Chapter 9

Combining Some of The Improvements of Apriori Algorithm

F. A. El-Mouadib, Salem A. Mohammed
Computer Department, Faculty of Science, Garyounis University, Benghazi,Libya

ABSTRACT

Data Mining (DM) is an important issue in the field of data and knowledge based systems, which have been prompted by an interesting new field called Knowledge Discovery in Databases (KDD). Discovered knowledge can come in many forms such as: association rules, correlations, sequences, episodes, classifiers, clusters and many more. Mining for association rules had received great attention in recent years. The original motivation for searching association rules came from the need to analyze the so-called supermarket transaction data, that is, to examine customer behavior in terms of the purchased products. Many algorithms have been developed for association rule mining. The algorithm APRIORI is one of the earliest and most well known algorithms for association rule mining. Some improvements have been carried out on the apriori algorithm.

In this paper, we will investigate some of the APRIORI algorithm improvements and combine two of them namely; Sampling and Partitioning into one algorithm that we call APRIORI-SP. We also will demonstrate the implementations of the APRIORI-SP via number of experiments that were conducted to measure the relative performance of the APRIORI-SP algorithm compared with the Sampling and Partitioning algorithms separately.

Key Words: Data Mining (DM); Association rule mining; Knowledge Discovery in Databases (KDD);APRIORI.

INTRODUCTION

The increased capacities and decreased costs of storage media, has led businesses to store huge amounts of external and internal information in large databases at low cost. Mining useful information and helpful knowledge from these large databases has thus evolved into an important research area (Chen , Han, Yu ,1996).Association rule mining has been one of the most popular data-mining subjects, which can be simply defined as finding interesting rules from large collections of data. Association rule mining has a wide range of applicability such Market basket analysis, Medical diagnosis/research, Website navigation analysis, Homeland security and so on. Association rules are used to identify relationships among a set of items in database. These relationships are not based on inherent properties of the data themselves (as with functional dependencies), but rather based on co-occurrence of the data items.

The idea of association rules were first introduced in (Agrawal, Imielinski, Swami, 1993). And subsequent research has made it one of the most interesting ideas by the introduction of the very-well known algorithm Apriori. The Apriori algorithm has not influenced only the association rule mining community, but it affected other data mining functionalities as well. Since the introduction of Apriori algorithm in (Agrawal, Srikant,1994). Number of improvements has been carried out on the original algorithm in order to elevate its efficiency.

Many algorithms have adapted Apriori as a basic search strategy, tended to adapt the whole set of procedures and data structures as well (Park, Chen, & Yu, 1995; Brin, Motwani, Ullman, 1997; Toivonen, 1996). Since the scheme of the Apriori algorithm was not only used in basic association rules mining, but also in other data mining fields (hierarchical association rules (Sarasere, 1995. and Han, 1995). Association rules maintenance (Cheung, Han, Wong, 1996). Sequential pattern mining (Agrawal, Srikant 1995). episode mining (Toivonen,1995). And functional dependency discovery (Huhtala, Karkkainen, Pokka, Toivonen, 1999; Huhtala,). Frequent pattern mining techniques can also be extended to solve many other problems, such as classification (Liu, Hsu, Ma, 1998). The reset of the paper is organized as follows: in section (2) we give a formal definition of association rules and the problem definition. Section (3) introduces our development. Experimental results are shown in section (4) and section (5) demonstrates our conclusion.

ASSOCIATION RULE PROBLEM

The problem of finding association rules can be stated as follows: given a database of sales transactions, it is desirable to discover the important associations among different items such the presence of some items in a transaction will imply the presence of other items in the same transaction. As example of an association rule is:

Contains (T,"baby food") $\Rightarrow$ Contains (T, "diapers") [Support= 4%, Confidence=40%]

The interpretation of such rule is as follows:

- 40% of transactions that *contains* baby food also *contains* diapers;
- 4% of all transactions *contain* both of these items.

