Journal of Universal Computer Science, vol. 15, no. 12 (2009), 2409-2428
submitted: 15/12/08, accepted: 25/6/09, appeared: 28/6/09 © J.UCS

Mining Dynamic Databases using Probability-Based Incremental Association Rule Discovery Algorithm

Ratchadaporn Amornchewin
(King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand
ramornchewin@yahoo.com)

Worapoj Kreesuradej
(King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand
worapoj@it.kmitl.ac.th)

Abstract: In dynamic databases, new transactions are appended as time advances. This paper is concerned with applying an incremental association rule mining to extract interesting information from a dynamic database. An incremental association rule discovery can create an intelligent environment such that new information or knowledge such as changing customer preferences or new seasonal trends can be discovered in a dynamic environment. In this paper, probability-based incremental association rule discovery algorithm is proposed to deal with this problem. The proposed algorithm uses the principle of Bernoulli trials to find expected frequent itemsets. 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 better performance than that of previous work.

Keywords: Incremental Association Rule Discovery, Association Rule Discovery, Data mining
Categories: I.1.2, I.2.6

1 Introduction

Recent works in the field of databases have been used in business management, government administration, scientific, engineering data management, and many other applications. This explosive growth in data and databases has generated an urgent need for new techniques and tools that can intelligently and automatically transform the processed data into useful information and knowledge.

During the past few years, data mining has been considered as new techniques and tools for intelligently transforming the processed data into useful information and knowledge. Data mining provides the technique to analyze and convert mass volume of data and/or detect hidden patterns in data into valuable information. Data mining may be applied in an intelligence environment in a number of domains [Guo, Zhao 2008; Thuraisingham, Ceruti 2000; Du et al. 2008; Gong 2008; Juan et al. 2008 and Luhr et al. 2005].

An important technique of data mining is association rules mining [Argrawal 1993] which studies the buying behaviors of customers and thus improves the quality of business decisions. It aims to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in the databases. Association

rule mining is widely used in several business such as mobile data service environment in [Pawar, Aggarwal 2004], intelligent transportation system in [Juan et al. 2008], market basket analysis in [Brin 1997] and [Liu et al.1999], hospital information system in [Elfangary, Atteya 2008], etc.

The rules discovered from a database only reflect the current state of the database. However, in a dynamic database where new transactions are inserted frequently, association rules discovered in the previous database possibly no longer valid and interesting rules in the updated database. As a result, new business information such as changing customer preferences or new seasonal trends may not be discovered. To create an intelligent environment such that new business information can be discovered in a dynamic database, association rules algorithms should be capable of mining a dynamic database incrementally. Several research work [Ayan et al. 1999; Chang et al. 2005; Lee et al. 2001; Feldman et al. 1990; Ratchadaporn, Kreesuradej 2007; Sarda et al. 1998 and Veloso et al. 2002] have proposed several incremental algorithms to deal with this problem. The review of related work will be introduced in section 2.

In this paper, we propose a new incremental association rule discovery algorithm, called Probability-Based Incremental Association Rule Discovery Algorithm, for the environment of a dynamic database. The algorithm is capable of dynamically discovering new association rules when a number of new records have been added to a database. In our approach, we use the probability to predict infrequent itemsets that have capable of being frequent itemsets after a number of new records have been added to a database. That infrequent itemsets is called expected frequent itemsets. 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 [Agrawal,Srikant 1993]. 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 discovered based on frequent itemsets. Based on Apriori algorithm, many new algorithms [Savesere et al. 1995] were designed with some modifications or improvements.

In dynamic databases, new transactions are appended as time advances. This may introduce new association rules and some existing association rules would become invalid. Thus, the maintenance of association rules for dynamic databases is an important problem. Maintaining association rules has been studied popularly in data mining. There are 2 main approaches to mining association rules for dynamic databases incrementally. The first approach assumes that association rules to be found are not stable over time. Recent patterns that represent new trends to be discovered are more interesting than old patterns. Thus, the first approach treats recent added transactions higher important than old transactions. The association rules obtained from the first approach are just the approximation of those obtained by re-running Apriori algorithm.

As the first approach, Zhang et al. [Zhang et al. 2003] proposed a weighting technique for discovering association rules in a dynamic database. The technique gives recent added transactions higher weight than old transactions. Then, frequent itemsets can be maintained incrementally by using a competitive model to promote infrequent itemsets to frequent itemset and degrade frequent itemset to infrequent itemset.

Similar to Zhang's work, Dudek et al. [Dudek et al. 2005] proposed a method of a time influence function in order to reflect that recent patterns are more interesting than old patterns. The time influence function gives recent added transactions higher weight than old transactions. In addition, this work also proposed using estimated support and confidence to deal with situation that new association rules are discovered only in new transactions but are not discovered only in old transactions. However, this work did not propose the method to estimate the estimated support and confidence.

Unlike the first approach, the second approach assumes that association rules to be found are stable over time. Thus, the second approach treats both old and new transactions as of equal importance. As a result, the association rules obtained from the second approach are the same as those obtained by re-running Apriori algorithm. According to Teng and Chen [Teng, Chen 2005], the second approach is categorized into Partition-based, Pattern growth and Apriori-based techniques.

Partition-based techniques partition a database into n partition and processes each partitions for an on going time variant database. For each frequent itemset must exist at least one of the n partitions. Sliding-window filtering (SWF) [Lee et al. 2001] is a partition algorithm for an incremental association mining. The concept of SWF is partitioning a transaction database into n partitions, P_1, P_2, …, P_n and processing one by one. The filtering threshold is employed for selecting frequent itemset in each partition to deal with the candidate itemset generation. The key idea of SWF is to compute candidate 2-itemset as close to frequent 2-itemset as possible. SWF need one scan over the updated database for finding the frequent itemset from the candidate ones. Based on SWF, FI_SWF and CI_SWF [Chang, Yang 2003] are proposed to reduce the number of candidate itemsets of SWF.

