Research on Frequent Itemsets Mining Algorithm based on Relational Database

Jingyang Wang
Hebei University of Science and Technology, Shijiazhuang, China
Email: ever211@163.com

Huiyong Wang, Dongwen Zhang, Wanzhen Zhou
Hebei University of Science and Technology, Shijiazhuang, China
Email: {wanghuiyong815, zdwwtx, houwz} @hebust.edu.cn

Pengpeng Zhang
Hebei University of Science and Technology, Shijiazhuang, China
Email: 15131119925@163.com

***Abstract*—Mining association rules between items is an important research direction of data mining, and the relational database is the most popular database, so mining association rules in the relational database is a very important research direction. At present, neither the Apriori algorithm nor its improvements resolve some problems generating candidate itemset and scanning the transaction set repeatedly, which lead to low efficiency. This paper proposes the frequent itemsets mining algorithm based on relational database based on the study of those important mining association rules algorithms and the storage characteristics of the transaction set and items in the relational database, and presents its concrete implementation and its optimization method. This algorithm combines items in a transaction to generate itemsets and counts the same itemsets in all transactions, which improve the efficiency of execution. Moreover, this algorithm doesn't produce candidate itemsets, and only scans transaction database once, so promotes considerably efficiency. The result of experiments shows that, the frequent itemsets mining algorithm based on relational database has higher efficiency than the classical Apriori algorithm under certain conditions.**

***Index Terms*—relational database, frequent itemsets, association rule, Apriori**

I. INTRODUCTION

Data mining technology's economic value has been recognized from its birth to now, and many companies have started to use the data mining technology to create economic value [1]. Initial mining knowledge is just for Boolean association rules in transaction database, and however, the mass data is stored in the relational database nowadays, and how to use the relational database and mine knowledge from relational databases has become a meaningful reality and theoretical issues.

One of the most important data mining techniques is the association rule mining algorithm, and usually we think that mining association rules is a two-step process. The first step is to find all frequent itemsets, and the second is to generate strong association rules from frequent itemsets [2]. In this two-step, the second step is easiest, and the overall performance of mining association rules is determined by the first step, so the research of mining frequent itemsets is of great significance, and therefore, improved mining performance algorithms focused on how to quickly and efficiently mine frequent itemsets[3].

Currently, the research of mining association rules algorithms gets rapid development, and kinds of improved algorithms are put forward, among which includes improved algorithms based on classical algorithms, like the frequent pattern mining based on hash table[4], the reverse Apriori frequent pattern mining algorithms[5], the enhanced scaling Apriori[6], the frequent pattern mining based on sampling[7-10], and so on, and new algorithms appear, like the distributed frequent itemset mining algorithms[11,12], the association rule mining method on OLAP cube[13, 14], the association rules mining algorithm with Co-evolution algorithm in high dimensional data[15], the frequent closed itemsets mining algorithms[16], the frequent mining algorithms in data streams[17, 18], and so on.

Among those algorithms, most of the mining algorithms for association rules in relational databases are taking Apriori algorithm as the core, and use related techniques of relational database to achieve and optimize Apriori algorithms and its improvement, for example, [19-20] use characteristics that those generated candidate itemsets actually only need to consider a part of items in transactions, to improve Apriori algorithm. This paper uses the storage characteristics of transactions in relational database to put forward the frequent itemsets mining algorithm based on relational database.

doi:10.4304/jsw.8.8.1843-1850

II. FREQUENT ITEMSETS AND ASSOCIATION RULES

Finding frequent itemsets can be formally stated as follows: let I={I1,I2,I3,…,In} to be a set of distinct of literals, called items. Let D to be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$. Each transaction has an identifier, called TID. Let A to be a set of items, and transaction T contains A if and only if $A \subseteq T$. Association rules is an implication shaped like A =>B where it must ensures that $A \subset I$, $A \subset I$ and $A \mathrm{I}\ B = \varnothing$ are correct. Rules A =>B is established in transaction D, and its support is C, which is the percent of those transactions which contain A or B, and its value equals P(A∪B). And its confidence is c, which is the percent of those transactions which contain A and B, and its value equals to conditional probability P(A|B).And their computational formulas are showed below.

$$support(A \Rightarrow B) = P(A \cup B) \quad (1\text{-}1)$$

$$confidence(A \Rightarrow B) = P(B|A) \quad (1\text{-}2)$$

From the formula (1-2), it has the formula (1-3):

$$confidence(A \Rightarrow B) = P(B|A) = \frac{support(A \cup B)}{support(A)} = \frac{support\_count(A \cup B)}{support\_count(A)} \quad (1\text{-}3)$$

Using the formula (1-3), it shows that the supports C and confidence c of the rule A =>B can be easily calculated from the count of A and A∪B, that is to say, as long as the count of A, B and A∪B can be got, and the rules A =>B and B =>A can be easily calculated.