The calculations of the *Support(S)* and *Confidence(C)* are very simple:

$$Confidence(A \Rightarrow B) = \frac{Support\,(A \cup B)}{Support\,(A)}$$

$$Support(A) = \frac{Number\ of\ transactions\ containing\ item\ A}{Total\ number\ of\ transactions\ in\ the\ database}$$

$$Support(A \cup B) = \frac{Number\ of\ transactions\ containing\ items\ A\ and\ B}{Total\ number\ of\ transactions\ in\ the\ database}$$

The above association rule is called single-dimension because it involves a single attribute or predicate (*Contains*). The main problem is to find all association rules that satisfy minimum support (min_sup) and minimum confidence (min_conf) thresholds, which are provided by user and/or domain experts. Due to (Agrawal, Srikant,1994).A formal definition of association rule is:
Let J= {i1, i2…im} be a set of items. Let D be the set of database transaction where each transaction T is a set of items such that is $T \subseteq J$. Each transaction is associated with an identifier, called TID. Let A be a set of items. A transaction T is said to contain A if and only if $A \subseteq T$. An association rule is an implication of the form $A \Rightarrow B$, where $A \subset J$, $B \subset J$, and $A \cap B = \emptyset$. The rule $A \Rightarrow B$ holds in the transaction set D with support s, where s is the percentage of transactions in D that contain $A \cup B$ (i.e., both A and B). This is taken to be the probability, P ($A \cup B$). The rule $A \Rightarrow B$ has confidence c in the transaction set D if c is the percentage of transactions in D containing A that also contain B. This is taken to be the conditional probability, P (B|A). A rule is frequent if its support is greater than the minimum support threshold and strong if its confidence is more than the minimum confidence threshold.
Discovering all association rules is considered as two-phase process which are:

1. Find all frequent itemsets having minimum support. The search space to enumeration all frequent itemsets is on the magnitude of 2n.
2. Generate strong rules. Any association that satisfies the threshold will be used to generate an association rule.

The first phase in discovering all association rules is considered to be the most important one because it is time consuming due to the huge search space (the power set of the set of all items) and the second phase can be accomplished in a straightforward manner.
Algorithms in Association Rule Mining (ARM)

The Apriori Algorithm:

The APRIORI algorithm is one of the most well-known algorithms for association rule mining. As it has been mentioned in (Zaki, 1999). The Apriori algorithm by Rakesh Agrawal and colleagues has emerged as one of the best ARM algorithms. Its name is based on the fact that the algorithm uses priori knowledge of frequent itemset properties. The main difference between Apriori algorithm and its' predecessors (AIS and SETM algorithms) is that it employs a different candidate's generation method and a new pruning technique. Details of the APRIORI algorithm can be found in (Agrawal, Srikant,1994.& Han, 2001). Despite of the fact that the Apriori algorithm is one of the most noted algorithms in association rule mining and according to (Han, 2001), it has two main draw back which are:

1. The complex candidate generation process in the sense of time and computer memory consumption.
2. The multiple scans of the databases.

Since the emergence of the Apriori algorithm, some improvements have been carried out on it such as: Partitioning (Sarasere, Omiecinsky, Navathe, 1995). Direct Hashing and Pruning (DHP) (Park,1995). Sampling (Toivonen, 1996). Dynamic Itemset Counting (DIC) (Brin, 1997). And

some other algorithms. Here we will focus only on two of the above mentioned improvements namely Sampling and Partitioning.

The Sampling Improvement Algorithm:

The sampling technique algorithm was proposed by (Toivonen, 1996). The purpose of this technique is to reduce the number of database scans to one in the best case and two in the worst. As it has been mentioned in (Han, 2001). The general idea of the sampling technique algorithm is to pick a random sample or subset of the data to find all frequent itemsets in the chosen sample by using an algorithm such as Apriori with a lower threshold and use negative border function to ensure completeness. For more details on the aspect of negative border, see (Toivonen, 1996; Han, 2001).

The Partitioning Improvement Algorithm:

The partitioning technique was first proposed by (Sarasere et al, 1995). The partitioning technique reduces the number of database scans to two. It divides the database into *n* non-overlapping small partitions to be handled in the main memory. Hence, only one partition is read in the memory at one time and finds all the local frequent itemsets for the given partition, Li, by using local-minimum-support for that partition. This process is performed on of all the partitions and considered to be the first scan of the database. From all the local-minimum-support of all partitions a global-minimum-support is formed to be used for the second scan of the whole database. For more details on the partitioning algorithm, see (Sarasere, Omiecinsky, Navathe 1995).

Apriori-SP algorithm:

In this algorithm, we have combined some of the improvements that have suggested to the Apriori algorithm. These improvements are being Sampling and Partitioning and this is how the name came to be. The following depicts the methodology adopted by the Apriori-SP algorithm.