Pattern-Growth techniques are basically based on FP-tree to discover association rules. Since FP-tree cannot be directly applied to the problem of incremental mining, two alternative forms of FP-tree, i.e., DB-tree [Ezeife, Su 2002] and Potential Frequent Pattern tree (PotFP-tree) [Ezeife, Su 2002], are proposed to solve the problem.

Apriori-based techniques, which are the most popular techniques, adopt the concepts of Apriori algorithm for incremental mining. Since the algorithm in this work can be classified as Apriori-based techniques, the more comprehensive reviews of Apriori-based techniques will be presented. The first incremental updating algorithm for maintaining association rules when new data are inserted into database is presented by Cheung et al. [Cheung et al. 1996]. 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 increment database. At each iteration, the supports of the size-*k* frequent item sets of an original database are updated by scanning an increment database. As well as any k-frequent itemset of the increment database are updated by scanning the original database to

find the new frequent itemsets. As a result, FUP requires multiple-scan for maintaining frequent itemsets.

For enhance the efficiency of FUP algorithm, UWEP algorithm is presented by Ayan et al. [Ayan et al. 1999]. The major idea of UWEP algorithm is reducing the number of candidate itemset by pruning the supersets of an old frequent itemset in previous mining that no longer remain in an updated database with at most once scanning in original database.

To deal with the rescanning problem, negative border approach is presented by Toivonen [Toivonen 1996], Thomas et al. [Thomas et al. 1997] and Feldman et al. [Feldman et al. 1999].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. [Adnan et al. 2005] 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.

To reduce memory space, Tsai et al. [Tsai et al. 1999], Hong et al. [Hong et al. 2001] and Amornchewin and Kreesuradej [Amornchewin, Kreesuradej 2007] propose a new approach. The approach maintains both frequent itemsets and expected frequent itemsets. Based on FUP algorithm, Tsai et al. use not only support threshold but also the degree, called tolerance degree, to finding the expected frequent itemset. For awhile, Hong et al. use the 2 thresholds, upper support and lower support threshold, for mining frequent and expected frequent itmeset. Amornchewin and Kreesuradej proposes the idea for estimate expected frequent itemsets by using maximum support count of 1-itemsets obtained from prior mining. An expected frequent itemset is not a frequent itemset but is expected to become a frequent itemset when a new database is added to an original database. An expected frequent itemset is not a frequent itemset but is expected to become a frequent itemset when a new database is added to an original database. The expected frequent 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 of Hong et al. [Hong et al. 2001] and Amornchewin and Kreesuradej [Amornchewin, Kreesuradej 2007], can only allow very small size of an increment database to insert into an original database.

To deal with the problem that the previous approach can only allow very small size of an increment database to insert into an original database, a new algorithm, called Probability-based Incremental Association Rule Discovery Algorithm, is introduced by Amornchewin and Kreesuradej [Amornchewin, Kreesuradej 2008]. Similar to the previous approach, the new algorithm also keeps both frequent itemsets and expected frequent itemsets. However, the new algorithm introduces a new technique, which is based on the principle of Bernoulli trials, to estimate expected frequent

itemsets. The preliminary experiments show that the new technique can allow larger size of an increment database to insert into an original database than that of the previous approach. Moreover, the experiments also show that the new technique has execution time faster than that of previous methods.

This paper is the extension work of Probability-based Incremental Association Rule Discovery Algorithm, introduced by Amornchewin and Kreesuradej [Amornchewin, Kreesuradej 2008]. The paper provides more theoretical fundamentals of the algorithm and comprehensive experimental results than that of Amornchewin and Kreesuradej [Amornchewin, Kreesuradej 2008].

3 Probability-based Incremental Association Rule Discovery Algorithm

When a dynamic database is inserted new transactions, 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. The problem description is given as section 3.1. Then, we describe our algorithm into 2 subsections. Firstly, probability-based expected frequent itemsets is presented. Secondly, updating frequent and expected frequent itemsets is introduced.

3.1 Problem description

Let F be the set of all frequent itemsets in the original database, i.e. DB, σ be the minimum support threshold and $|DB|$ be the number of transactions in DB. Assume that the support count of each frequent itemset in the original database is kept. After some updates, an increment database, i.e. db, is added into the original database DB, resulting in an updated database, i.e. UP. The numbers of transactions in db and UP are denoted $|db|$ and $|UP|$, respectively, and the support counts of itemset X in DB, db and UP are c(X,DB), c(X,db), c(X,UP), respectively. With the same minimum support threshold σ, a frequent itemset X of DB remains a frequent itemset of UP if and only if its support in UP is greater than or equal to σ, i.e. $c(X,UP) \geq (|DB|+|db|)*\sigma$.

According to Tsai et al. [Tsai et al. 1999], an itemset, i.e. X, can be categorized into four cases:

Case 1: X is a frequent itemset in both DB and UP.

Case 2: X is a frequent itemset in DB and an infrequent itemset in UP.

Case 3: X is an infrequent itemset in DB and a frequent itemset UP.

Case 4: X is neither a frequent itemset in DB nor UP.

Obviously, the itemset of case 4 cannot change an association rule because the itemset is an infrequent itemset in both DB and UP. The itemset of case 1 and 2 can easily be discovered because updating support count of the frequent itemset, which is trivial tasks, requires only the frequency of the itemset from db. The task of discovering the itemset of case 3 is the hardest task because it requires to rescan an original database. Thus, how to efficiently discovery the itemset of case 3 is an

important problem. In the next subsection, a new algorithm to deal with this problem is proposed.

