False Positive Item set Algorithm for Incremental Association Rule Discovery

Ratchadaporn Amornchewin[1], and Worapoj Kreesuradej[2]

[1]Thepsatri Rajabhat University, Lopburi, Thailand
[2]King Mongkut's Institute of Technology Ladkrabang, Thailand
ramornchewin@yahoo.com, *worapoj@it.kmitl.ac.th*

Abstract

In a dynamic database where the new transaction are inserted into the database, keeping patterns up-to-date and discovering new pattern are challenging problems of great practical importance. This may introduce new association rules and some existing association rules would become invalid. Thus, the maintenance of association rules for dynamic database is an important problem. In this paper, false positive itemset algorithm, which is an incremental algorithm, is proposed to deal with this problem. The proposed algorithm uses maximum support count of 1-itemsets obtained from previous mining to estimate infrequent itemsets, called false positive itemsets, of an original database. False positive itemsets will capable of being frequent itemsets when new transactions are inserted into an original database. This can reduce a number of times to scan an original database. This paper also proposes a new updating and pruning algorithm that guarantee to find all frequent itemsets of an updated database efficiently. The simulation results show that the proposed algorithm has a good performance.

1. Introduction

The knowledge discovery in database (KDD) and data mining is the process of discovering and extracting information or pattern from the relevant sets of data in database. Among discovering many kinds of knowledge in database, Association rules mining was a form of data mining to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in the databases. This is due to its wide applicability in many areas, including decision support, market strategy and financial forecast.

The problem of finding association rules was introduced by aggrawal[2] and has been attracted great intention in database research communities in recent years. Association rule mining is widely used in several business such as mobile data service environment [20], intelligent transportation system [16], market basket analysis [6, 18], hospital information system [11], etc.

Mining association rules can be decomposed into two steps: the first is generating frequent itemsets. The second is generating association rules. The main challenge in association rule is to identify frequent itemsets. Finding frequent itemset is one important step in association rule mining. Because the solution of second subproblem was rather straightforward, so most research had focus on how to generate frequent itemsets.

The rules discovered from a database only reflect the current state of the database. However, in a dynamic database where the new transaction are inserted into the database, keeping patterns up-to-date and discovering new pattern are challenging problems of great practical importance. Because of these update activities, new association rules may appear and some existing association rule would become invalid. As a brute force approach, apriori algorithm may be applied to mining a whole dynamic database when the database has been changed. To re-mining the frequent itemsets of the whole updated database is clearly inefficient, because all computations done in the previous mining are wasted. The main idea of the association rule mining in dynamic database refers to optimizations that can be done across mining computations over updated dataset based on previously stored knowledge. Then several research works [5, 7, 12, 17, 21, 26] have proposed several incremental algorithms to deal with this problem. Review of previous works will be introduced in section 2.

In this paper, we propose a new incremental association rule discovery algorithm, called False Positive Itemset algorithm for incremental association rule discovery. The main goal of this work is to solve the updating problem of association rules after a number of new records have been added to a database. False positive itemsets are infrequent itemsets that may be potentially frequent when new transactions are inserted into the original database. In our approach, we can find false positive itemsets by using the maximum support count of 1-itemsets obtained from the previous mining. Our algorithm can reduce a number of times to scan an original database. As a result, the algorithm has execution time faster than that of previous methods.

2. Previous Work

An influential algorithm for association rule mining is Apriori [2]. Apriori computes frequent itemsets in a large database through several iterations based on a prior knowledge. Each iteration has 2 steps which are a joining step and a pruning step. For a frequent itemset, its support must be higher than a user-specified minimum support threshold. The association rule can be discoverd based on frequent itemsets. Based on Apriori algorithm, many new algorithms [22] were designed with some modifications or improvements.