Among them, the rules meeting the minimum support threshold and the minimum confidence threshold are called strong rules. The core of mining association rules is to find all frequent rules, which are the strong rules.

III. ANALYSIS OF APRIORI ALGORITHM

Apriori uses a technique of layer-by-layer search called iterative method that uses k-itemsets to explore (k+1)-itemsets. Firstly, Apriori gets the count of each item by scanning the database, and then eliminates the items that don't meet the minimum support to get the frequent 1-itemsets collection, which is called collection L_1. Then Apriori uses L_1 to explore the frequent 2-itemsets collection L_2, and uses L_2 to explore L_3, and so on, until Apriori can no longer find frequently k-itemsets. It needs a transaction set scan to get the frequent k-itemsets collection L_k.

As can be seen from the top, the main time overhead of Apriori algorithm is in the scanning transactions stage, especially when the scan transaction set is stored in a relational database. When transaction set is stored in relational database, Apriori must first get out each transaction from the relational database, and then one by one to compare. Each itemset generated through connecting between items, Regardless of whether or not to meet the minimum support threshold of candidate sets and does not meet the minimum support threshold of candidate itemsets, must scan the entire transaction set to get the number of itemsets, which results in an unnecessary waste of resources. Among them, the number of occurrences of some candidate set is very small, but also has to scan the entire transaction set, which have a huge impact on the algorithm of space and time efficiency. Therefore, the improved Apriori algorithms mainly optimize the statistical of number of itemsets, such as hash-based technology [4], the transaction compression [21], sampling [8], and so on.

In addition, the pruning step stage of generating (k+1)-itemsets from the k-itemsets connection will produce a relatively large space and time overhead. As we all know, the property Apriori mainly use: all nonempty subset that the frequent itemsets contain must be frequent. In order to find the collection of candidate (k+1)-itemsets, Apriori algorithm must connect to itself to generate (k+1)-itemsets, however, the number of (k+1)-itemsets may contain infrequent itemsets, therefore, Apriori must scan the collection of the (k+1)-itemsets generated and the collection of the k-itemsets that doesn't meet the minimum support threshold of itemsets to remove (k+1)-itemsets that does not meet the minimum support threshold, which can lead to a great deal of time and space overhead, especially when the item number is large.

In order to effectively avoid and overcome the shortcomings and deficiencies of Apriori, this paper puts forward a frequent itemsets mining algorithm based on relational database through combining the characteristics that the transaction set is stored in relational databases.

IV. FREQUENT ITEMSETS MINING ALGORITHM BASED ON RELATIONAL DATABASE

The frequent itemsets mining algorithm based on relational database fully utilizes the storage characteristics of the transaction set in a relational database to maximize efficiency of mining frequent itemsets. The database structure that the algorithm applies to is shown in Fig.1. In the figure D_TID is the primary key of table D, which is used to uniquely identifies a transaction; D_I_ID is the foreign key of table D, which references the primary key I_ID of table I; D_OTHERS indicates the other information of transactions; I_ID is the primary key of table I, which is used to uniquely identifies a item; I_OTHERS indicates the other information of items; I_D_FOREIGNKEY indicates that the table D and I are joined by the foreign key. The relational database structure is the mainstream database design, especially in the retail sector, which ensures the broad application of the algorithm.

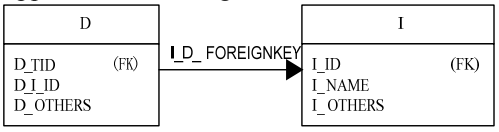

Figure 1. The database structure that the algorithm applies to

The idea of the frequent itemsets mining algorithm based on relational database is very simple and direct, that is, in turn gets out each transaction from relational databases, and then generates all possible itemsets, and finally counts the number of different itemsets through appropriate methods. The idea is very simple, but it is very critical to minimize the difficulty of implementation of algorithm and improve the efficiency of the algorithm, which are the key problems solved in this paper. The

implementation of the algorithm consists of two steps: one is to generate itemsets according to the items of transactions and the other is to count the number of each itemset. Next, this two-step implementation and optimization is described in detail in the following sections.

