ISSN: 2321-7782 (Online)
Volume 3, Issue 2, February 2015
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Probability-based Incremental Association Rules Algorithm Using Hashing Technique

Jhanvi V. Kothari[1]
Research scholar, Department of C.E
School of Engineering, R.K. University,
Rajkot - India

Dr. K. M. Patel[2]
Assoc. Professor, Department of C.E
School of Engineering, R.K. University,
Rajkot - India

Abstract: Association Rule is most intriguing area of Data Mining which uses Apriori algorithm. In a dynamic Database where new transactions are added to original database for that we need new association rule and existing will be invalid. In study of incremental association rule, rescan original database several times is inefficient. Next, apply probability rules with incremental association rule. Probability based algorithm is based on Bernoulli trials. This rule will find frequent and expected frequent Itemset from candidate itemset. Generating and testing candidate itemset is time consuming. Candidate 2-itemset need to rescan original database and check a large set of Item, we can reduce the number of candidate by using hash technique for the generation of candidate 2-itemset, particularly for frequent and expected frequent 2-itemset. This technique will improve the performance of probability. The Algorithm can reduce not only time of rescanning database but also number of candidate itemset.

Keywords: Data mining, Incremental Association Rule Discovery, Hashing.

I. INTRODUCTION

Data mining has technique and tools for intelligently transforming the data into useful information and knowledge. Data Mining is one of the processes of Knowledge Discovery in Database (KDD) that is used for extracting information or pattern from large Database. Association Rule Mining is major area of Data mining. Association rule discover using two parameter i.e. 1) support ($A \Rightarrow B$) = $P(A \cup B)$ 2) Confidence ($A \Rightarrow B$) = $P(B|A)$. It is an implication of the form X=>Y(s,c) where X and Y are frequent itemset. Apriori is algorithm for association rule mining.

In Dynamic Database when new transaction are inserted it is challenging problem. The updating database may cause not only the generation of new rules but also invalidation of existing rules. Apriori algorithm is mining all new transaction of updating database when database has been changed.

The mining of association rules on transactional database is usually an offline process since it is costly to find the association rules in large databases. With usual market-basket applications, new transactions are generated and old transactions may be no longer in use as time advances [13]. As a result, incremental updating techniques should be developed for maintenance of the discovered association rules to avoid redoing mining on the whole updated database [13]. A database may allow frequent or occasional updates and such updates may not only invalidate existing association rules but also activate new rules. Thus it is nontrivial to maintain such discovered rules in large databases.

To rescanning a frequent database of original database is inefficient because it already done in previous scan. The several research work [1,2,3,4,5,6,7] have proposed several incremental algorithm to deal with this problem.

Probability based algorithm used Bernoulli trials. This rule is used to find all frequent itemset and expected frequent itemset of an updated database efficiently [6]. When new transactions are added to original database, expected frequent itemset is capable to be frequent itemset. We modify some rules of probability and incremental association.

In addition, candidate itemsets are very large when pattern is too long. So, the running time of large database algorithm would be long. Hash technique is reduced time of execution by reducing size of the candidate k-itemset(especially when k=2). The issue of work is use hash technique for the generation of candidate 2-itemset, specifically for 2-itemsets to improve performance of probability based algorithm. The hashing based algorithm reduced number time scan original database and also reduce number of candidate itemset to generate frequent 2 itemset. So, this execution time may faster than others.