For dynamic databases, several incremental updating techniques have been developed for mining association rules. One of the previous works for incremental association rule mining is FUP algorithm that was presented by Cheung et al [8]. FUP algorithm is the first incremental updating technique for maintaining association rules when new data are inserted into database. Based on the concepts of Apriori algorithm, FUP computes frequent itemsets using large itemsets found at previous iteration. The major idea of FUP is re-using frequent itemsets of previous mining to update with frequent itemsets of an incremental database. At each iteration, the supports of the size-k frequent item sets of an original database are updated by scanning an incremental database. As well as any k-frequent itemset of the incremental database are updated by scanning the original database to find the new frequent itemsets. As a result, FUP algorithm requires to scan passes over an original database several times when new frequent itemsets are found. This can degrade the performance of FUP algorithm.

To deal with the rescanning problem, negative border approach is presented by Toivonen [25], Thomas et al [23] and Feldman et al [12].This approach maintains both frequent itemsets and border itemsets. The border itemset is not a frequent itemset but all its proper subsets are frequent itemsets. The approach need to keep a large number of border itemsets in order to reduce scanning times of an original database. Basically, the border-based algorithms start by scanning a new database. Then, the border-based algorithms update support counts of all frequent sets and border sets. Most updated frequent itemsets can be found not only from frequent itemsets but also from border itemsets. This can reduce scanning times of an original database. However, when new frequent itemsets are introduced as updated frequent itemsets, several database scanning is required to obtain support counts of the new frequent itemsets and their subsets. Adnan et al [1] shows that the execute time of the border-based algorithms can severely slower than that of Apriori when new frequent itemsets are introduced as updated frequent itemsets.

Although a large number of itemsets in the border itemsets is not become frequent items when a new database is added to an original database, the border-based algorithms still need to keep them in order to guarantee that all frequent itemsets can be found. Thus, the border-based algorithms need large memory space to keep the border itemsets.

To reduce memory space, pre-large algorithm [15] is presented. The algorithm maintains both frequent itemsets and pre-large itemsets. Pre-large itemsets are not a frequent itemsets but is capable of being frequent itemsets when a new database is added to an original database. Pre-large itemsets have lesser members than the border itemsets. As a result, the approach uses lesser memory space than the border based approach. In order to guarantee that all frequent itemsets can be found when a new database is added to an original database, the approach can only allow very small size of an incremental database to insert into an original database. When the size of an incremental database exceeds the limited threshold, the algorithm requires rescanning of an original database in order to guarantee that all frequent itemsets can be found.

3. False positive incremental association rule discovery algorithm

When a dynamic database is inserted new transactions in original database, not only some existing association rules may be invalidated but also some new association rules may be discovered. This is the case because frequent itemsets can be changed after inserting new transactions into a dynamic database. Therefore, an association rule discovery algorithm for a dynamic database has to maintain frequent itemsets when new transactions are inserted into the dynamic database. However, we also endeavor to maintain itemsets that are false positive itemset, i.e., infrequent itemsets that have the potential to become frequent itemsets when new transactions are added to original database.

In an observation the itemset will be frequent itemset in updated database if it is member of large itemset in original database or incremental database. The main problem of incremental update is changing of frequent itemset that cause to re-execute from original database again.

In this paper, we present the new idea to avoid scanning the original database. Then we compute not only frequent itemset but also compute itemset that may be potentially large in an incremental database called “False positive Itemset”. An algorithm finds all possible k-itemset of false positive itemset in original database. If member of frequent for each iteration is more than or equal to k-itemset. This idea is guarantee that false positive itemset algorithm are cover all frequent itemset that occur in updated database.Thus, updating the new transactions are quickly because it can use the information from the existing original database.

In this section we describe our algorithm into 2 subsections. In our approach, an original database is firstly mined and all frequent itemsets and false positive itemset. Secondly each incremental dataset in mined and updated to frequent and false positive itemset. The result of updating, some infrequent itemsets or new itemsets may be changed into frequent itemset.