A. Generation of Itemsets

According to the permutation and combination theory of probability theory, as long as the number of A，B and A∪B can be got, the support and confidence information of the association rules A=>B and B=>A can be deduced. Therefore, the appropriate algorithm can be chosen to halve itemsets generated by a transaction, and meanwhile can guarantee to generate all possible itemsets in order to improve the efficiency of the algorithm.

The frequent itemsets mining algorithm based on relational database uses an iterative method to generate itemsets, in which (k+1)-itemsets is generated from k-itemsets. The implementation is as follow:

Step1: To ensure that the order of items remain relatively consistent between each other in all transactions. For example, I_1, I_2 is two items consisting of transactions, and I_1 is always ranked after or before I_2 in all transactions.

Step2: When 1-itemset collection is used to generate 2-itemset collection in a transaction, items in the transaction are in turn got out and then connect with items located after them in the transactions, and the last item doesn't connect with other items, and the 2-itemset collection is saved for the generation of the 3-itemset collection.

Step3: When the 2-itemset collections saved in Step2 is used to generate 3-itemset collection, every 2-itemset saved is get out and then connects with items following the item last connected in the 2-itemset in the transaction to generate 3-itemset collection and then save. The same method can be used to generate (k+1)-itemset collection from k-itemset collection.

The above steps are completed in a transaction, and every transaction is through the same steps. Then the algorithm uses appropriate method to count the number of itemsets generated by all transactions. In this way, the algorithm can't only significantly reduce the number of itemsets generated by a transaction, and is convenient for counting the number of itemsets. In the relational database, implementing the ordering of items in all transactions is very simple and just need a few simple relational data manipulation commands. After generating all itemsets, an appropriate method should be chosen to count the number of itemsets, which have a great impact on the efficiency of the algorithm. In below section, methods are detailed.

B. Counting the Number of Itemsets

Because the order of items in all transactions is relatively consistent, and the order of items in the itemsets generated is save in order, so it can determined where itemsets are in the data structure of counting the number of itemsets only through comparing the items of itemsets in order, and the number of itemsets can easily be counted, which greatly facilitate the selection of data structure.

Counting the itemsets is the main optimization stage of this algorithm, and its optimization methods are selecting the appropriate data structure to reduce the space overhead and improve efficiency. When the number of items consisting of transactions is relatively small, the linked list data structure is enough, in which each node represents an itemset. And if we want faster speed, we can select the combination of the linked list and the hash table, which can achieve the optimal retrieval time $O(1)$. When the number items is a relatively large, and the execution time of the algorithm is relatively long, and the memory space is insufficient, we can build the multilevel complete hash table, in which the same items at the beginning of all itemsets generated by all transactions are only saved once in the data structure. As shown in Fig.2, this is a two-level complete hash table, in which the first level stores the first item of all itemsets, and the second level stores the second item of all itemsets, and the multiple-level complete hash table structures is similar to it. This structure saves a portion of memory overhead, and its execution speed doesn't decrease a lot. Despite this, when the number of items is large, the memory overhead is difficult to unacceptable. The space complexity of the algorithm can be calculated accurately, and its value is $O(kn^2)$, in which k represent the maximum of items of itemsets, and n represents the number of items consisting of all transactions. It can be learned from $O(kn^2)$ that the space memory overhead is only about the number of items and the maximum number contained by transactions.

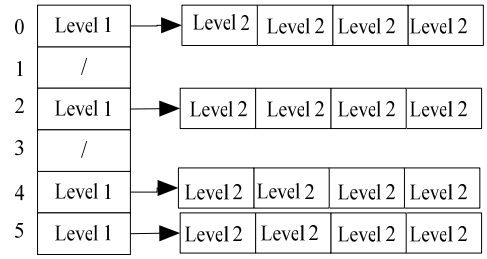

Figure 2. The two-level complete hash table structure

When the number of items is very large, the memory overhead is out of memory processing ability. And under these circumstances, Apriori algorithm's time overhead is difficult to unacceptable. In this case, the algorithm can use the function of the relational database to count the number of itemsets which can make up for the shortage of memory, and itemsets generated by transactions can be stored in a table with a proper format. Now, the processing speed of relational database management system is very faster, and its execution time won't reduce a lot.

Moreover, the execution efficiency also can be improved with the following properties:

(1) Property 1: All non-empty sub itemsets that frequent itemsets contain must also be frequent.

(2) Property 2: If there are frequent k-itemsets, the proportion of the number of transactions, the number of items who contains is equal to or greater than k, to the

total number of transactions is equal to or greater than the minimum support threshold.