1. Pick random sample, S, from transaction database (D)
2. Partition the random sample (S) into subset (or partitions), so each partition size is suitable to the computer memory.
3. Read one partition in to memory.
4. Collect all the candidate 1-itemsets in the partition.
5. Count the number of occurrences of each 1-itemsets to compose the candidate set called C1.

   <C1: candidate 1-items>
6. Delete infrequent candidate 1-items (itemsets did not meet local-minimum-support requirement) to generate a list called L1.

< Local-minimum-support = (User threshold/ Number of partitions)>
<L1 is a list of frequent or large 1-itemset>

7. Joins L1 with itself to generate candidate 2-itemsets C2. C2 consists of all combinations ($\binom{L_1}{2}$) of 2-itemsets and the algorithm checks for the property that states 'all nonempty subsets of frequent itemsets must also be frequent' which is the bases for pruning infrequent itemsets.
8. Recursively perform steps 4 to 6 with the addition of an itemset every pass, until some Lk becomes empty which means that the join operation (step 6) is not possible.
9. Combine all frequent Ln-itemsets into one list called local frequent itemsets (LLp) for the given partition.

   <where n=1, 2, …, K>
10. Recursively perform steps 2 to 8 for each partition.
11. Combine all LLp to generate global candidate itemsets (LG).
12. Scan the whole transactional database to count the number of occurrences of each candidate K-itemset in LG.
13. Delete infrequent candidate K-items (itemsets did not meet global minimum-support threshold) to generate global frequent itemset in the database (all partitions).

EXPERIMENTAL EVALUATION

Here, we demonstrate the obtained results of our system APRIORI-SP using different types and sizes of data. The results are compared with the results of the Sampling and Partitioning algorithms separately. Our system was tested by the use of two synthetic datasets (databases) and two real datasets. The synthetic databases were generated using publicly available synthetic generation program of the IBM Quest data mining project [18] and detailed in (Agrawal, Srikant,1994). The used datasets are available on public domain [1], which have been used previously in association rules mining research.

Before reporting on our experimental results, let us briefly introduce the environment under which our experiments were conducted. All the experiments were performed on a Personal computer with the following specifications: 2500MHz Pentium IV with 512MB Memory, running on Microsoft Windows XP professional operating system.

Synthetic Data Experiments:

We have conducted two experiments for this type of data with different sizes. The chosen parameters for generating the data are the same as the ones that have been used previously in (Agrawal,1993., Agrawal, Srikant, 1994., Toivonen, 1996., Sarasere et al, 1995 & Park, 1995). The rational of choosing such parameters is to compare our results with the previous ones.

Small Synthetic Data Experiment:

The small synthetic data experiment has been conducted on a database that consists of 100,000 transactions, 1000 different items, the average number of items per transaction is 5, the average size of maximal large itemsets is 2 and the number of maximal large itemsets is 2000.

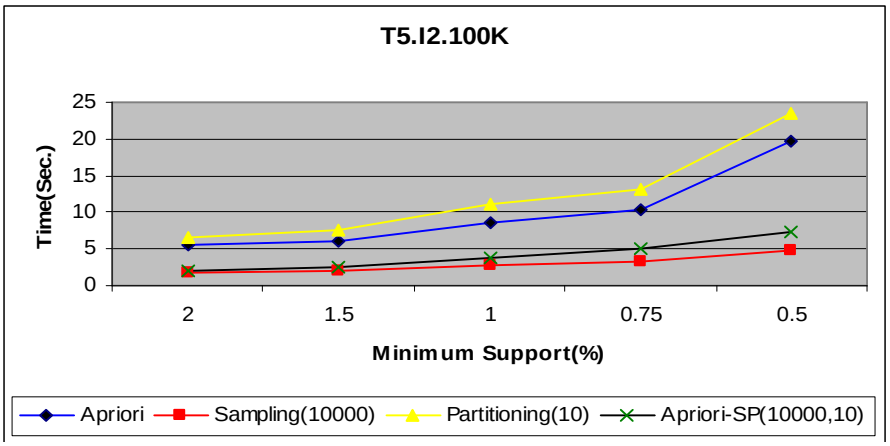

Fig. 1. Small synthetic data experiment (Minimum support vs. Time).