3.1. Original database Discovery

A dynamic database may allow insert new transactions. This may not only invalidate existing association rules but also activate new association rules. Maintaining association rules for a dynamic database is an important issue. Thus, this paper proposes a new algorithm to deal with such updating situation. Our assumption for the new algorithm is that the statistics of new transactions slowly change from original transactions. According to the assumption, the statistics of old transactions, obtained from previous mining, can be utilized for approximating that of new transactions. Therefore, support count of itemsets obtained from previous mining may slightly different from support count of itemsets after inserting new transactions into an original database that contains old transactions. The new algorithm uses maximum support count of 1-itemsets obtained from previous mining to estimate infrequent itemsets of an original database that will capable of being frequent itemsets when new transactions are inserted into the original database. With maximum support count and maximum size of new transactions that allow insert into an original database, support count for infrequent itemsets that will be qualified for frequent itemsets, i.e. min_FP, is shown in equation 1*:*

$$\mathrm{min\_sup}_{DB} \quad \frac{\mathrm{maxsupp}}{\mathrm{total\ size}}) \times \mathrm{inc\_size} \leq \mathrm{min\_FP} < \mathrm{min\_sup}_{DB} \quad (1)$$

where $\mathrm{min\_sup}_{(DB)}$ is minimum support count for an original database, maxsupp is maximum support count of itemsets, inc_size is a maximum number of new transactions and total size is the summation of transaction of an original database and inc_size.
Here, a false positive itemsets is defined as following definition:
Definition : A false positive itemset is an infrequent itemset that satisfies the equation 1.

As an example, an original database shown in figure 1. has 10 transactions, i.e. |DB|=10. Then, three new transactions is inserted into the original database, i.e. |db| =3. Here, minimum support count for mining association rules is set to 4 (40 percent).

Original Database	
TID	List of item
1	A, B, E
2	B, D
3	B, C
4	A, B, D
5	A, C
6	B, C
7	A, C
8	A, B, C, E
9	A, B, E
10	A, C

C1(DB)	Support
A	7
B	7
C	6
D	2
E	3

Figure 1. Transaction data and candidate 1-itemsets

From figure 1, maximum support count of 1-itemsets of the original database is 7. min_FP is computed as the follows:

$$min\_FP = 4 \quad \frac{7}{13} \times 3 \quad = 3 \qquad (2)$$

According to equation 2, if any itemset has support count at least 3 but less than 4, then it will be false positive itemsets. Thus, the frequent 1- itemset is {A, B, C} and the false positive 1- itemset is { E}.

In this paper, Apriori algorithm is applied to find all possible frequent k- itemsets and false positive k-itemsets. Apriori scans all transactions of an original database for each iteration with 2 steps processes are join and prune step. Unlike typical apriori algorithm, items in both frequent k- itemsets and false positive k-itemsets can be joined together in the join step. For a frequent item, its support count must be higher than a user-specified minimum support count threshold and for a false positive item, its support count must be higher than min_FP but less than the user-specified minimum support count. As examples, figure 2. and 3. show the false positive and frequent 2- itemsets and the false positive and frequent 3- itemsets respectively.

F1(DB)	Support
A	7
B	7
C	6

FP1(DB)	Support
E	3

$(F_1 \cup FP_1) \quad (F_1 \cup FP_1)$

C2(DB)	Support
AB	4
AC	4
AE	3
BC	3
BE	3
CE	1

Figure 2. frequent, false positive 1-itemset and candidate 2- itemset

F2(DB)	Support
AB	4
AC	4

FP2(DB)	Support
AE	3
BC	3
BE	3

$(F_2 \cup FP_2) \quad (F_2 \cup FP_2)$

C3(DB)	Support
ABE	3

FP3(DB)	Support
ABE	3

Figure 3. candidate, frequent and false positive itemset 2- 3 itemset

3.2. Updating frequent and false positive itemsets

When new transactions are added to an original database, an old frequent k-itemset could become an infrequent k-itemset and an old false positive k-itemset could become a frequent k-itemset. This introduces new association rules and some existing association rules would become invalid. To deal with this problem, all k-itemsets must be updated when new transactions are added to an original database. The notation used in this section is given in Table1.