The property 1, which is used by the Apriori algorithm, is used to remove these items that don't meet the minimum support threshold. And the property 2, which is obvious, is used to give a rough estimate of the maximum number of items itemsets contained, which is frequent. Combining the properties 1 and 2 can minimize the useless operation, and further improve the efficiency of execution of the algorithm.

In addition, the algorithm has strong applicability and only does slight modifications in the data preparation phase to meet the needs of most of the associated analysis, such as the association rules analysis between layers, association mining between categories, and the correlation coefficient between the commodities in the retail industry.

V. EXPERIMENTS

A. Experiment Results

In this experiment, we compare the frequent itemsets mining algorithm based on relational database to Apriori algorithm, which is the most popular algorithm. In order to test and verify the performance of the frequent itemsets mining based on relational database, with the same data sample and the same environment, we compare the execution time of two algorithms, as well as the characteristics of them.

Meanwhile, in order to better illustrate the stability and scalability of the algorithm, the experiment was divided into three parts: in the first part of the experiment, we compare the execution time of both algorithms with the change of the number of items consisting of all transactions; in the second part of the experiment, we compare the execution time of both algorithms with the change of the number of the total transactions; in the third part of the experiment, we compare the execution time of both algorithms with the increase of the minimum support threshold. And this can give us more comprehensive understanding of the efficiency and the stability of both algorithms. Each experiment is divided into fifteen times, and each experiment gradually increases the number of transactions or the items or the minimum support threshold, and the data structure of counting the number of itemsets in the frequent itemsets mining algorithm based on relational database is the multi-level complete hash table, and we use the Property 1 and 2 to optimize the execution speed implementing the frequent itemsets mining algorithm based on relational database. The following is the detail process of experiment.

1) Experiment 1

This experiment compares the execution time of Apriori algorithm and the frequent itemsets mining algorithm based on relational database when the number of items consisting of all transactions changes. This experiment only changes the number of items consisting of all transactions and doesn't change the number of all transactions and the minimum support threshold is set to 1% in terms of the actual situation of transaction set. The number of transactions is a fixed value, and its value is 100000, and data information of experiment 1 is displayed in Table Ⅰ. The result of comparison is illustrated in Fig.3.

TABLE I.

THE DATA INFORMATION OF EXPERIMENT 1

No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Items number (10^3)	0.07	0.1	0.15	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.5	2.0	2.5
Item number left	995	995	995	995	995	995	995	995	995	995	995	995	995	995	995
Mean number of items	6.5	6.4	6.4	6.4	6.4	6.4	6.4	6.4	6.4	6.5	6.4	6.4	6.4	6.4	6.4

TABLE II.

THE DATA INFORMATION OF EXPERIMENT 2

No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Trans num(10^4)	0.1	0.2	0.3	0.4	0.5	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0
Item number left	995	995	995	995	995	995	995	995	995	995	995	995	995	995	995
Mean number of items	6.5	6.4	6.4	6.4	6.3	6.4	6.4	6.4	6.4	6.4	6.3	6.3	6.3	6.3	6.3

TABLE III.

THE DATA INFORMATION OF EXPERIMENT 3

No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
min_sup(%)	1	2	3	4	5	6	7	8	9	10	15	20	25	30	35
Item number left	995	985	979	968	951	943	931	919	906	899	843	795	742	690	602
Mean number of items	6.40	6.39	6.38	6.38	6.37	6.37	6.35	6.30	6.25	6.20	6.02	5.77	5.56	5.01	4.50

