
…	…	…

TABLE II. REPAIRING RECORD ATTRIBUTES

Field ID	Field Name	Data Structure	Data Description
1	rID	double[9]	Reparing ID
2	WorkerID	Char[10]	Reparing Person ID
3	SN	Char[20]	Main board SN
4	agentID	Char[5]	Agency ID
5	serviceDate	Char[8]	Reparing Date
6	MB_Type	Char[10]	Main board type
7	rStatus	Bool	Reparing Status
8	TBF	Int	Time of no fault
9	class1	Char[10]	Fault accessory 1
10	classNM1	Char[20]	Fault accessory name
	…	…	…
23	spotID	Char[10]	Reparing site ID
24	opID	Int	Operator ID
25	opDate	Char[8]	Operation Date
26	remark	Char[50]	Remarks

TABLE. III. CORRESPONDENCE BETWEEN ORIGINAL AND THE TRANSFORMED DATABASE

Attribute Code	Attribute Name	Item Code
	Region	
A1	South China	1
…	…	…
A6	East China	6
	Season	
S1	Spring	7
…	**…**	…
S4	Winter	10
	Duration without fault	
L1	<0.5 Years	11
…	…	…
L7	>3 years	17
	Fault Accessory	
C1	CPU	18
C2	IC	19
C3	BIOS	20
C4	Capacitances	21
…	…	…
	Repairing Status	
ST1	Repaired	41
ST2	Cannot be Repaired	42
	Main board Type	
T1	Type1	43
T2	Type2	44

TABLE. IV. THE TRANSACTION DATABASE

TID	Items	Transaction Length
1	1,8,12,20, 41,55	6
2	4,7,11,19, 20,41,80	7
3	3,10,17,19, 21,22,44	7
……	……	……

TABLE. V. PART OF RULES OF MINING RESULTS

ID	Results	Item size	Support(%)	Confidence(%)
1	North China => Winter	2	16.20	63.83
2	North China => Spring	2	14.82	60.37
3	West North => Winter	2	18.95	62.45
4	West North => Spring	2	17.27	58.11
5	South China => Summer	2	14.62	57.25
6	Type23 => Winter	2	13.73	55.53
7	Type2 => Summer	2	20.06	63.63
…	...	…	…	…