Table 1. The notation for Updating frequent and false positive itemsets algorithm

DB	Original database
db	Incremental Database
UP	Updated database
k	Number of itemset
σ	Minimum support (min_sup)
ρ	Minimum False positive (min_FP)
C_k	Candidate k-itemset
F_k	Frequent k-itemset
FP_k	False positive k-itemset

Here, a new updating algorithm shown in figure 4 is proposed in this paper. The algorithm consists of three phases. The first phase is generating k- candidate itemsets, i.e. line2-7. The second phase is repeatedly updating the k- frequent and false positive itemsets by using only an incremental database, i.e. line 8. The third phase is scanning an original database, i.e. line 11-16.

Algorithm1 : Main Algorithm

Input : DB, db, k, $\sigma^{UP}, \rho^{UP}, \rho^{DB}, C_1^{DB}, F_1^{DB}, FP_1^{DB}$ and their count

Output : F_k^{UP}, FP_k^{UP}

1. for $(k = 1; F_k^{UP} \neq \phi; k++)$ do
2.   if $k = 1$ then
3. $C_1^{db} = \{X \mid X \in itemset_1^{db}\}$
4.   //X = each item that is member of the set of candidate1 itemset
5.   else //$k > 1$
6.   Generate Candidate Itemset
7.   end if
8.   Update k itemset (return m, Temp_scanDB)
9.   //m is the maximum itemset of Temp_scanDB
10. end do
11. $k = 2$
12. while ($Temp\_scanDB_k \neq \phi$ and $(k \leq m)$) do
13.   Scan Original Database($Temp\_scanDB_k$)
14. $k = k + 1$
15. end do
16. clear Temp_scanDB

Figure 4. Main Algorithm

The algorithm for generating k- incremental candidate itemsets for k greater than or equal to 2 is shown in figure 5. For k=2, the 2- incremental candidate itemsets are easily obtained by joining F_1^{UP} with F_1^{UP}, i.e. line 3. For k>2, the algorithm is firstly find k- candidate itemsets of an incremental database, i.e., C_k^{db} by joining F_{k-1}^{db} with F_{k-1}^{db} , i.e. line 7. Similar to Apriori algorithm, the k- candidate itemsets of an incremental database can be the updated frequent itemsets, i.e., F_1^{UP} only if the subsets of the k-candidate itemsets of an incremental database must be in the (k-1) - updated frequent. Thus, the k- incremental candidate itemsets, i.e. C_k^{new} , will keep only the k- candidate itemsets of an incremental database whose subsets of the k- candidate itemsets are in the (k-1) - updated frequent intemsets, i.e. line 9. This can prune the k- candidate itemsets of an incremental database that can't be the k- updated frequent intemsets.

Algorithm 2: Generating Candidate k- itemsets

Input : $F_1^{UP}, F_{k-1}^{UP}, F_{k-1}^{db}, k$

Output : C_k^{new}

1. if $k = 2$ then
2. $\quad C_2^{db} = F_1^{UP} * F_1^{UP}$
3. $\quad$ for all $X \in C_2^{db}$ do
4. $\quad\quad C_2^{new} = \left\{X \in C_2^{db} \middle| X \notin \left(F_2^{DB} \cup FP_2^{DB}\right)\right\}$
5. $\quad$ end do
6. else if $k > 2$ then
7. $\quad C_k^{db} = F_{k-1}^{db} * F_{k-1}^{db}$
8. $\quad$ for all $X \in C_k^{db}$ do
9. $\quad\quad C_k^{new} = \left\{X \in C_k^{db} \middle| X \in F_{k-1}^{UP} \text{ and } X \notin \left(F_k^{DB} \cup FP_k^{DB}\right)\right\}$
10. $\quad$ end do
11. end if

Figure 5. Generating candidate k- itemsets algorithm

The second phase show in figure 6. This phase has 2 major steps which are a an updating support count of k- incremental frequent and k- incremental false positive itemsets step and finding support count of new k- incremental candidate itemsets step. According to the algorithm, the 1-candidate itemsets of an updated database, i.e. C_1^{UP}, can be found by combining the 1-candidate itemsets of an original database, i.e. C_1^{DB}, with the 1-candidate itemsets of an incremental database, i.e. C_1^{db} .Then, the support count of C_1^{UP} can be updated by scanning only an incremental database, i.e. line 1-6.