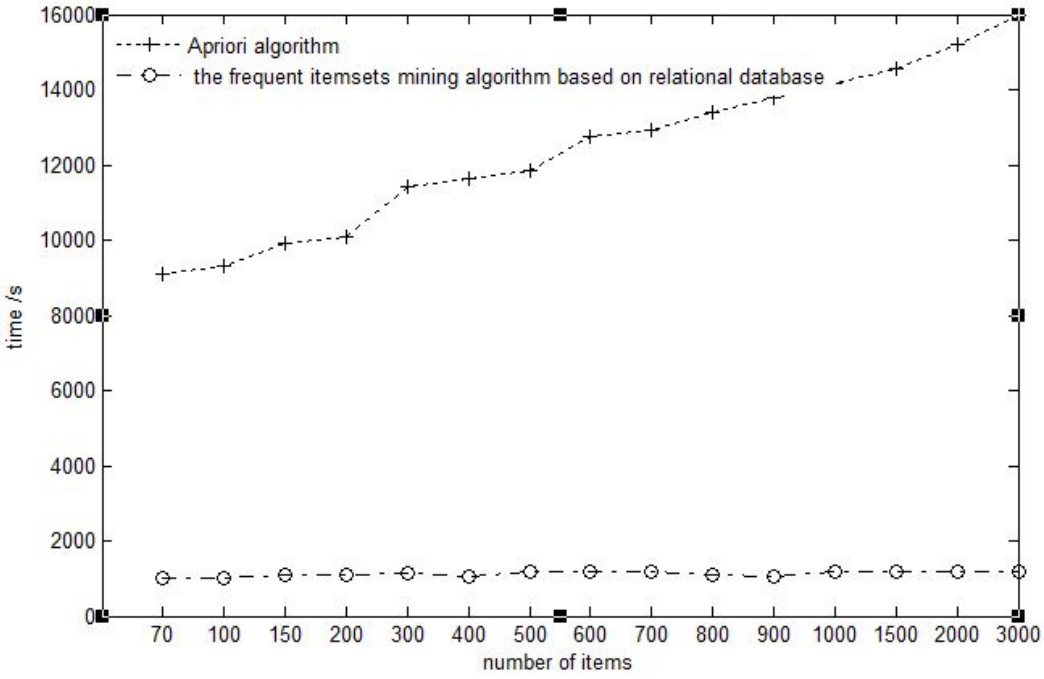

Figure 3. The result of experiment 1

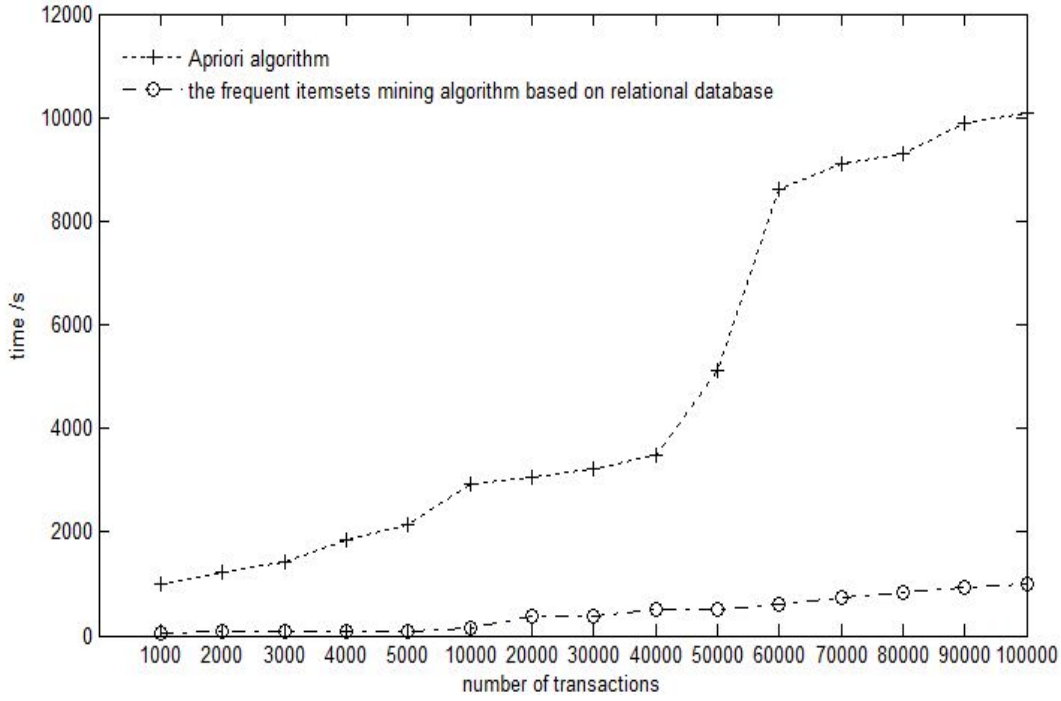

Figure 4. The result of experiment 2

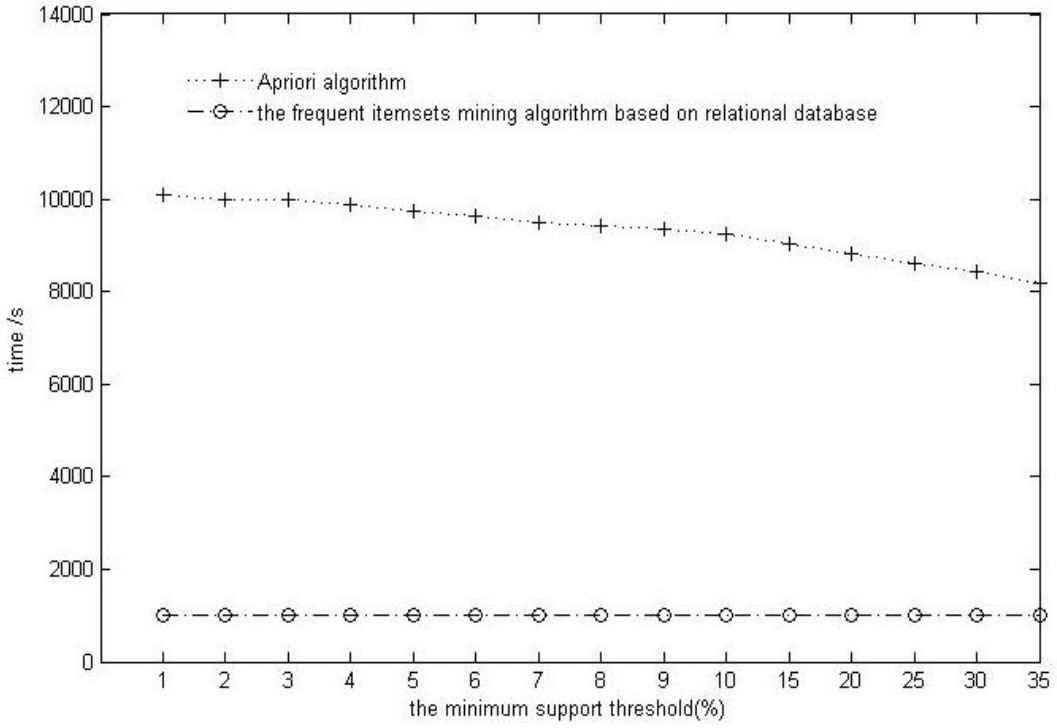

Figure 5. The result of experiment 2

2) Experiment 2

This experiment compares the execution time of Apriori algorithm and the frequent itemsets mining algorithm based on relational database when the number of transactions in the transaction set changes. This experiment only changes the number of all transactions and doesn't change the number of items consisting of all transactions, and the minimum support threshold is set to 1% in terms of the actual situation of transaction set. The number of items consisting of all transactions is a fixed value, and its value is 1000, and data information of experiment 2 is displayed in Table Ⅱ. The result of comparison is illustrated in Fig.4.

3) Experiment 3

This experiment compares the execution time of Apriori algorithm and the frequent itemsets mining algorithm based on relational database when the minimum support threshold changes. This experiment only changes the minimum support threshold (min_sup) and doesn't change the number of items and the number of transactions in transaction set. The number of transactions in the transaction set is a fixed value, and its value is 100000, and the number of items consisting of the transaction set is 1000, and data information of experiment 3 is displayed in Table Ⅲ. The result of comparison is illustrated in Fig.5.

B. Results Discussion

From the results of the experiment 1 and 3 displayed in Fig.3 and Fig.4, we can see that, with the increase of the number of the items, the execution time of the frequent itemsets mining algorithm based on relational database has almost no change, and however, the execution time of Apriori algorithm increases rapidly. However, from the results of the experiment 2 displayed in Fig.4, we can see that, with the increase of the