3.2 Predicting Expected Frequent Itemsets

The task of an association rule discovery algorithm for a dynamic database can be divided into three tasks. The first task is to update support count of existing frequent itemsets. The second task is to prune existing frequent itemsets that have support count below a minimum support threshold after updating the database. The third task is to discover new frequent itemsets that have support count equal or above a minimum support threshold after updating the database.

Updating support count of existing frequent itemsets and pruning existing frequent itemsets are trivial tasks. To perform updating and pruning tasks, the association rule discovery algorithm requires only frequency of itemsets from new transactions. Thus, an association rule discovery algorithm does not need to rescan an original database to perform the tasks.

Unlike the other tasks, the task for discovering new frequent itemsets requires not only frequency of itemsets from new transactions but also that from an original database. An association rule discovery algorithm need to rescan an original database to perform the tasks. Therefore, the discovering new frequent itemsets task is a nontrivial task for maintaining frequent itemsets

For our algorithm, an original database, which is a database before being inserted new transactions, is firstly mined to find all frequent itemsets that satisfy a minimum support count, denoted $k_{original}$. The proposed algorithm also predicts and keeps expected frequent itemsets that may become frequent itemsets if new transactions are inserted into the original database.

The process of inserting m transactions into an original database of n transaction can be considered as (m+n) Bernoulli trials, which are (m+n) sequence of identical trials. Each itemset has its probability of appearing in a transaction, denoted by $p_{itemset}$, i.e. the probability of success. According to the principle of Bernoulli trials, the probability of the number of an itemset to appearing in (n+m) transactions, denoted by $P(x)_{itemset}$, can be found by the following equation:

$$P(x)_{itemset} = \binom{n+m}{x} \cdot p_{itemset}{}^{x} \cdot (1 - p_{itemset})^{n+m-x} \quad ,$$

where $p_{itemset}$ is the probability of an itemset appearing in a transaction, m is a number of new transactions, and n is a number of transactions of an original database. Figure 1 shows the scheme of predicting expected frequent itemsets based on Bernoulli trials.

If k is a minimum support count after inserting new transactions into an original database, the probability of an itemset to be a frequent itemset in an updated database can be obtained as the following equation:

$$P(x \geq k)_{itemset} = 1 \quad P(x < k)_{itemset} \tag{1}$$

According to equation 1, P(x < k) itemset can be found as the following equation:

$$P(x < k)_{itemset} = \sum_{x=0}^{k-1} \binom{n+m}{x} p_{itemset}^{\,x}(1 - p_{itemset})^{n+m-x} \qquad (2)$$

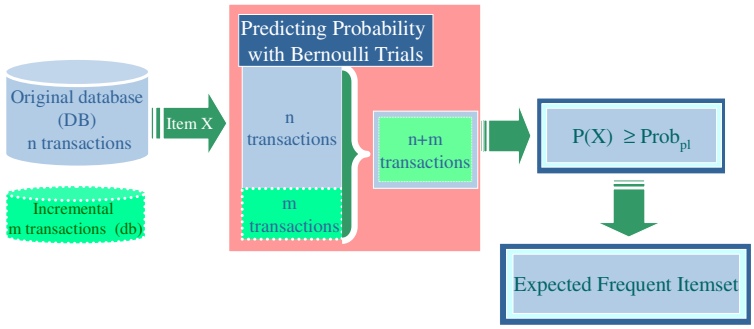

Figure1: The scheme of predicting expected frequent itemsets based on Bernoulli

Here, an expected frequent itemset is an itemset that is not a frequent itemset but has its probability to be a frequent itemset greater than $Prob_{pl}$. $Prob_{pl}$ is a threshold constant specified by users. $Prob_{pl}$ indicates the minimum confidence level that a promising frequent itemset will be a frequent itemset after inserting new transactions into an original database. The higher $Prob_{pl}$ is set, the lesser expected frequent itemsets are kept. As results, the algorithm may need more number of rescanning times in the original database when the algorithm performs the discovering new frequent item task.

From equation 2, the probability of success of an itemset, i.e., $p_{itemset}$, can be derived from the following equation:

$$p_{itemset} = \frac{c(itemset, DB) + c(itemset, db)}{|DB| + |db|}, \qquad (3)$$

where c(itemset, DB) is support counts obtained from an original database, c(itemset, db) is support counts obtained from an increment database, $|db|$ is the total number of transactions of an increment database, and $|DB|$ is the total number of transactions of an original database.

According to the scheme of predicting expected frequent itemsets, an expected frequent itemset need to be obtained from an original database before an increment database is available. Therefore, the probability of success of an itemset has to approximate from an original database. Here, an original database, which has n transaction, is considered as a sample data of (n+m) transactions. The approximation of the probability of success of an itemset can be obtained as

$$\hat{p}_{itemset} = \frac{c(itemset, DB)}{|DB|}, \qquad (4)$$

where c(itemset, DB) is support counts obtained from an original database, i.e., DB, and $|DB|$ is the total number of transactions of an original database.

It is important to evaluate the accuracy of the approximation of the probability of success of an itemset. The following theorem gives a lower bound, i.e. ε , and maximum probability, i.e. δ, for an error of the approximation of the probability of success of an itemset.

Theorem 1[Mannila et al. 1994; Toivonen 1996] The probability that

$$error = \left| p_{itemset} - \hat{p}_{itemset} \right| > \varepsilon$$

at most δ. If the size of an original database satisfies the following equation:

$$\left|DB\right| \geq \frac{1}{2\varepsilon^2} ln \frac{2\left|UP\right|}{\delta},$$

where $|DB|$ is the size of an original database and $|UP|$ is the size of an updated database.