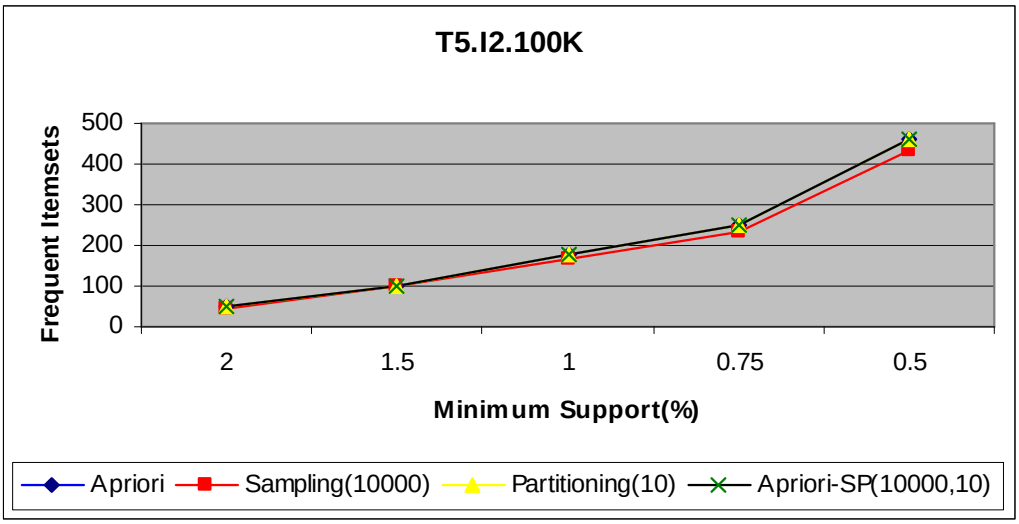

Fig. 2. Small synthetic data experiment (Minimum support vs. frequent itemsets).

The size of such database is about 6.41 MB. The experiment was carried out five, each with different minimum support (2.0, 1.5, 1.0, 0.75 and 0.5). The final results are depicted in Fig. 1 and Fig. 2. Fig.1 is a plot of minimum support vs. CPU time and Fig.2 is a plot of minimum support vs. frequent itemsets.

From Fig. 1, we can see that the sample size is 10000 transactions and the number of partitions is 10 whenever they are used. The interpretation of the results of Fig. 1 is: (1) Sampling and Apriori-SP gave a better result than Apriori and Partitioning and (2) when the minimum support is decreased, Sampling has shown a better result than Apriori-SP.

The results of Fig.2 can be interpretation as: (1) Partitioning, Apriori and Apriori-SP gave better results than Sampling and (2) Partitioning, Apriori and Apriori-SP gave exactly the same results.

Large Synthetic Data Experiment:

The large synthetic data experiment consists of 100,000 transactions with 1000 different items, the average number of items per transaction is 20, the average size of maximal large itemsets is 6 and the number of maximal large itemsets is 2000. The size of such database is about 19.7 MB.

This experiment was also conducted five times, each of the different minimum support (2.0, 1.5, 1.0, 0.75 and 0.5). The final results are depicted in Fig. 3 and Fig.4.

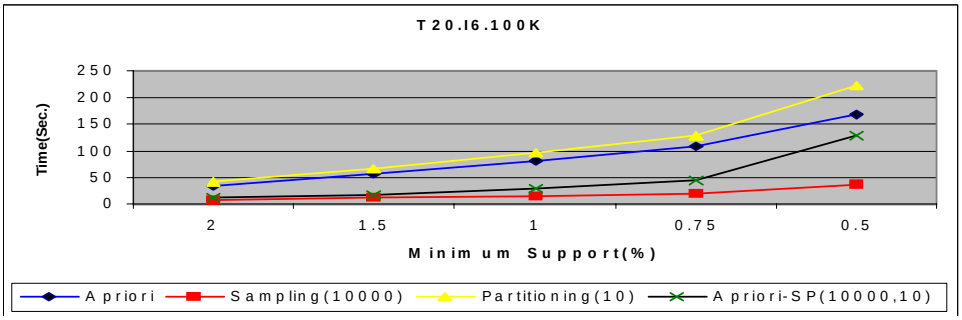

Fig. 3. Large synthetic data experiment (Minimum support vs. Time).