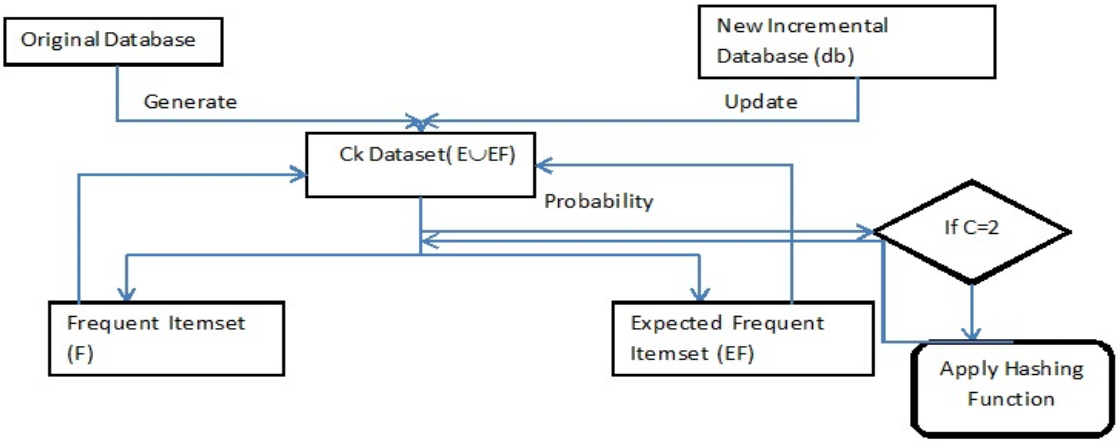

Figure 1 shows Flow of Algorithm.

II. Related Work

In 1993, Agrawal [1] first defined the association rules mining in databases. They considered the example following example; where transactions in which bread is purchased are also transactions in which milk is purchased. Let $J = \{j1, j2, \ldots, jm\}$ represent the set of literals, called *items* [1]. The symbol T represents an arbitrary transaction, which is a set of items (*itemset*) such that $T \subseteq J$. All itemset has a specific identifier, *TID*. Let *DB* be a database of transactions. Assume A is an itemset; a transaction T contains A if and only if $A \subseteq T$ [1].

An association rule applies in the form $A \Rightarrow B$, where $A \subseteq J$, $B \subseteq J$ and $A \cap B = \phi$(For example, $J = \{UVXYZ\}$, $A = \{UX\}$, $B = \{VZ\}$). An association rule A B has two properties, *support* and *confidence*. When s% of transactions in *DB* contain $A \cup B$, the support of the rule $A \Rightarrow B$ is s%. If some of the transactions in *DB* contain A and, and c% also contain Y, then the confidence in the rule $A \Rightarrow B$ is c%.

In general, the confidence is expressed in the form confidence($A \Rightarrow B$) = support($A \cup B$)/support(A). Given the user assigned minimum support (*minSup*) and minimum confidence (*minConf*) thresholds for the transaction database *DB*, association rules is to find frequent itemset whose support and the confidence are greater than the two respective minimum thresholds [5]. When its support is no less than the *minSup* threshold; otherwise, it is an *infrequent itemset* [5].

2.1 Apriori Algorithm

Apriori[1] is a classic algorithm for learning association rules. Apriori is designed to operate on databases containing transactions[5]. The algorithm relies on the fact that an itemset could be frequent only when each of its subset is frequent; otherwise, the itemset is infrequent. In the first scan, the Apriori algorithm constructs and counts support of all 1-itemsets[5]. (A k-itemset is an itemset that includes k items.) After we get all frequent 1-itemsets, the algorithm joins the frequent 1-itemsets with each other to generate candidate 2-itemsets. Apriori scans the transaction database and counts the candidate 2-itemsets to determine which of the 2-itemsets are frequent. The other passes are made accordingly. Frequent (k - 1)-itemsets are joined to form k-itemsets whose first k-1 items are identical. If $k \geq 3$, Apriori prunes some of the k-itemsets; of these, (k –1)-itemsets

have at least one infrequent subset[5]. All remaining *k*-itemsets constitute candidate *k*-itemsets. The process is reiterated until no more candidates can be generated.

2.2 FUP

One of the previous works for incremental association rule mining is FUP [2] algorithm. FUP first scans the incremental part of the dataset and detects (*i*) the looser single itemsets, i.e. the itemsets that become infrequent due to the inclusion of the incremented part and (*ii*) it finds the candidate frequent itemsets [2]. Then the whole dataset (i.e. the old and new together) is scanned to find their support in the complete dataset. Next, it performs similar operations iteratively for *k*-itemsets. Finally, after multiple scanning of the dataset it finds all the maximal frequent sets. 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.