According to the theorem, if size of an original is sufficient large, $\hat{p}_{itemset}$ is a good approximation of $p_{itemset}$. Therefore, equation 2 can be rewritten as follows,

$$P(x < k)_{itemset} = \sum_{x=0}^{k-1} \binom{n+m}{x} \hat{p}_{itemset}^{\,x} (1 - \hat{p}_{itemset})^{n+m-x} \qquad (5)$$

As an example, an original database has 10 transactions, i.e. |DB| =10.Then, five new transactions are inserted into the original database, i.e. |db| =5. Here, minimum support count for mining association rules is set to 4 (40 percent). If we defined probability for a expected itemset is $Prob_{pl} = 0.10$, $P(x \geq k)$ is computed shown in figure 2.

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

$$P(x \geq 6)_A = 1 - \sum_{x=0}^{5} \binom{15}{x} \frac{7}{10}^{x} \frac{3}{10}^{15-x} = 1$$

$$P(x \geq 6)_B = 1 - \sum_{x=0}^{5} \binom{15}{x} \frac{7}{10}^{x} \frac{3}{10}^{15-x} = 1$$

$$P(x \geq 6)_C = 1 - \sum_{x=0}^{5} \binom{15}{x} \frac{6}{10}^{x} \frac{4}{10}^{15-x} = 1$$

$$P(x \geq 6)_D = 1 - \sum_{x=0}^{5} \binom{15}{x} \frac{2}{10}^{x} \frac{8}{10}^{15-x} = 0.06$$

$$P(x \geq 6)_E = 1 - \sum_{x=0}^{5} \binom{15}{x} \frac{3}{10}^{x} \frac{7}{10}^{15-x} = 0.28$$

Figure 2: Transaction data and candidate 1-itemsets

From the example, the frequent 1- itemset is {A, B, C}. Item {E} is an expected frequent 1-itemset because its probability, which is equal to 0.28, is greater than $Prob_{pl}$.

The proposed algorithm generates candidate next k-itemsets by using both frequent k-itemsets and expected frequent k-itemsets. Firstly, new itemsets is obtained by union of frequent k-itemsets and expected frequent k-itemsets. Then, the candidate (k+1)-itemsets

is obtained by self joining the new itemset together. Both frequent (k+1)-itemsets and expected frequent (k+1)-itemsets can be found similar to that of k-itemsets. The example of generating candidate next k-itemsets is shown in figure 3.

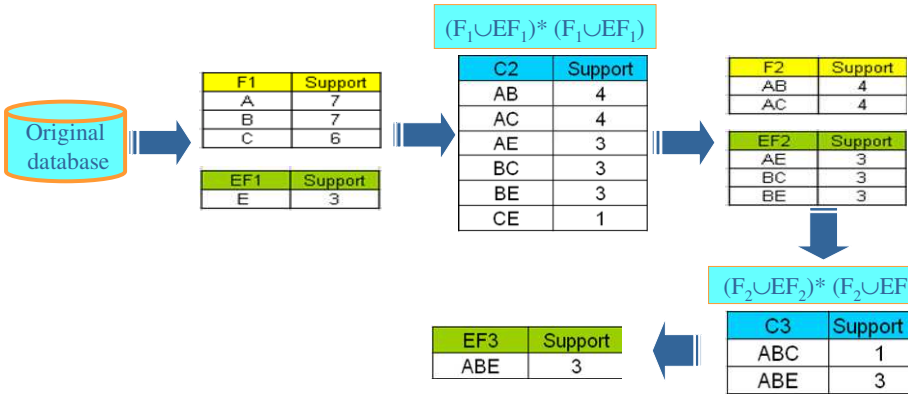

Figure 3: The example of generating candidate next k-itemsets

3.3 Updating frequent and expected frequent itemsets

When new transactions are added to an original database, an old frequent k-itemset could become an infrequent k-itemset and an old expected frequent 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.

DB	Original database
db	Increment database
UP	Updated database
k	Number of itemset
σ	Minimum support
ρ	Minimum expected frequent
C_k	Candidate k-itemset
F_k	Frequent k-itemset
EF_k	Expected frequent k-itemset

Table 1: The notation for updating frequent and expected frequent itemsets algorithm

Here, a new updating algorithm shown in figure 4 is proposed in this paper. The algorithm consists of three phases. The first phase is updating 1- frequent and expected frequent itemsets, i.e. line1-3. The second phase is repeatedly updating the other frequent and expected frequent itemsets by using only an increment database, i.e. line 6-11. The third phase is scanning an original database, i.e. line 13-16.

Figure 5 shows the algorithm for updating 1- frequent and expected frequent itemsets. 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 increment database, i.e. C_1^{db} . Then, the support count of C_1^{UP} can be updated by scanning only an increment database, i.e. line 1-4. Then, the 1-frequent and expected itemsets of an updated database can be found as shown in line 5 and 6 respectively.