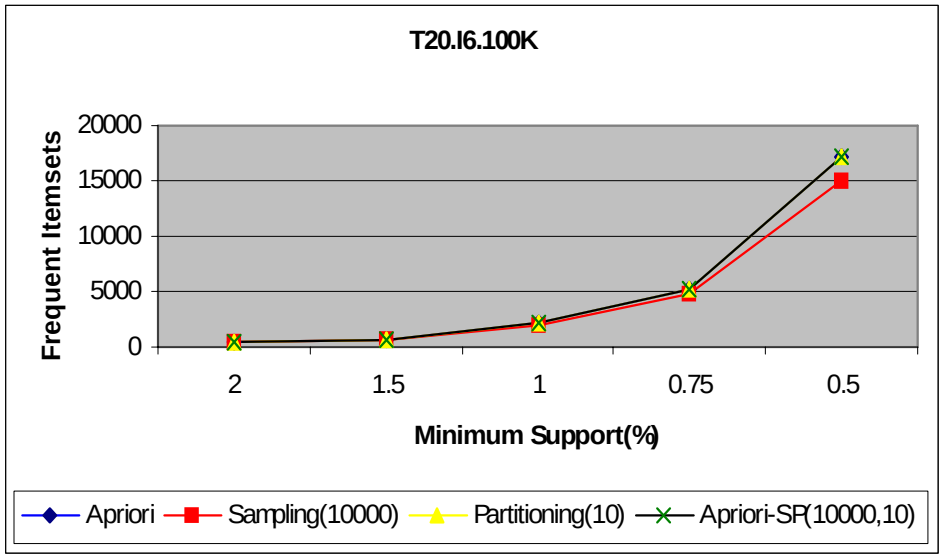

Fig. 4. Large synthetic data experiment (Minimum support vs. frequent itemsets.

Fig. 3 is a plot of minimum support and CPU time and Fig.4 is a plot of minimum support and frequent itemsets. The number of transactions and number of partitions are as explained before in the small synthetic data experiment.

In Fig. 3, it can be seen that: (1) Sampling and Apriori-SP gave better results than Apriori and Partitioning, (2) as the minimum support is decreased, Sampling has shown a better result than Apriori-SP and (3) the performance of Apriori-SP and Partitioning has worsened as the minimum support decreased.

From Fig. 4, we can see that: (1) Partitioning, Apriori and Apriori-SP gave better results than Sampling and (2) Partitioning, Apriori-SP and Apriori gave exactly the same results.

Real Data Experiments:

Here, we have conducted two experiments on real data. The first data came from an anonymous Belgian retail store. The second data is mushroom database that describes the characteristics of various mushroom species. These databases have been used in the association rule research community.

Retail Data Experiment:

The Retail database consists of 88162 transactions (records), the number of different items is 16469 and the average number of items per transaction is 10.3. The size of such database is about 11.1 MB.
As before, this experiment has been conducted five times on each of the four different implemented algorithms with different minimum support (2.5, 2, 1.5, 1 and 0.5).
The results of this experiment are depicted in fig.5 and fig.6.

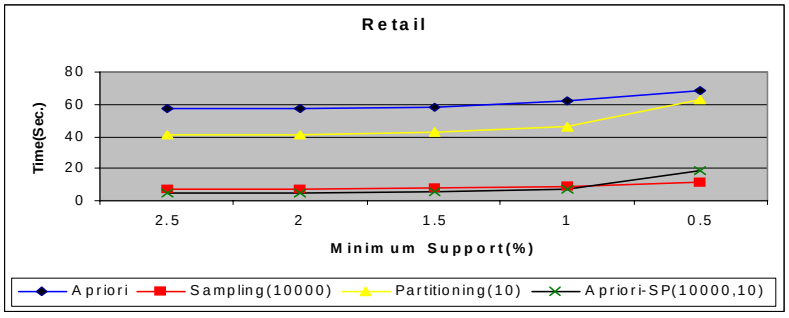

Fig. 5. Retail data experiment (Minimum support vs. Time).