number of transactions, the execution time of the frequent itemsets mining algorithm increases smoothly, and however, the execution time of Apriori algorithm have increases rapidly. In a word, it can be concluded that the frequent itemsets mining algorithm based on relational database is more stable and efficient than Apriori algorithm.

But, the performance of the algorithm is not only reflected in the running time, and is also reflected in the size of the memory requirements. The memory requirements of Apriori algorithm is relatively small, and however, the memory requirements of the frequent itemsets mining algorithm based on relational database is very large, which is the biggest drawback. From the results of the experiment 1 and 3, we can see that the execution speed of the frequent itemsets mining algorithm based on relational database is only related to the number of transactions, but its memory requirements increase dramatically with the increase of the number of items, which limits its scope of application. So, its memory requirements must be reduced in order to broaden its scope of application.

One method is that the data structure of counting the number of itemsets is set to be the multiple-level complete hash table structures, which is showed in Fig.2, and its key is the primary key of the item table. In order to minimize the memory requirements, the primary key is set to be consecutive integers after the order the items in the transaction set is sorted, and its space complexity is O(kn2), in which k represents the maximum of items of itemsets, and n represents the number of items consisting of all transactions. But the execution speed of

the frequent itemsets mining algorithm based on relational database is easier to implement and much faster than ones of Apriori algorithm, and so if the number of items is not too much and the memory is enough, the frequent itemsets mining algorithm based on relational database is suitable.