Algorithm1 : Main Algorithm

$Input: DB, db, k, \sigma^{UP}, \rho^{UP}, \rho^{DB}, C_1^{DB}, F_1^{DB}, EF_1^{DB}$ *and their count*

$Output: F_k^{UP}, EF_k^{UP}$

1. $k = 1$
2. *if* $k = 1$
3. *Update* $1-itemset$
4. $k = k + 1$
5. *else*
6. *for* $\left(k = 2; F_k^{UP} \neq \phi; k++\right)$ *do*
7. *Generate Candidate Itemset*
8. *Update* $k-itemset(return\ m, Temp_scanDB)$
9. *// m is the* max*imum itemset of Temp_scanDB*
10. $k = k + 1$
11. *end do*
12. *end if*
13. $k = 2$
14. *while* $(Temp_scanDBl_k \neq \phi$ *and* $(k \leq m)$ *do*
15. *Scan Original Database*$(Temp_scanDB_k)$
16. $k = k + 1$
17. *end do*
18. *clear Temp_scanDB*

Figure 4: Main algorithm

The second phase has 2 major steps which are a generating k-increment candidate itemsets step and an updating support count of k-increment frequent and k-increment expected frequent itemsets step for k greater than or equal to 2. The algorithm for generating k- increment candidate itemsets for k greater than or equal to 2 is shown in figure 6. For k = 2, the 2-increment 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 increment database, i.e. C_k^{db} , by joining F_{k-1}^{db} with F_{k-1}^{db}, i.e. line 6. Similar to Apriori algorithm, the k-candidate itemsets of an increment database can be the

updated frequent itemsets, i.e. F_k^{UP}, only if the subsets of the k-candidate itemsets of an increment database must be in the (k-1)-updated frequent. Thus, the k- increment candidate itemsets, i.e. C_k^{new}, will keep only the k- candidate itemsets of an increment database whose subsets of the k- candidate itemsets are in the (k-1)-updated frequent itemsets, i.e. line 7-9. This can prune the k- candidate itemsets of an increment database that can't be the k- updated frequent itemsets.

Algorithm 2 : Updating 1-itemsets

Input : $DB, db, \sigma^{UP}, \rho^{UP}, C_1^{DB}, F_1^{DB}, EF_1^{DB}, C_1^{db}$ *and their count*

Output : $F_1^{UP}, EF_1^{UP}, C_1^{UP}$ *and their count*

1. *Scan db and find count* $c(X,db)$ *for all* $X \in C_1^{DB} \cup C_1^{db}$
2. *for all* $X \in C_1^{DB} \cup C_1^{db}$ *do*
3. $\quad c(X,UP) = c(X,DB) + c(X,db)$
4. *end do*
5. $F_1^{UP} = \left\{ X \in C_1^{UP} \middle| \, c(X,UP) \geq \sigma^{UP} \right\}$
6. $EF_1^{UP} = \left\{ X \in C_1^{UP} \middle| \, \rho^{UP} \leq c(X,UP) < \sigma^{UP} \right\}$

Figure 5: Update 1-itemsest algorithm

Algorithm 3: 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. $\quad$ *if* $\left(length\left(F_1^{UP}\right) \geq 2 \right)$ *then*
3. $\quad\quad C_2^{db} = F_1^{UP} * F_1^{UP}$
4. $\quad\quad$ *for all* $X \in C_2^{db}$ *do*
4. $\quad\quad\quad C_2^{new} = \left\{ X \in C_2^{db} \middle| X \notin \left(F_2^{DB} \cup EF_2^{DB} \right) \right\}$
5. $\quad\quad$ *end do*
6. $\quad$ *end if*
7. *else if* $k > 2$ *then*
8. $\quad\quad C_k^{db} = F_{k-1}^{db} * F_{k-1}^{db}$
9. $\quad\quad$ *for all* $X \in C_k^{db}$ *do*
10. $\quad\quad\quad C_k^{new} = \left\{ X \in C_k^{db} \middle| \, X \in F_{k-1}^{UP} \text{ and } X \notin \left(F_k^{DB} \cup EF_k^{DB} \right) \right\}$
11.. $\quad\quad$ *end do*
12. *end if*

Figure 6: Generating candidate k- itemsets algorithm

Figure 7 shows an algorithm for updating support count of k-updated frequent itemsets and k-updated expected frequent itemsets for k greater than or equal to 2. As shown in line 1, the algorithm scans an increment 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*- expected frequent itemsets, i.e. k-itemsets $\notin$ $F_k^{DB} \cup EF_k^{DB}$, but is in the k- increment candidate itemsets, i.e. k-itemsets $\in$ C_k^{new} , their support counts need to be specially updated. This is the case because their support counts obtained from an original database are not available. Since these k-itemsets are not in $F_k^{DB} \cup EF_k^{DB}$, their support counts are at best equal to ρ^{DB} - 1 . Here, their support counts are assumed to be equal to the sum of ρ^{DB} - 1 and their support counts obtained from an increment 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 set aside for finding their true support counts from an original database, i.e. line 9.

Algorithm 4 : Update (k ≥ 2) itemset

Input: $DB, db, \sigma^{UP}, \rho^{UP}, \rho^{DB}, F_k^{DB}, EF_k^{DB}$ *and their count*

Ouiput: F_k^{UP} *and* EF_k^{UP}, F_k^{db}, *Temp_ scanDB and their count, m*

1. *Scan db and find count* $c(X, db)$ *and* $c(Y, db)$
2. \\ $X \in (F_k^{DB} \cup EF_k^{DB})$ *and* $Y \in C_k^{new}$
3. *for all* $X \in (F_k^{DB} \cup EF_k^{DB} \cup C_k^{new})$ *do*
4. *if* $X \in (F_k^{DB} \cup EF_k^{DB})$ *and* $X \in C_k^{new}$ *then*
5. $c(X,UP) = c(X,DB) + c(X,db)$
6. *else if* $X \in (F_k^{DB} \cup EF_k^{DB})$ *and* $X \notin C_k^{new}$ *then*
7. $c(X,UP) = c(X,DB)$
8. *else if* $X \notin (F_k^{DB} \cup EF_k^{DB})$ *and* $X \in C_k^{new}$ *then*
9. $Temp_scanDB_k = \{ X \mid (c(X,db) + (\rho^{DB} - 1)) \geq \sigma^{UP} \}$
10. *end if*
11. *end do*
12. $F_k^{UP} = \{ X \mid c(X,UP) \geq \sigma^{UP} \}$
13. $EF_k^{UP} = \{ X \mid \rho^{UP} \leq c(X,UP) < \sigma^{UP} \}$

Figure 7: Update (k ≥ 2) itemset algorithm

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 8. 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. Then, all k-updated frequent itemsets and k- updated expected frequent itemsets are found as shown in line 7-8.