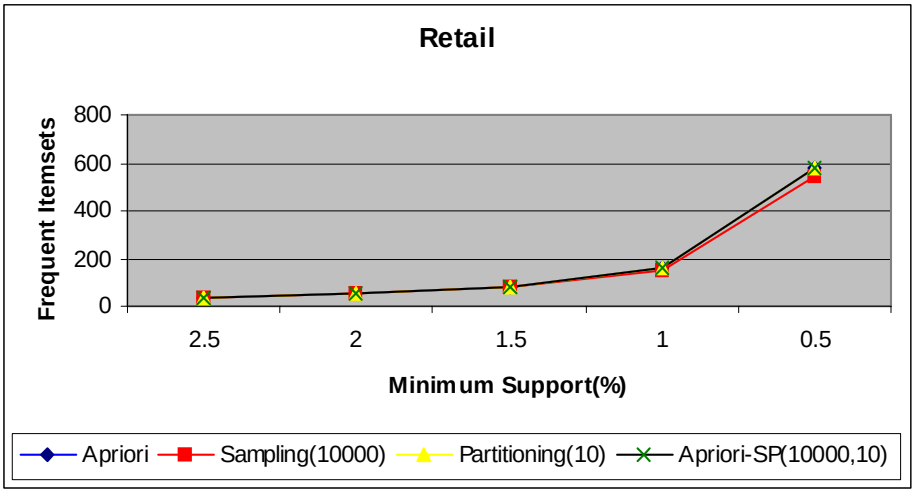

Fig. 6. Retail data experiment (Minimum support vs. frequent itemsets).

The results in Fig. 5 can be interpretation as: (1) Sampling and Apriori-SP gave a better result than Apriori and Partitioning as before, (2) Apriori-SP has gave better results than Sampling until minimum support of 1.0 and as the minimum support decreases, Sampling has shown a better result than Apriori-SP, and (3) Partitioning has better performance than Apriori. The interpretation of the results of Fig.e 6 can be seen as: (1) Partitioning, Apriori and Apriori-SP gave better results than Sampling and (2) Partitioning, Apriori and Apriori-SP gave exactly the same results.

Mushroom Data Experiment:

The Mushroom database consists of 8124 transactions with 119 different items and the average number of items per transaction is 23. The size of this database is about 1.59 MB.
This experiment was conducted five times, each with different minimum support (60, 55, 50, 45 and 40). The final results are depicted in Fig. 7 and Fig. 8. The sample size is 1000 transactions and the number of partitions is 2 whenever they are used.

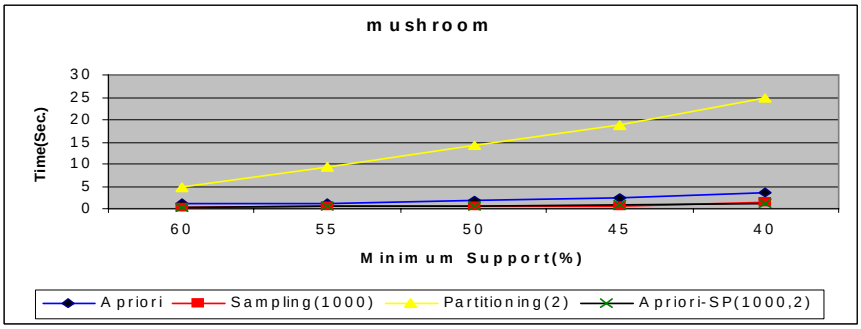

Fig.7. Mushroom data experiment (Minimum support vs. Time).

