**Available Online at www.ijcsmc.com**

**International Journal of Computer Science and Mobile Computing**

**A Monthly Journal of Computer Science and Information Technology**

**ISSN 2320–088X**

***IJCSMC, Vol. 3, Issue. 5, May 2014, pg.923 – 929***

**RESEARCH ARTICLE**

# Discovery of Frequent Itemset and Promising Frequent Itemset Using Incremental Association Rule Mining Over Stream Data Mining

**Anju Kakkad[1], Anita Zala[2]**

[1]Computer KSV University, India

[2]Computer KSV University, India

[1] anju.kakkad@gmail.com; [2] ani.zala@gmail.com

***Abstract*— Data mining is a technique to extract interesting knowledge from large data sets. One of the most used techniques in data mining is association rule mining technique which discovers rule. Discovering rule is one of the most interesting are of research in data mining. Now a days databases are became as the time goes in database new transactions are added and some old transaction may be deleted. For this traditional association rule mining is not working so that incremental association rule mining techniques come in scenario. As the time goes new rules may be generated and old rule may be obsolete. In this paper I have proposed new technique for finding rule with incremental association rule mining to provide faster execution without rescanning of original database.**

***Keywords*— *incremental association rule mining; promising frequent itemset; dynamic database***

## I. INTRODUCTION

Data mining has a great emphasis in real world applications. Day by day database size is increasing due to increasing use of large data requires high computation for various applications so that the importance of data mining has grown rapidly. One of the major applications of data mining is association rule mining technique. Association rule mining is a technique which finds correlation between two item sets. For the static database traditional association rule mining can work but for dynamic database traditional association rule mining is a tradeoffs, for dynamic database incremental association rule mining technique can be used [4] It is a technique in which after some period of time new transaction are added so new rule must be calculated and it may be possible that some old rule may be obsolete.

We can discover rule using two parameter i.e. 1) support 2) confidence. An association rule is an implication of the form $X \Rightarrow Y$ (s, c), where X and Y are frequent itemsets in a transactional database and $X \cap Y = \varnothing$, s is the percentage of records that contain both X and Y in the database, called support of the rule, and c is the percentage of records containing X that also contain Y, called the confidence of the rule [1]. Association rule mining is to find all association rules the support and confidence of which are above or equal to a user-specified minimum support and confidence, respectively. Using support count we get frequent itesem. Frequent itemset are those itemset which are frequently appearing in dataset.

Stream data mining is an ordered sequence of instances which finds knowledge from unbounded, continues and rapid data [7]. Stream data mining can be classified in two types one is online stream and another is offline stream. Example of offline stream is generating reports based on web log reports another example is queries on updating warehouse. Examples of online stream are that data which come one by one i.e. network packet, sensor data. Bulk data processing is not possible in online data stream [5]

## II. DATA MINING PRELIMINARIES

Let I= (i1, i2 ... in) be a set of literals, referred to as items. Each transaction T contains a set of items (i1, i2 ... ik) ⊂ I. We will refer to a set of items as an itemset. The number of items in the itemset is the length of the itemset. Therefore, an itemset of length k is a k-itemset [3]. The association rule mining problem is to find out all the rules in the form of X=> Y, where and Y ⊂ I are sets of items, called itemsets. The association rule discovery algorithm is usually decomposed into 2 major steps. The first step is find out all large itemsets that have support value exceed a minimum support threshold and the second steps is find out all the association rules that have value exceed a minimum confidence threshold [6]. Frequent itemset are itemsets having frequent occurrences (transaction) in datasets. Promising frequent itemsets are set of non frequent itemset having near future to be frequent. The task is that what is to be done after extracting these so many transactions and how to summarize only the helpful transaction in the single database with existing and upcoming transaction.

New incremental algorithm, called promising frequent itemset algorithm, is introduced. The goal of this work is to solve the efficient updating problem of association rules after a nontrivial number of new records have been added to a database. This new approach is to introduce a promising frequent itemset for an infrequent itemset that has capable of being a frequent itemset after a number of new records have been added to a database. This can reduce a number of times to scan an original database [6].