In the process of generating itemsets, if the number of items a transaction contained is n, the number of connecting operation and itemsets generated is 2n-n. And if the number of itemsets is m and the data structure is the complete hash table, the connecting operation is m times and the comparing operation is m*k times. However, Apriori algorithm needs to scan all transactions, and removes the candidate itemsets, which has a bad affect on the efficiency.

It can be seen from the above discussion that the frequent itemsets mining algorithm based on relational database is applicable to the case when the number of items is not very large and the number of transactions is very large.

VI. CONCLUSIONS

This paper puts forward the frequent itemsets mining algorithm based on relational database in order to effectively avoid the shortcomings and deficiencies of Apriori and take full advantage of the characteristics that the transaction set is stored in a relational database. This algorithm only needs a scan again transactions, and does not generate candidate itemsets. The iteration within a transaction produces itemsets, and it does not repeat the scan transaction set and maximizes efficiency.

This paper compares the frequent itemsets mining algorithm based on relational database to Apriori algorithm under the same conditions and the results show that the frequent itemsets mining algorithm based on relational database is more efficient and better stable than Apriori algorithm.

Although the paper optimizes the memory overhead of the algorithm, the memory overhead is very large when the number of items is large. So, the next step is to further optimize the memory overhead.

ACKNOWLEDGMENT

The authors wish to thank the editor and referees for their careful review and valuable critical comments. This work is supported by the following funds: the Science Fund of Hebei of China No. 2011228 and the Initial fund for doctors of Hebei University of Science and Technology No.QD201223.

REFERENCES

[1] Endu Duneja R.I.T.S, Bhopal M.P., A.K. Sachan R.I.T.S., "A Survey on Frequent Itemset Mining with Association Rules," *International Journal of Computer Applications*, 46(23),pp.18-24,2012.

[2] AA Raorane, RV Kulkarni, BD Jitkar, "Association Rule–Extracting Knowledge Using Market Basket Analysis," *Research Journal of Recent Sciences,* vol.1, pp. 19-27,2012.

[3] Ziauddin, Shahid Kammal, Khaiuz Zaman Khan, Muhammad Ijaz Khan, "Research on Association Rule Mining," *Advances in Computational Mathematics and its Applications(ACMA)*,vol.2,pp.226-236,2012.

[4] Nazish Asad, M. Younus Javed, Usman Qamar, Memoona Javeria Anwar, "Association Rules Mining for Urdu Language Using Transaction Hash Tables based Apriori (THT-Apriori),"*Journal of Basic and Applied Scientific Research*,vol.2,pp.5908-5914,2012.

[5] Kamrul, Shah, Mohammad, Khandakar, Hasnain, Abu, "Reverse Apriori Algorithm for Frequent Pattern Mining,"*Asian Journal of Information Technology*, pp.524-530,2008.

[6] S. Praksh, R.M.S. Parvathi, "An enhanced Scalling Apriori for Association Rule Mining Efficiency," *European Journal of Scientific Research*, vol.39, pp.257-264,2008.

[7] Riondato, Matteo, and Eli Upfal. "Efficient discovery of association rules and frequent itemsets through sampling with tight performance guarantees." *Machine Learning and Knowledge Discovery in Databases*, vol.1, pp.25-41, 2012

[8] V. T. Chakaravarthy, V. Pandit and Y. Sabharwal, "Analysis of sampling techniques for association rule mining," *in Proceedings of the 12th international conference on database theory*, pp. 276-283, 2009.

[9] Pietracaprina, Andrea, et al. "Mining top-K frequent itemsets through progressive sampling." *Data Mining and Knowledge Discovery*, vol.21, pp.310-326, 2010.

[10] Bagheri, M., Mirian-Hosseinabadi, S. H., Mashayekhi, H., & Habibi, J., "Mining Distributed Frequent Itemsets Using a Gossip Based Protocol," *In Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC)*, vol.9,pp. 780-785,2012.

[11] Verma, Harish, Durga Toshniwal, and Sateesh Kumar Peddoju, "Distributed frequent itemset mining framework for incremental data using MPI-style WSRF services,". *Proceedings of the International Conference on Advances in Computing, Communications and Informatics. ACM*, pp.74-81, 2012.

[12] Jadav J J, Panchal M, "Association rule mining method on OLAP cube," *International Journal of Engineering Research and Applications (IJERA)*,vol.2,pp.1147-1151,2012.

[13] Lee, C. K. H., Choy, K. L., Ho, G. T. S., Chin, K. S., Law, K. M. Y., & Tse, Y. K.,"A hybrid OLAP-association rule mining based quality management system for extracting defect patterns in the garment industry," *Expert Systems with Applications*, vol.40,pp.2435-2446,2013.