The figure 9 shows the increment database and the figure 10 shows the example of three phases of updating frequent and expected frequent itemsets. In the first phase, the 1-frequent and expected frequent itemset can be updated by scanning only an increment database. In the second phase, the 2-updated candidate itemset is generated by joining frequent 1-itemset of updated database together. The new candidate 2-itemset is pruned if it is member of frequent and expected frequent itemset of an original database. In this case, set of {AD, BD, CD} are new candidate 2-itemset in the updated database.

The increment database is scanned for 2-itemset of frequent, expected frequent and new updated candidate. Only new candidate itemset that has support count greater than or equal to minimum support of increment database, it require to scan in the original database.

Algorithm 5 : Scanning an original database

Input : $Temp_scanDB_k, \sigma^{UP}, \rho^{UP}, F_k^{UP}, EF_k^{UP}$ *and their count*

Output : F_k^{UP}, EF_k^{UP} *and their count*

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

Figure 8: Algorithm for scanning an original database

Increment Database	
TID	List of item
11	A, B, D, E
12	B, C, D
13	B, D, E
14	A, B, C, D
15	A, C

Figure 9: Increment database

To reduce the number of itemset for scanning original database, if sum of any new candidate's support count and support of $prob_{pl}$ minus 1 is greater than minimum support of updated database, then it will be moved to Temp_scanDB. In this example, only set of BD is moved to Temp_scanDB. The last phase, original database is scanned for finding true support of BD. Finally, the frequent and expected frequent k-itemsets for updated database are found.

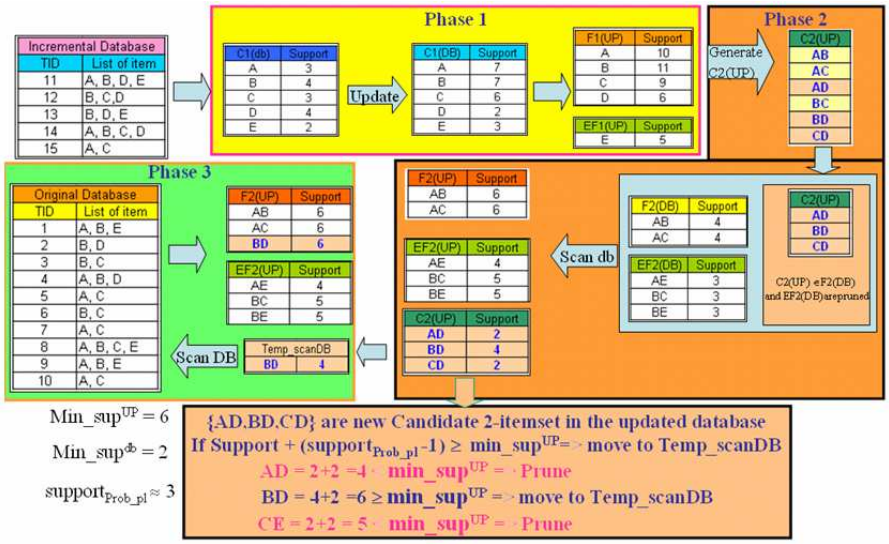

Figure 10: Example of three phases of updating frequent and expected frequent itemsets

4 Experiment

The purpose of this research is to explore an alternative way to improve the performance of the association rule mining while facing dynamic database environments. To evaluate the performance of probability-based incremental association rules discovery 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 T10I4D20K. The technique for generating the dataset is proposed by Agrawal [Agrawal 1994]. The synthetic dataset comprises 250,000 transactions over 70 unique items. Each transaction of the synthetic dataset has 10 items on average and the maximal size itemset is 4.

Firstly, we show the performance of our algorithm with minimum support 3% and 4%. The proposed algorithm with $Prob_{pl} = 0.06$ is used to find frequent itemsets, expected frequent itemsets and association rules from an original database of 20,000 transactions. Then, several sizes of increment databases, i.e. 25%, 50%, 75% and 100% of the original database, are added to the original database. Then, the proposed algorithm is used to maintain frequent itemsets, expected frequent itemsets and association rules of the updated database. For comparison purpose, FUP, Borders and Pre-large algorithm are also used to find frequent itemsets, expected frequent itemsets and association rules from the same original database and the same increment databases.

The experimental results with various minimum support thresholds are shown in figure 11, table 2 and table 3. Table 2 and table 3 show the number of frequent k-itemsets and k- expected itemsets in an original database for FUP, Borders, Pre-large algorithm and the proposed algorithm. Since these algorithms are in the second approach which is assumed that association rules to be found are stable over time, the association rules or frequent itemsets should be the same as those obtained from re-runing Apriori algorithm. Thus, each frequent itemset obtained from these algorithms is compared with that obtained from re-runing Apriori algorithm. The experiments show that all frequent itemset obtained from FUP, Borders, Pre-large algorithm and the proposed algorithm are the same as those obtained from re-runing Apriori algorithm.