TABLE I FUP and its Result [14]

Case:Original-Incremental	Result
Case1:Large-Large	Always Large
Case2:Large-Small	Determined from existing information
Case3:Small-Large	Determined by rescanning the original Database
Case4:Small-Small	Always small

Advantage: FUP is faster than re-runing apriori for newly added transactions. The FUP algorithm reuses information from old frequent itemsets to improve its performance.

Disadvantage:1. FUP algorithm requires to scan passes over an original database several times when new frequent item sets are found. 2. It only works for insertion of data in transaction, it can not work with deletion.

2.3 FUP2 and UWEP

The extension algorithm of FUP is FUP2 [3] that is proposed to handle all update cases when database are added to, deleted from a database. Ayan et al [4] present an algorithm called UWEP (Update With Early Pruning). UWEP follows the approaches of FUP and partition algorithm. It employs a dynamic look-ahead strategy in updating existing large itemsets by detecting and pruning superset of large itemsets in an original database that will no longer remain large in updated database. UWEP scans at most once in both original database and incremental database. UWEP generates smaller candidate set from the set of itemsets that are large both an original and incremental database.

Advantage: It works on incremental and Decrement database both.

Disadvantage: FUP2 algorithm requires to scan passes over an original database several times when new frequent item sets are found

2.4 Negative Border

Negative Border algorithm (NBd) [4] was proposed to reduce the number of rescanning times of an original database by collecting both frequent itemsets and border itemsets (itemset which is not frequent itemsets but its proper subsets are frequent itemset). This algorithm is successful for reducing the number of rescanning times but a large number of border itemsets have to collect. Thus this Negative Border consumes a large amount of memory. Moreover, in the worst case, Negative Border algorithm needs to rescan an original database several times when new frequent itemsets are discovered in an updated database.

Advantage: It Reduce rescanning Original database several times.

Disadvantage: Unnecessary Candidate Generation, Large number of border set.

2.5 NFUP

To mine new interesting rules in updated database, NFUP partitions the incremental database logically according to unit time interval [5]. For each item, assume that the ending time of exhibition period is identical. NFUP progressively accumulates the occurrence count of each candidate according to the partitioning characteristics[5]. The latest information is at the last partition of incremental database. Therefore, NFUP scans each partition backward, namely, the last partition is scanned first and the first partition is scanned last[5].

Advantage: NFUP does not require the rescanning of the original database to detect new frequent

itemsets or delete invalidate itemsets.

Disadvantage: The running time of NFUP rises almost in direct proportion with the transaction number of the incremental database.

2.6 Promising Frequent Itemset Algorithm

In this paper the new idea is to avoid scanning the original database. Then it computes not only frequent itemset but also compute itemset that may be potentially large in an incremental database called “Promising frequent Itemset” [9]. An algorithm is find all possible k-itemset of promising frequent itemset in original database. If member of frequent itemset for each iteration is more than or equal to k-itemset then it will be in promising frequent itemset [9]. This idea is guarantee that promising frequent itemset algorithm are cover all frequent itemset that occur in updated database [9]. Thus, updating the new transactions are quickly because it can use the information from the existing original database.

Advantage:scan the original database only once

Disadvantage: incremental datset size is fixed.

2.7 EIRM:

An Efficient Incremental Rule Mining (EIRM) [13] algorithm is proposed to speed-up the process of frequent – pattern mining. By storing the Transaction Identifiers (TIDs) of itemsets and exactly calculating support count helps to reduce the required scan iterations to a database. To avoid the problem of multiple scans and to improve performance, the EIRM (Efficient Incremental Rule Mining Algorithm) is proposed in this paper, so the dataset need to be scanned only once[13]. In the proposed algorithm[13], each transaction has their unique Transaction identifier (TID).By using the hash function concept, to store TIDs in a table structure, helps to calculate the number of itemsets quickly without the need of re-scanning the dataset. The algorithm works as 2 subsections. An original dataset is firstly mined and all promising and unpromising itemsets are found. Secondly, the incremental dataset in mined and updated to promising and unpromising itemsets. As the result of updation, some unpromising itemsets or new itemsets may be changed into promsing itemset[13].

2.8 probability based algorithm

Hong et al and Amornchewin et al proposed a new algorithm which reduce memory. The algorithm maintains both frequent itemsets and expected frequent itemsets. 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. 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.