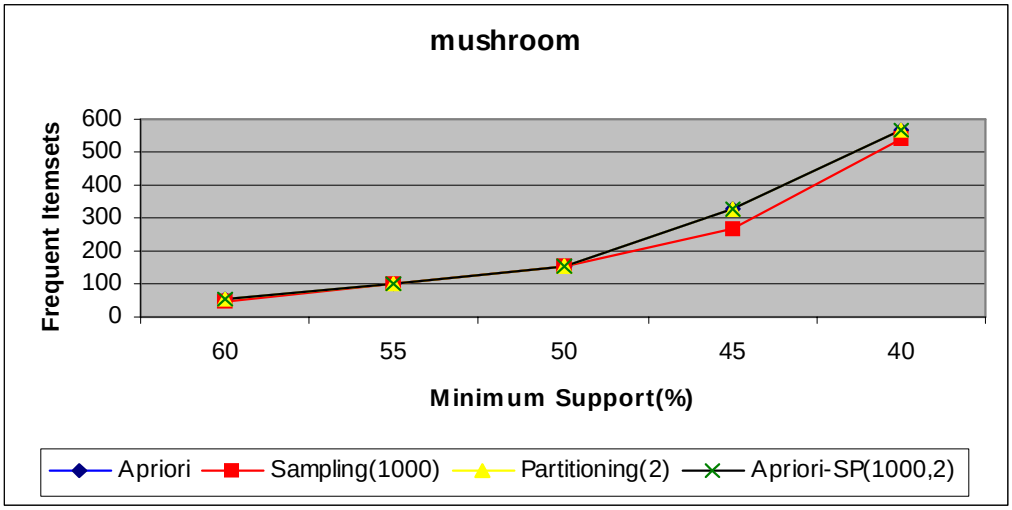

Fig. 8. Mushroom data experiment (Minimum support vs. frequent itemsets).

The interpretation of the results of Fig. 3 is that: Sampling, Apriori and Apriori-

CONCLUSION

The objectives of this study are to review the concept of ARM and to combine some of the improvements of the Apriori algorithm in a system that we call Apriori-SP. We have accomplished the objectives of the study by the implementation the Apriori algorithm, the Sampling and the Partitioning improvements algorithms. All of the above mentioned algorithms

have been analyzed and design by the use of UML. All the algorithms are implemented the Apriori-SP system by the use of VB.NET 2005 programming language.

We have conducted about 80 experiments to test the Apriori-SP system. The databases used in these experiments are real and synthetic data with different sizes ranging from 1.59 MB to 19.7 MB. From the experiments we have conducted, the authors noticed the followings:

1. The Apriori algorithm has shown low performance when the minimum support is low and the number of transactions and large number of items per transactions is high.
2. The Apriori algorithm has shown better performance when the minimum support is high.
3. The Sampling and Apriori-SP algorithms gave better results than Apriori due to the reduction of the number of scans (actually two scans) of the database.
4. In most of the experiments, the Sampling algorithm has produced fewer numbers of frequent itemsets than the other algorithms.
5. In all but one of the experiments that we have conducted, the Partitioning algorithm was inferior to Apriori algorithm where it is supposed to be an improvement of Apriori.

We have noticed that there is a slight differences in our results in comparison with the results of the plots in (Agrawal,1994, Toivonen, 1996.& Sarasere, 1995)., (actual statistical tables of these plots were not available). The differences in results could be due:

1. To the differences in the implementation of some of the used data structures and programming methodology.
2. To different programming languages, compilers, operating system and different machine architectures.
3. The unavailability of the implementations of original algorithms.

From the experience gained during the course of this works, the authors would like to advice with the following recommendations:

1. To experiment the system Apriori-SP with more data of different sizes.
2. To find criteria to relate number of partitions, sample size and the computer memory.
3. The use concept of the negative boarder to find the missed frequent itemsets in the Sampling algorithm.
4. To extend the algorithm Apriori-SP to work in a parallel fashion.

REFERENCE

Agrawal, R., Imielinski, T., & Swami, A. (1993). Mining association rules between sets of items in large databases. *In Proceedings of International ACM SIGMOD Conference on Management of Data,* Washington D.C. 1, 207-216.

Agrawal, R., & Srikant R., (1995). Mining sequential patterns. In P. Yu & A. Chen (Eds.), *Int. Conf. Data Engineering (ICDE)(* pp. 3-14). IEEE computer press.

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. *In Proceedings of 20th International Conference on Very Large Databases*. 1, 478-499.

Brin, S., Motwani, R., Ullman, J. D., & Tsur, S. (1997). Dynamic itemset counting and implication rules for market basket data. *SIGMOD Record (ACM Special Interest Group on Management of Data),* 26(2). 255-264.

Cheung, D., Han, J. & Wong, C.(1996). *Maintenance of discovered association rules in large databases: An incremental updating technique*. New Orleans: IEEE computer society press

Chen, M. S., Han, J., & Yu, P.S. (1996). Data Mining: An overview from a database perspective. *IEEE Transactions on Knowledge and Data Engineering* . 8(6), 866-833.

Han, J. (1995). Discovery of multiple-level association rules from large databases. *In Proc. Of the 21st International Conference on Very Large Databases (VLDB)*. 1, 420-431.

Han, J., & Kamber, M., (2001). *Data mining: concepts and techniques.* San Francisco: Morgan Kaufmann publishers.