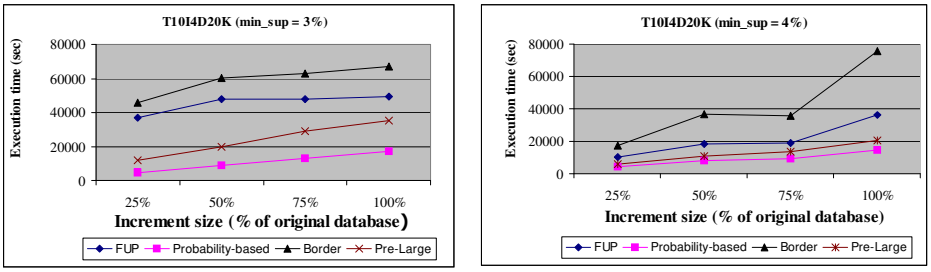

Figure 11: Execution time in FUP, Borders and Probability-based when (a)min_sup =3% and (b) min_sup =4%

Original Database : Min_sup = 3%										
k-item	Apriori		FUP		Borders		Pre-Large		Probability-based	
	Number of Frequent itemset	Number of kept Infrequent itemset	Number of Frequent itemset	Number of kept Infrequent itemset	Number of Frequent itemset	Number of kept Infrequent itemset	Number of Frequent itemset	Number of kept Infrequent itemset	Number of Frequent itemset	Number of kept Infrequent itemset
k=1	68	0	68	0	68	1	68	1	68	0
k=2	700	0	700	0	700	1578	700	418	700	14
k=3	158	0	158	0	158	5003	158	399	158	15
k=4	39	0	39	0	39	44	39	80	39	1
k=5	21	0	21	0	21	0	21	10	21	0
k=6	7	0	7	0	7	0	7	1	7	0
k=7	1	0	1	0	1	0	1	0	1	0

Table 2: Number of itemset of an original database for each algorithm with min_sup=3%

Original Database : Min_sup = 4%										
k-item	Apriori		FUP		Borders		Pre-Large		Probability-based	
	Number of Frequent itemset	Number of kept Infrequent itemset	Number of Frequent itemset	Number of kept Infrequent itemset	Number of Frequent itemset	Number of kept Infrequent itemset	Number of Frequent itemset	Number of kept Infrequent itemset	Number of Frequent itemset	Number of kept Infrequent itemset
k=1	65	0	65	0	65	4	65	3	65	1
k=2	442	0	442	0	442	1638	442	258	442	25
k=3	24	0	24	0	24	2160	24	134	24	3

Table 3: Number of itemset of an original database for each Algorithm with min_sup = 4%

From both table 2 and table 3, the results also show that the proposed technique requires slightly more space to keep the k- expected itemsets than that of FUP. This is the case because FUP does not need to keep some infrequent itemsets to maintain updated frequent itemsets but FUP requires multiple-scan an original database to maintain updated frequent itemsets. Thus, FUP requires more times for scanning an original database than that of the proposed algorithm. As a result, the proposed algorithm has better running time than that of FUP. As shown in figure 11, the proposed requires lesser memory space to keep the k- expected itemsets than that of Borders algorithm and Pre-large algorithm. However, the algorithm still has better running time than that of Borders algorithm and Pre-large algorithm. This is the case because the proposed algorithm, which is based on the principle of Bernoulli trials, has more efficient to predict expected frequent itemsets than Borders algorithm and Pre-large algorithm. Thus, the proposed algorithm requires less times for scanning an original database than Borders algorithm and Pre-large algorithm. As a result, the proposed algorithm has better running time than that of Borders algorithm and Pre-large algorithm.

	Average of Execution time for 100 trials (sec)			
min_sup (%)	**FUP**	**Borders**	**Pre-Large**	**Probability-Based**
4%	5900.722	7588.617	4207.582	2661.803

Table 4: The average execution time for FUP, Borders and Probability-based

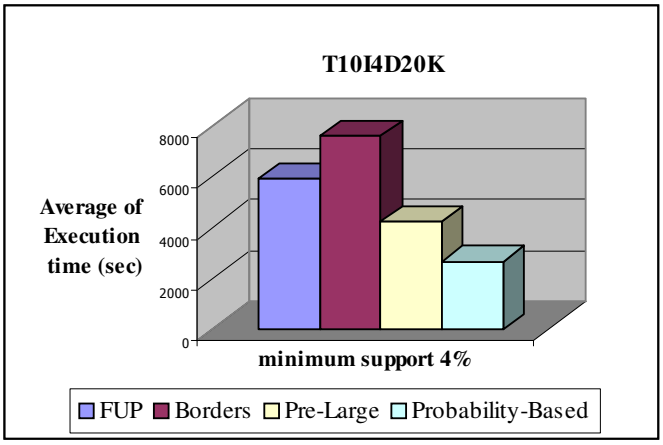

Figure 12: The Average of execution time in FUP, Borders and Probability-based when min_sup =4%

Secondly, the proposed algorithm with $Prob_{pl}$ = 0.06 used to find frequent itemsets, expected frequent itemsets and association rules from an original database of 20,000 transactions. Then, the same sizes of increment databases, i.e. 10% of the original database, are added to the original database for 100 trials. For comparison purpose, FUP, Borders and Pre-large algorithm are also used to find association rules from the same original database and the same increment databases. Table 4 and figure 12 show the average execution time of FUP, Borders, Pre-large and the proposed algorithm. The results show that the proposed algorithm has much better running time than that of the other algorithms.

5 Conclusion

To create an intelligent environment such that new information can be discovered in a dynamic database, this paper proposes mining a dynamic database using probability-based incremental association rule discovery algorithm. The proposed algorithm uses the principle of Bernoulli trials to find expected frequent itemsets. This can reduce a number of times to scan an original database. Assuming that the minimum support and confidence do not change, the algorithm can maintain association rules for a dynamic database. The experiments show that our algorithm has better running time than that of FUP, Borders and Prelarge algorithm.