For k>1 , the algorithm scans an incremental database to find and update support count of the *k*- updated candidate itemsets, i.e. C_k^{UP} . When any *k*-itemsets are not in the union set of the original *k*-frequent and the original *k*- false positive itemsets, i.e. k-itemsets $\notin F_k^{DB} \cup FP_k^{DB}$, but is in the k- incremental candidate itemsets, i.e. *k*- itemsets $\in C_k^{new}$, their support counts need to be specially updated.

Algorithm 3 : Update k-itemset

Input: $DB, db, \sigma^{UP}, \rho^{UP}, \rho^{DB}, C_1^{DB}, F_1^{DB}, FP_1^{DB}, C_1^{db}, F_k^{DB}, FP_k^{DB}$ and their count

Ouiput: F_k^{UP} and FP_k^{UP}, F_k^{db}, Temp_scanDB and their count, m

if $k = 1$ then
  Scan db and find count $c(X,db)$ for all $X \in C_1^{DB} \cup C_1^{db}$
    for all $X \in C_1^{DB} \cup C_1^{db}$ do
      $c(X,UP) = c(X,DB) + c(X,db)$
      $F_1^{db} = \left\{ X \notin C_1^{DB} \text{ and } X \in C_1^{db} \middle| c(X,db) \geq \sigma^{UP} \right\}$
    end do
    $m = \phi$
    $\text{Temp\_scanDB}_k = \phi$
    //$k = 1$, return empty set for m and Temp_scanDB
else//$k > 1$
  Scan db and find count $c(X,db)$ and $c(Y,db)$
    \\ $X \in \left(F_k^{DB} \cup FP_k^{DB}\right)$ and $Y \in C_k^{new}$
    for all $\left(X \in \left(F_k^{DB} \cup FP_k^{DB} \right) \right)$ and $\left(Y \in C_k^{new} \right)$ do
      if $X \in \left(F_k^{DB} \cup FP_k^{DB}\right)$ and $X \notin C_k^{new}$ then
        $c(X,UP) = c(X,DB) + c(X,db)$
      else if $X \in \left(F_k^{DB} \cup FP_k^{DB}\right)$ and $X \notin C_k^{new}$ then
        $c(X,UP) = c(X,DB)$
      else if $Y \notin \left(F_k^{DB} \cup FP_k^{DB}\right)$ and $Y \in C_k^{new}$ then
        if $c(Y,db) \geq \sigma^{db}$
          $\text{Temp\_scanDB}_k = \{ Y \,\Big|\, \left(\left(c(Y,db) + \left(\rho^{DB} - 1 \right) \right) \geq \sigma^{UP} \right)$
            or $\left(c(Y,db) + \left(\rho^{DB} - 1 \right) \right) \geq \rho^{UP} \}$
          $F_k^{db} = \left\{ Y \mid c(Y,db) \geq \sigma^{db} \right\}$ // $k \geq 2$
          $m = k$
        end if
      end if
    end do
end if
$F_k^{UP} = \left\{ X \mid c(X,UP) \geq \sigma^{UP} \right\}$
$FP_k^{UP} = \left\{ X \mid \rho^{UP} \leq c(X,UP) < \sigma^{UP} \right\}$