## III. RELATED WORK

In the incremental mining, data are not only added but also obsolete data are being deleted. The main aim of incremental mining algorithm is to re-run the mining algorithm on the only incremented database. However, it is obviously less efficient than traditional association rule mining since previous mining rules are not utilized for discovering new rules while the updated portion is usually small compared to the whole dataset. Consequently the efficiency and the effectiveness are most crucial issue of incremental mining. Algorithms should be such that only updated transactions and previous mined rules to be taken into account for generating new rules [4].

Apriori is a seminal algorithm proposed by R. Agrawal and R.Srikant in 1994 for Apriori is the best-known algorithm to mine association rules. It uses a breadth-first search strategy to counting the support of itemsets and uses a candidate generation function which exploits the downward closure property of support.

Most of the incremental association rule mining algorithms are based on apriori algorithm. However there are many algorithms which are based on FP-growth. But in this paper I have proposed one promising frequent itemset algorithm based on bucket sort approach which is based on apriori algorithm. There are many incremental algorithms which are based on apriori algorithm like FUP, FUP2, UWEP, and NFUP.

### *A.FUP*

The FUP algorithm modified the Apriori mining algorithm and adopted the pruning techniques used in the DHP algorithm. It first calculated large itemsets mainly from newly inserted transactions, and compared them with the previous large itemsets from the original database. According to the comparison results, FUP determined whether re-scanning the original database was needed, thus saving some time in maintaining the association rules. FUP does rescanning of original database [4]. FUP2 is same as FUP but only difference is that FUP works with only incremented database while FUP2 work with incremented as well as decremented database also.

### *B.FUP2*

FUP2 is same as FUP but only difference is that FUP works with only incremented database while FUP2 work with incremented as well as decremented database also[10].

### *C.UWEP*

This significantly reduces the number of candidate itemsets in Δ+. Consequently, these early pruning techniques can enhance the efficiency of FUP-based algorithms. The advantage of algorithm UWEP over other FUP-based algorithms is that it prunes the supersets of an originally frequent itemset in *D* as soon as it becomes infrequent in the updated database rather than waiting until the k-th iteration[10].

### *D. NFUP*

To mine new interesting rules in updated database, NFUP partitions the incremental database logically according to unit time interval (month, quarter or year, for example). 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. The latest information is at the last partition of incremental database[4] Therefore, NFUP scans each partition backward, namely, the last partition is scanned first and the first partition is scanned last [4]. NFUP is suited frequently updated databases.

## IV. PROPOSED METHODOLOGY

In this paper we have proposed new approach of finding frequent itemset and promising frequent itemset based on bucket sort approach without scanning original database. Firstly original database is scanned and make the table of frequent itemset and promising frequent itemset. Promising frequent itemset can be found out using below formula.

$$\min\_\sup_{DB} - \left( \frac{\max supp}{total\ size} \times inc\_size \right) \leq \min\_PL < \min\_\sup_{DB}$$

In this paper apriori algorithm is applied to find frequent and promising frequent itemset. It process in two step prune and join step. In join step frequent itemset and promising frequent itemset of incremented database can be joined together. For a frequent item, its support count must be higher than Then, If any k-item has support count greater than or equal to min_sup(DB∪db), this itemset is moved to a frequent k-item of an updated database. In the other case, if any k-item has support count less than min_sup (DB∪db) but it is greater or equal to min_PL(update) , this k-item is moved to a promise frequent itemset of an upated database. The following algorithms are developed to update frequent and promising frequent k-tems of an updated database.

$$\min\_PL_{DB\cup db} = \min\_\sup_{DB\cup db} - \left( \frac{\max supp}{total\ size} \times inc\_size \right)$$

Here min_sup $_{DB}$ is minimum support threshold maxsupp is the maximum support of one itemset.Total size is total size of original database and inc_size is incremented size of incremented database. Then after as time advances some new data may be arrived so on incremented database we have to calculate frequent itemset and promising frequent itemset. On the Promising frequent itemset list of incremented database we have to apply bucket sort approach. We have to make buckets of new promising list. How many buckets are going to be select for frequent itemset that are based on below formula?