References

[Adnan et al. 2005] 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, September 16-19, 2005, 129-134.

[Agrawal et al. 1993] 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, 207-216.

[Agrawal, Srikant 1994] 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.

[Alon et al. 1992] Alon, N. and Spencer, J. H., 1992, "The probabilistic method"; Wiley Interscience, New York.

[Amornchewin, Kreesuradej 2007] Amornchewin, R., and Kreesuradej, W.: "Incremental association rule mining using promising frequent itemset algorithm "; In Proceeding 6th International Conference on Information, Communications and Signal Processing, Singapore, 2007.

[Amornchewin, Kreesuradej 2008] Amornchewin, R., and Kreesuradej, W.: "Probability-based incremental association rule discovery algorithm"; The 2008 International Symposium on Computer Science and its Applications (CSA-08), Australia, 2008.

[Ayan et al. 1999] 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.

[Brin et al. 1997] Brin, S., Motwani, R., and Silverstein, C.: "Beyond Market Baskets: Generalizing Association Rules to Correlations"; Proceedings of the ACM SIGMOD Conference, 1997, 265-276.

[Chang, Yang 2003] Chang, C.-H. and Yang, S.-H.: "Enhancing SWF for Incremental Assocaition Mining by Itemset Maintenance"; Proceedings of 7th Pacific-Asia Conference on Knowledge Discovery and Data Mining, April 2003.

[Chang et al. 2005] 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.

[Cheung et al. 1996] 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.

[Cheung et al. 1997] 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.

[Du et al. 2008] 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.

[Dudek, Zgrzywa 2005] Dudek, D., Zgrzywa, A.:"The Incremental Method for Discovery of Association Rules"; Springer Berlin / Heidelberg, volume 30, 2005, 153-160.

[Elfangary, Atteya 2008] Elfangary, L.,Atteya, W.A.: "Mining Medical Database using Proposed Incremental Association rule Algorithm (PIA)"; Second International Conference on the Digital Society, IEEE (2008).

[Feldman et al. 1999] 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.

[Ezeife, Su 2002] Ezeife, E. I. and Su, Y.: "Mining Incremental Association Rules with Generalized FP-Tree"; Proceedings of the 15th Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Argificial Intelligence, May 2002, 147-160.

[Gong 2008] Gong M. :"Personalized E-learning System by Using Intelligent Algorithm"; 2008 Workshop on Knowledge Discovery and Data Mining, IEEE, 2008.

[Guo, Zhao 2008]Guo, M., Zhao Y.: "An Extensible Architecture for Personalized Information Services in An Ambient Intelligence Environment"; IEEE 2008.

[Han et al. 2000] Han, J.,Pei, J. and Yin, Y.: "Mining Frequent Pattern without Candidate Generation";Proceedings of the 2000 ACM-SIGMOD International Conference on Management of Data, May 2000, 355-359.

[Hong et al. 2001] 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.

[Juan et al. 2008] Juan X., Feng Y., Zhiyong Z.: "Association Rule Mining and Application in Intelligent Transportation System"; Proceedings of the 27th Chinese Control Conference, China, 2008, 538-540.

[Lee et al. 2001] 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.

[Liu et al. 1999] 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.

[Luhr et al. 2005] Luhr, S., Venkatesh, S., and West, G.: "Emergent Intertransaction Association Rules for Abnormality Detection in Intelligent Environments"; Proceedings of the 2005 International Conference on Intelligent Sensors, Sensor Networks and Information Processing Conference, December 2005, 343-347.

[Manku, Motwani 2002] Manku, G.S., and Motwani, R.: "Approximate Frequency Counts over Streaming Data"; Proceedings of the 28th International Conference on Very Large Database, August 2002, 346-357.

[Mannila et al. 1994]Mannila, H., Toivonen, H., and A. Verkamo I.: "Efficient algorithms for discovering association rules. In Knowledge Discovery in Databases"; 1994 AAAI Workshop (KDD'94), Seattle, Washington, July 1994, 181- 192.

[Pawar, Aggarwal 2004] Pawar P.A.and 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.

[Sarda, Srinivas 1998] 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.

[Savasere et al. 1995] Savasere, A., Omiecinski, E., and Navathe S.: "An Efficient Algorithm for Mining Association Rules in large database"; Proceedings of the 21st VLDB Conference, Zurich, Switzerland, 1995.

[Teng, Chen 2005] Teng W.-G. and Chen, M.-S.: " Incremental Mining on Association Rules"; Springer Berlin/ Heidelberg, vol. 180, 2005, 125-162.

[Thomas et al. 1997] 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.

[Thuraisingham, Ceruti 2000] 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.

[Tsai et al 2005] Tsai, Paury S.M.,Lee, Chih-Chong, and Chen, Arbee L.P. : "An Efficient Approach for Incremental Association Rule Mining"; Proceedings of the Third Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining, Lecture Notes In Computer Science; Vol. 1574 archive, 1999.

[Toivonen 1996] 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.

[Veloso et al. 2002] Veloso, A. Meira Jr., W., Carvalho, M., Parthasarathy, S., and Zaki ,M. J.: "Parallel, Mining frequent itemsets in evolving Databases"; In Proc. 2nd SIAM Intl. Conf. on Data Mining, Arlington, VA, April 2002.

[Zhang et al. 2003] Zhang, C., Zhang, S., and Webb, G. I.: "Identifying Approximate Itemsets of Interest in Large Databases"; Applied Intelligence 2003, 91–104.

[Zhang et al. 2003] Zhang, S., Zhang, C., and Yan X.: "Post-mining: maintenance of association rules by weighting"; Information System, 2003, 691-707.