Figure 6. Update k-itemsest algorithm

This is the case because their support counts obtained from an original database are not available. Since these *k*- itemsets are not in $F_k^{DB} \cup FP_k^{DB}$, their support counts are at best equal to $\rho^{DB} - 1$. Here, their support counts are assumed to be equal to the sum of $\rho^{DB} - 1$ and their support counts obtained from an incremental database, i.e. $c(X, db) + (\rho^{DB} - 1)$. If any k- itemsets have support counts below updated min support count, i.e. σ^{UP}, the *k*-itemsets can't be the *k*-updated frequent itemsets. On the other hand, if any *k*-itemsets have support counts above or equal to an updated min support count, the *k*-itemsets are likely to be the *k*-updated frequent itemsets. Thus, the *k*-itemsets, which have support counts above or equal to an updated min support count, are move to Temp_scanDB, i.e. line 10-29. Temp_scanDB_k is set aside for finding their true support counts from an original database, i.e. line 20-21. Then, the k-frequent and false positive itemsets of an updated database can be found as shown in line 28 and 29 respectively.

At the third phase, an original database is scanned to find true support counts for the *k*-itemsets that are likely to be the *k*-updated frequent itemsets. The algorithm is shown in Figure 7. The support counts of the likely *k*-updated frequent itemsets are found and updated by scanning an original database as shown in line 1-6. Finally, all *k*-updated frequent itemsets and *k*-updated false positive itemsets are found as shown in line 7-8.

Algorithm 4 : Scanning an original database

Input : $\text{Temp\_scanDB}_k, \sigma^{UP}, \rho^{UP}, F_k^{UP}, FP_k^{UP}$ and their count

Output : F_k^{UP}, FP_k^{UP} and their count

1. Scan DB and obtain count $c(X, DB)$ for all Temp_scanDB_k
2. for all $X \in \text{Temp\_scanDB}_k$ do
3. $\quad c(X, UP) = c(X, DB) + c(X, db)$
4. end do
5. $F_k^{new} = \{ X \mid X \in \text{Temp\_scanDB}_k \text{ and } c(X, UP) \geq \sigma^{UP} \}$
6. $FP_k^{new} = \{ X \mid X \in \text{Temp\_scanDB}_k \text{ and } \rho^{UP} \leq c(X, UP) < \sigma^{UP} \}$
7. $F_k^{UP} = F_k^{UP} \cup F_k^{new}$
8. $FP_k^{UP} = FP_k^{UP} \cup FP_k^{new}$

Figure 7. Algorithm for scanning an original database

4. Experiment

To evaluate the performance of false positive algorithm, the algorithm is implemented and tested on a PC with a 2.8 GHz Pentium 4 processor, and 1 GB main memory. The experiments are conducted on a synthetic dataset, called T10I4D10K. The technique for generating the dataset is proposed by Agrawal and etc. [2]. The synthetic dataset comprises 100,000 transactions over 70 unique items, each transaction has 10 items on average and the maximal size itemset is 4

Firstly, the proposed algorithm is used to find association rules from an original database of 20,000 transactions. Then, several sizes of incremental databases, i.e. 1%,2%,3%,4%, 5%, 6% and 7% of the original database, are added to the original database. For comparison purpose, FUP, Border and Pre-large algorithms are also used to find association rules from the same original database and the same incremental databases. The experimental results with the same minimum support thresholds, i.e.4% of minimum support threshold, are shown in Table2 and Figure 8. From the results, the proposed algorithm has much better running time than that of FUP, Border and Pre-large algorithms.

Table 2. Execution time with varying size of incremental database

Incrementa size(%)	Execution time (sec.)			
	Min_sup = 4%			
	False positive Itemset	FUP	Pre-large	Border
1	51.89	5465.47	558.86	882.97
2	111.22	3960.03	1880.23	1797.94
3	195.47	4117.7	1941.44	2309.41
4	310.28	4070.98	2572.74	2734.64
5	471.74	4265.91	3502.88	3783.8
6	713.92	4367.81	4868.39	4234.3
7	1149.06	4601.11	7766.53	4971.33