Advantages: It reduce memory consumption, maintain both frequent itemset and expected frequent itemset.

Disadvantage: It allows small size of an incremental database to insert in to 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, Probability-based Incremental Association Rule Discovery Algorithm, is introduced by Amornchewin and Kreesuradej [5]. Similar to the previous approach, the new algorithm also keeps both frequent itemsets and expected frequent itemsets. The Probability-based algorithm uses the principle of Bernoulli trials to estimate expected frequent itemsets. This technique can allow larger size of an increment database to insert into an original database than that of the previous approach.

III. USING PROBABILITY BASED INCREMENTAL ASSOCIATION RULE

In Probability based incremental association rule algorithm, insert m transaction to n original database. Each itemset of candidate set has its probability which is denoted by P. Transaction of (n+m) probability is denoted by P(X), it can be founded by equation,

$$P(x \geq K)^{kitems} = \sum_{x=0}^{K-1} \binom{n+m}{x} p^x (1-p^{n+m-x})$$

Where K= Minimum Support after inserting new database

n= Original Database

m=Incremental Database

Here, an expected frequent itemset is not a frequent itemset but if its probability will be greater than $Prob_{pl}$ then it will be a frequent itemset. Probpl is a constant threshold which is specified by users. $Prob_{pl}$ indicates the minimum confidence level that a expected frequent itemset will be a frequent itemset after inserting new transaction into an original database. The higher $Prob_{pl}$ is set, the lesser expected frequent itemsets are kept. As results, Number of rescanning time would be more in original database when the algorithm performs the discovering new frequent iteset task.

The algorithm use support count of itemset and total transaction of original databse.

$$p = \frac{C(Itemset,DB)}{|DB|}$$

Where, c(itemset,DB) is support count of itemset of original database

IV. HASHING TECHNIQUES

Here Paper [7] use hashing technique to 2-itemset. In hashing technique they took one hash formula which is based on 2-itemset. This formula is used to store itemset in different buckets, number of bucket is fixed but size of bucket is not fixed its unlimited. There is bucket count which store number of items count in bucket count. The itemsset which has 0 counts it will not store in any bucket it will be discard. As, result we can get itemset which is frequent n expected frequent easily, plus reduce size of candidate k-itemset(K=2). In paper by Ratchadaporn Amarchewin [7] describe hashing technique compare with other technique, in result execution time using hashing technique lesser then all other technique. But it removes candidate itemset which has less support count then minimum support.

The notation for updating frequent and expected frequent algorithm:

TABLE II Notation

DB	Original Database

db	Incremental database
UP	Updated Database
k	Number of Itemset
σ	Minimum Support
α	Minimum Expected Frequent
C_k	Candidate k-itemset
F_k	Frequent k-itemset
EF_k	Expected frequent k-itemset
P	Probability
$Prob_{pl}$	Thresold Constant

Incremental Rule Mining Algorithm:

Input: DB,db,k,$Prob_{pl}$,σ^{DB},α^{db}

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

Step1: Scan DB & generate C_k^{DB}

Step2: C(x,DB) $F_k^{DB}=\{x€C_k^{DB}|C(x,DB)\geq\sigma^{DB}\}$

Step3: Find P_k for all itemse of C_k^{DB}

Step4: $EF_k^{DB}=\{x€C_k^{DB} \mid Prob_{pl}<P_{If}<\sigma^{DB}\}$

Step5: Generate next (k+1) candidate itemset of $(F_k^{DB} \cup EF_k^{DB})$

Step6: Repeat step 2 to 5 till you get most frequent items.