.Huhtala, Y., Karkkainen J., Pokka, P., & Toivonen, H. (1999). An efficient algorithm for discovering functional and approximate dependencies. *The computer Journal,* 42(2), 100-111.

.Huhtala, Y., Kinen, J., Pokka P., & Toivonen, H. (1998). Efficient discovery of functional and approximate dependencies using partitions. In P. Yu & A. Chen (eds.). *In Proceedings of 11th International Conference on* Data Engineering (ICDE) (pp. 392-40).

IBM Quest Data Mining Project Synthetic Data Generation Program: http://www.almaden.ibm.com/cs/quest/syndata.html.

Liu, B., Hsu, W., and Ma, Y. (1998). *Integrating classification and association rule mining. :Proc. 1998 Int. Conf. Knowledge Discovery and Data Mining (KDD'98).* New York.

Park, J. S., Chen, M. S. and YU, P. S. (1995). An effective hash based algorithm for mining association rules. In M. J. Carey and D.A. Schneider (eds.), *Proceedings of the 1995 ACM SIG-MOD International Conference on Management of Data.*(PP. 175-186). California.

Sarasere, A., Omiecinsky, E., and Navathe, S., (1995). An efficient algorithm for mining association rules in large databases: *In Proceedings of 21st International Conference on Very Large Databases (VLDB)*. 432-244.

Toivonen, H. Mannila H., and Verkamo, A. (1995). Discovering frequent episodes in sequences. In proceedings of the First International Conference on knowledge Discovery and Data Mining.(PP. 210- 215). AAAI press.

Toivonen, H. (1996). Sampling large databases for association rules. *VLDB Journal*, 134-145.

.Zaki, M. (1999). Parallel and distributed association mining: A Survey. IEEE concurrently. *Special Issue on Parsllel Mechanism for Data Mining* . 7,(4), 14-25.

التجَمْيع البعض مِنْ تحسيناتَ خوارزميةِ أبراير

فرج المؤدب، سالم محمد
قسم الكمبيوتر ، كلية العلوم، جامعة قاريونس، بنغازي، ليبيا

ملخص البحث

التنقيب فى البيانات قضيةٌ مهمةٌ في مجال الانظمة الخاصة بالبياناتِ والمعرفةِ، و التي كانت دافِعَا لاستحداث مجال جديد يعرف بإكتشافَ المعرفةِ في قواعدِ البيانات إلمعرفة المُكتَشَفة يُمْكِنُ أنْ تَأتَ ي في العديد مِنْ الأشكالِ مثل : قواعد الارتباطُ، إلارتباطات، السلاسل، السناريوات، اشجار التصنيفات، التصنيفات العنقودية و الكثير من الانواع الاخرى التنقيب على قواعدِ الارتباط كان موضع اهتمام السَنَوات الأخيرة . الحافز الأصلي للبحث في قواعد الارتباطُ جاءَ مِنْ الحاجةِ لتَحليل ما يسمّى ببياناتِ عمليات الاسواق أو المحلات المركزية، و ذلك لفَحْص سلوكِ الزبون من ناحية المُنتَجاتِ الشُتَريةِ . لقد طور العديد من الخوارزميات للتنقيب على قواعدِ الارتباط . إنّ الخوارزمية ابرير هى إحدى وأكثر الخوارزمياتِ المشهورةِ للتنقيب على قواعدِ الارتباط. و هناك بَعْض التحسيناتِ التي اجريت على هذه الخوارزميةِ

في هذه الورقةِ، سَنَتحرّى عن البعض مِنْ التحسيناتِ لخوارزميةِ ابرير ونَجْمعُ إثنان منهم و هما : أخذ العينات والتَقسيم فى خوارزميةِ واحدة و نسميها خوارزمية ابرير العينة و التقسيمية. سوف سَنَعْرضُ أيضاً خوارزمية ابرير العينة و التقسيمية عن طريق عددِ التجاربِ لقيَأس الأداءِ النسبي خوارزمية ابرير العينة و التقسيمية مقارنَة بخوارزمية العينة و خوارزمية التقسيم كل منهم على حدة.

مفتاح الكلمات: التنقيب فى البيانات؛ قواعد الارتباطُ؛ البحث في قواعد الارتباطُ؛ خوارزمية ابرير.