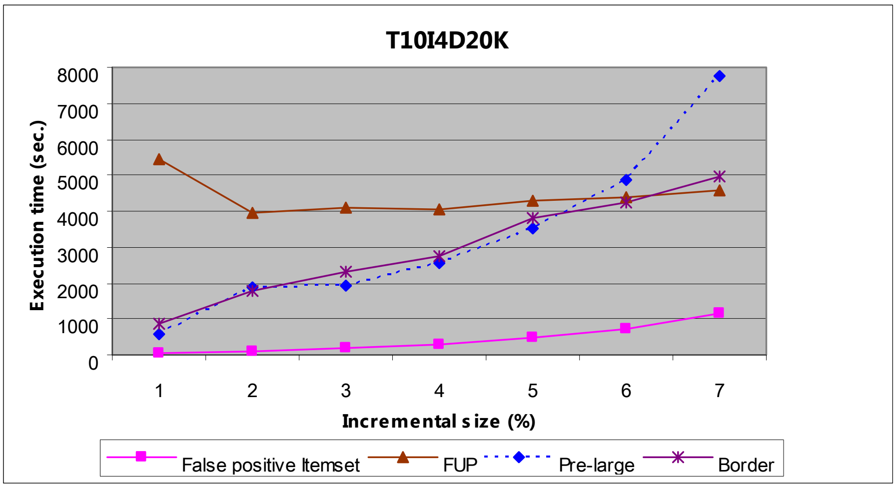

Figure 8. The execution time of False positive itemset, FUP,Borders and Pre-large algorithm

5. Conclusions

We have proposed false positive itemset algorithm for incremental association rule mining. Assuming that a minimum support factor and a confidence factor do not change, false positive itemset algorithm can guarantee to discover frequent itemsets. Our algorithm can reduce a number of times to scan an original database. From the experiment, our algorithm has better running time than that of FUP, Border and

Pre-large algorithm. In the future, further thorough investigation and experiments on the proposed algorithm will be conducted.