Step7: For k=2, k=k+1 Generate Hash Candidate store items in to bucket

For db

Step8: Scan db and generate C_k^{db}

Step9: Update C_k^{db} while $C_k^{UP}= C_k^{DB} + C_k^{db}$

Step10: Update F_k^{DB} and EF_k^{DB} , find p for C_k^{UP}

$F_k^{UP} =\{x|C_k^{UP} \mid C(x,UP) \geq\alpha^{UP}\}$

$EF_K^{UP}=\{x|C_K^{UP}| Prob_{pl} < P_{IF}<\alpha^{UP}\}$

Step11: Generate next K+1 itemset of $(F_K^{UP} \cup EF_K^{UP})$

Step12: Repeate step 9 to 11 till you get most frequent itemset

Step13: Update Hash Table (Itemset 2)

Figure 2 Algorithm

V. CONCLUSION

In this paper we described all incremental association rule algorithm. In which probability based incremental rule discovery by using hashing technique has lowest execution time. Probability allows both frequent and expected frequent dataset and also allows using large size of database. Thus the hash technique might reduce size of candidate itemset and execution time of algorithm.

References

1. R. Agrawal., T. Imielinski, and A. Swami, “Mining association rules between sets of items in large databases”, In Proceeding of the ACM SIGMOD Int'l Conf. on Management of Data (A CM SIGMOD '93),Washington, USA,May 1993, pp. 207-216.
2. D. Cheung, J. Han, V. Ng, and C. Y. Wong, “Maintenance ofdiscovered association rules in large databases: An incremental updating technique”, In 12th IEEE International Conference on Data Engineering, February 1996, pp. 106-114.
3. D. W. Cheung, S.D. Lee, B. Kao, “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, pp. 185-194.
4. R. Feldman, Y. Aumann, and O. Lipshtat, “Borders: An efficient algorithm for association generation in dynamic databases”,Journal,Intelligent Information System, 1990, pp. 61-73.
5. C.C. Chang, Y.C. Li and J.S. Lee, “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.
6. R. Amornchewin and W. Kreesuradej, “Probability-based incremental association rule discovery algorithm”; The 2008 International Symposium on Computer Science and its Applications (CSA-08),Australia (2008).
7. Ratchadaporn Amornchewin,” Probability-based Incremental Association Rules Discovery Algorithm with Hashing Technique”, International Journal of Machine Learning and Computing, Vol.1, No. 1, April 2011.
8. Araya Ariya,Worapoj Kreesuradej, “Probability Based Incremental Asoociation Rule Discovery Using the Normal Approximation”, IEEE IRI 2013,August 14-16 2013,San Francisco, California,USA978-1-4799-1050-2/13/31.00 c 2013 IEEE.
9. R. Amornchewin and W. Kreesuradej, “Incremental association rule mining using promising frequent itemset algorithm”, In Proceeding 6th International Conference on Information, Communications and Signal Processing, Dec. 10-13 2007, pp.1-5.
10. C. H. Lee, C. R. Lin, and M. S. Chen, “Sliding-Window Filtering: An Efficient Algorithm for Incremental Mining” , Proceeding of the ACM 10th International Conference on Information and Knowledge Management ,November 2001.
11. R. Amornchewin and W. Kreesuradej, “False Positive Itemset Algorithm for Incremental Association Rule Discovery” ,International Journal of Multimedia and Ubiquitous Engineering. Vol. 4, No. 2,April,2009.
12. H Toivonen, “ Sampling large database for association rules,” In Proceedings of the 22th International Conference on Very Large Database (VLDB’96), September 1996, pp. 134-145.
13. Kavitha J.K , Manjula D and Kasthuri Bha J.K, “Effective and Efficient rule Mining Technique for Incremental Dataset” Journal of Theoretical and Applied Information Technology,Vol.57,No 3,30 November 2013.

AUTHOR(S) PROFILE

Jhanvi V. Kothari Received the B.E. degree in Information Technology from Gujarat Technology University, Gujarat, India, in June 2013 and Pursuing Master of Technology degree in Computer Engineering from RK University, Gujarat, India. Her main area of interest includes Database Management System, Data Structure and Data Mining.

Dr. K. M. Patel received the B.E. Computer Engineering degree from the Sardar Patel University, in 1998 and the M.E. Computer Engineering from the Sardar Patel University, in 2007, and Ph.D. from Monad University, in 2013. During the course of his M.E and Ph.D. he has authored a more than 10 different journal and conference papers. He is currently working as Associate Professor in computer engineering and information technology at school of engineering, RK University. His research interests are Data Mining, Web Mining and SEO.