[14] Lou, Wei, Lei Zhu, and Limin Yan, "The Research on Association Rules Mining with Co-evolution Algorithm in High Dimensional Data," *AsiaSim 2012* , pp.338-346,2012.

[15] Keming Tang, Caiyan Dai, Ling Chen, "An efficient mining algorithm by Bit Vector Table for frequent closed itemsets," *Journal of Software*, vol.6, pp.2121-2128, November 2011.

[16] Keming Tang, Caiyan Dai, Ling Chen, "A novel strategy for mining frequent closed item sets in data streams," *Journal of Computers*, vol.7, pp. 1564-1573, 2012.

[17] Tang, Keming, Caiyan Dai, and Ling Chen. "A Novel Strategy for Mining Frequent Closed Itemsets in Data Streams." *Journal of Computers* vol.7, pp.1564-1573, 2012.

[18] Li, Haifeng, Ning Zhang, and Zhixin Chen. "A Simple but Effective Maximal Frequent Itemset Mining Algorithm over Streams." *Journal of Software*, vol.7, pp.25-32, 2012.

[19] Danubianu, M., Pentiuc, S. G., & Tobolcea, I, "Mining Association Rules Inside a Relational Database–A Case Study," *In ICCGI 2011, The Sixth International Multi-Conference on Computing in the Global Information Technology*, pp.14-19,2011.
[20] Bart Goethals,Wim Le Page, Michael Mampaey, "Mining interesting sets and rules in relational databases," *Proceedings of the 2010 ACM Symposium on Applied Computing*,pp.997-1001,2010.
[21] Ren, Jiadong, Juan Yi, and Haitao He. "Frequent Itemset Mining Based on Bit-Sequence." *Advances in Information Technology and Industry Applications. Springer Berlin Heidelberg*, pp.597-603, 2012.
[22] Gupta, Ruchita, and C. S. Satsangi. "An Efficient Range Partitioning Method for Finding Frequent Patterns from Huge Database." *International Journal of Advanced Computer Research,* vol.2,num.2,pp. 62-69, 2012.
[23] Savasere A, Omiecinski E and Navathe S, "An Efficient Algorithm for Mining Association Rules in Large Databases, Very Large Data Bases," *In Proc. of the 21th International Conference on Very Large Data Bases*,pp.432-444,1995.
[24] M. J. Zaki, S. Parthasarathy, W. Li, and M. Ogihara, "Evaluation of sampling for data mining of association rules," i*n Proceedings. Seventh International Workshop on Research Issues in Data Engineering*, pp. 42-50, 1997.
[25] Jin, Ruoming, et al. "Estimating the number of frequent itemsets in a large database." *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*. ACM, 2009.
[26] Quan, Jiang, et al. "High-Efficiency Algorithm for Mining Maximal Frequent Item Sets Based on Matrix." *Computational Intelligence and Communication Networks (CICN), 2012 Fourth International Conference on*. IEEE, 2012.

Jingyang Wang, Associate Professor, born in 1971. He received the B.Eng. degree in computer software from Lanzhou University, China, in 1995. He received the M.Sc. degree in software engineering from Beijing University of Technology, China, in 2007. Now he is working in School of Information Science and Engineering, Hebei University of Science and Technology, Shijiazhuang Hebei, China. His main research areas include data mining, active network, transmission control, and distributed computing.

Huiyong Wang, Lecturer, born in 1980. He received the B.Eng. degree in Mechanical and Electronic Engineering from Hebei University of Technology, China, in 2002. He received the M.Sc. degree in Mechanical and Electronic Engineering from Hebei University of Technology, China, in 2005. Now he is working in School of Information Science and Engineering, Hebei University of Science and Technology, Shijiazhuang Hebei, China. His main research areas include data mining, swarm intelligence, and distributed computing.

Dongwen Zhang, Professor, born in 1964. She received the M.Sc. degree in Electrical Engineering from Harbin University of Technology, China, in 1990. Her main research interests include network and communication, system modeling and automatic control.

Wanzhen Zhou, Professor, born in 1966. He received the B.Eng. degree in applied mathematics from Harbin University of Technology, China, in 1988. He received the M.Sc. degree in computer science from Harbin University of Technology, China, in 1992. His main research interests include network and database, system modeling and image processing.

Pengpeng Zhang, Student, born in 1987. He will receive the B.Eng. degree in computer software from Hebei University of Science and Technology, China, in 2013. His main research interests include data mining and data warehouse.