References

[1] Adnan, M., Alhajj, R., and Barker, K.: "Performance Analysis of Incremental Update of Association Rules Mining Approaches"; In Proceeding of 9th IEEE International Conference on Intellgent Engineering System 2005, Sept. 16-19, 2005. 129-134.
[2] Agrawal, R., Imielinski, T., and Swami, A.: Mining association rules between sets of items in large databases , In Proceeding of the ACM SIGMOD Int'l Conf. on Management of Data (ACM SIGMOD '93), Washington, USA, May 1993, pp. 207-216.
[3] Agrawal, R. and Srikant, R.: "Fast algorithms for mining association rules , In Proceedings of 20 th Intl Conf. on Very Large Databases (VLDB'94) "; Santiago, Chile (1994), 478 -499.
[4] Alon, N., and Spencer, J. H. 1992. "The probabilistic method"; Wiley Interscience,New York.
[5] Ayan, N.F., Tansel, A.U., and Arun, E.: "An efficient algorithm to update large itemsets with early pruning"; Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, August 1999, 287-291.
[6] Brin, S., Motwani, R., and Silverstein, C.: "Beyond Market Baskets: Generalizing Association Rules to Correlations"; Proceedings of the ACM SIGMOD Conference,(1997), 265-276.
[7] Chang, C.C., Li Y.C., and Lee, J.S.: "An efficient algorithm for incremental mining of association rules "; Proceedings of the 15th international workshop on research issues in data engineering: stream data mining and applications (RIDE-SDMA'05), IEEE, 2005.
[8] Cheung, D., Han, J., Ng, V., and Wong, C. Y.: "Maintenance of discovered association rules in large databases: An incremental updating technique"; In 12th IEEE International Conference on Data Engineering, February 1996, 106-114.
[9] Cheung, D. W., Lee, S.D., and Kao, B.: "A General incremental technique for maintaining discovered association rules";In Proceedings of the 5 th Intl. Conf. on Database Systems for Advanced Applications (DASFAA'97), Melbourne, Australia, April 1997, 185-194.
[10] Du, Y. Zhao, M., Fan G.: "Research on Application of Improved Association rules Algorithm in intelligent QA system"; Second International Conference on Genetic and Evolutionary Computing, 2008.
[11] Elfangary, L.,Atteya, W.A.: "Mining Medical Database using Proposed Incremental Association rule Algorithm (PIA)"; Second International Conference on the Digital Society, IEEE (2008).
[12] Feldman, R., Aumann,Y., and Lipshtat, O.: "Borders: An efficient algorithm for association generation in dynamic databases"; Journal of Intelligent Information System, 1999. 61-73.
[13] Gong M. :"Personalized E-learning System by Using Intelligent Algorithm"; 2008 Workshop on Knowledge Discovery and Data Mining, IEEE, 2008.
[14]Guo, M., Zhao Y.: "An Extensible Architecture for Personalized Information Services in An Ambient Intelligence Environment"; IEEE 2008.
[15] Hong, T.P., Wang, C.Y., and Tao, Y.H.: "A new incremental data mining algorithm using pre-large itemsets"; Journal of Intelligent Data Analysis, Vol. 5, No.2 2001. 111-129.
[16] Juan X., Feng Y., Zhiyong Z.: "Association Rule Mining and Application in Intelligent Transportation System"; Preceedings of the 27th Chinese Control Conference, China (2008), 538-540.
[17] Lee, C. H., Lin, C. R., and Chen,M. S.: "Sliding-Window Filtering: An Efficient Algorithm for Incremental Mining"; Proceeding of the ACM 10th International Conference on Information and Knowledge Management , November 2001.
[18] Liu, B., Hsu W., and Ma, Y.: "Mining Association Rules with Multiple Supports:; ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-99), SanDiego, CA, USA. August 1999.
[19] Luhr, S., Venkatesh, S., West, G.: "Emergent Intertransaction Association Rules for Abnormality Detection in Intelligent Enviroments"; Proceedings of the 2005 International Conference on Intelligent Sensors, Sensor Networks and Information Processing Conference, December 2005, 343-347.
[20] Pawar P.A., Aggrawal A. K.: "Associative Rule Mining of Mobile Data Services Usage for Preference Analysis, Personalization & Promotion"; Proceeding of WSEAS International Conference on Simulation, Modeling and Optimization, Izmir, Turkey, September 2004.
[21] Sarda, N. L., and Srinivas, N. V.: "An adaptive algorithm for incremental. mining of association rules"; In Proc. 9th Intl.Workshop on Database and Expert System Applications,Vienna, Austria, Aug 1998. 240-245.
[22] Savasere, A., Omiecinski, E., Navathe S.: "An Efficient Algorithm for Mining Association Rules in large database"; Proceedings of the 21st VLDB Conference, Zurich, Switzerland (1995).
[23] Thomas, S., Bodagala, S., Alsabti, K., and Ranka, S.: "An efficient algorithm for the incremental updation of association rules in large databases"; In Proceedings of the 3rd Intl. Conf. on Knowledge Discovery and Data Mining (KDD'97), New Port Beach, California, August 1997, 263-266.

[24] Thuraisingham, B.M., Ceruti, M.G.: "Understanding Data Mining and Applying it' to
Command, Control, Communications and Intelligence Environments"; The Twenty-Fourth Annual International Computer Software and Applications Conference (COMPSAC), 2000, 171-174.
[25] Toivonen, H. 1996.: "Sampling large databases for association rules"; In Proceedings
of the 22th International Conference on Very Large Databases (VLDB), September 1996, 134-145.
[26] Veloso, A. Meira Jr., W., Carvalho, M., Parthasarathy, S., and Zaki ,M. J.: "Parallel, Mining frequent itemsets in evolving databases Databases"; In Proc. 2nd SIAM Intl. Conf. on Data Mining, Arlington, VA, April 2002.
[27] Zhang, C., Zhang, S., and Webb, G. I.: "Identifying Approximate Itemsets of Interest in Large Databases"; Applied Intelligence 2003, 91–104.