$$\text{Needed bucket} = \frac{\text{Transaction size}}{\text{Bucket Size}}$$

Now how to decide bucket size is based on following formula

$$BU_s = \left( \frac{ST_{DB}}{BU_N} \right) + 1$$

A. *Algorithm*

**ALGORITHM 1: TO FIND FREQUENT AND PROMISING FREQUENT ITEMSET**

**Input:**

**(**1) $L^k$ DB : frequent k-itemset in original database,

(2) $PL^k$ DB: promising frequent k-itemset in original, database

**Output:**

1. $L^k$(DBUdb) : frequent k-itemset in updated database,.
2. $PL^k$ (DBUdb): Promising frequent k-itemset in updated database
3. Temp1 : new estimated frequent k-itemset and new estimated promising frequent k-itemset in updated database

```
1.  k=2
2.  While k <= (length(Lk) +length(P Lk)) < n do
3.  Scan db for ( Lk) , (P Lk) and (items) Є Temp_newCk
4.  X.support(DBUdb) = X.supportDB + X.supportdb
5.  For each X ∈ Lk DB do
6.  If X.support(DBUdb) >= min_sup(DBUdb) Then
7.  Add X to Lk (DBUdb)
8.  Else
9.  If X.support(DBUdb)>= min_PL(DBUdb) Then
10. Add X to P Lk (DBUdb)
11. For each X ∈ P Lk DB do
12. If X.support(DBUdb) >= min_sup(DBUdb) Then
13. Add X to Lk (DBUdb)
14. Add X to temp1
15. Else
16. If X.support(DBUdb) >= min_PL(DBUdb) Then
17. Add X to PL1 (DBUdb)
//apply now bucket sort on promising frequent itemset
  18 n ← length [PL1]
  19 For $i$ = 1 to $n$ do
  20 Insert $PL[i]$ into list $B[nPL[i]]$
  21 For $i$ = 0 to $n$-1 do
  22 Sort list $B$ with Insertion sort
  23 Concatenate the lists $B[0]$, $B[1]$, . . $B[n-1]$ together in order.
```

FIG 1. ALGORITHM OF FINDING FREQUENT AND PROMISING LIST

**ALGORITHM 2: TO FIND BUCKET SIZE**

```
1.  1.I=1;
2.  K=1;
3.  START While(k<= $ST_{DB}$)
4.  J=0;
5.  START While(j<10)
6.  $BU_i = ST_{DB}$ ;
7.  j=j+1;
8.  k=k+1;
9.  END WHILE
10. I=i+1;
11. END WHILE
```

FIG 2. ALGORITHM OF FINDING BUCKET SIZE

## V. EXPERIMENTAL ANALYSIS

To evaluate the performance of promising frequent 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. [1]. The synthetic dataset comprises 1, 00,000 transactions has 10 items on average and the maximal size itemset is 4.

In the experiment we compare the performance of existing promising frequent itemset algorithm and proposed promising frequent itemset algorithm based on bucket sort approach. The results of the experiment are shown in below figure.

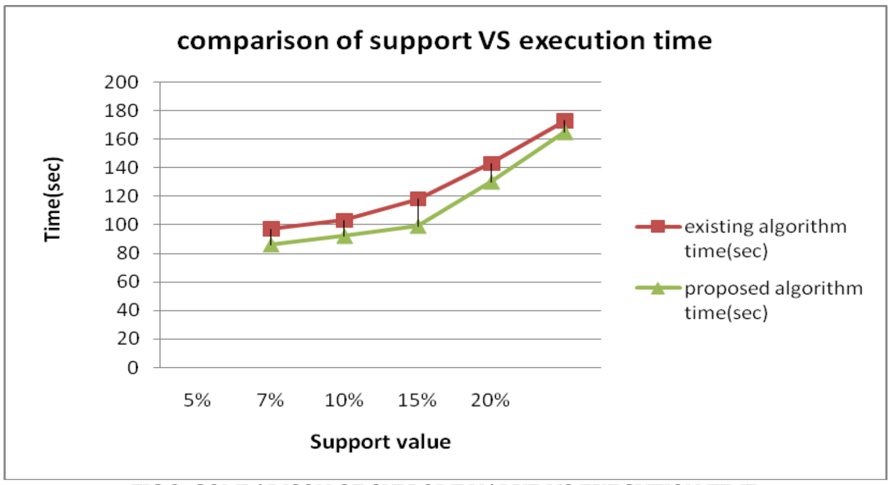

FIG 3. COMPARISON OF SUPPORT VALUE VS EXECUTION TIME

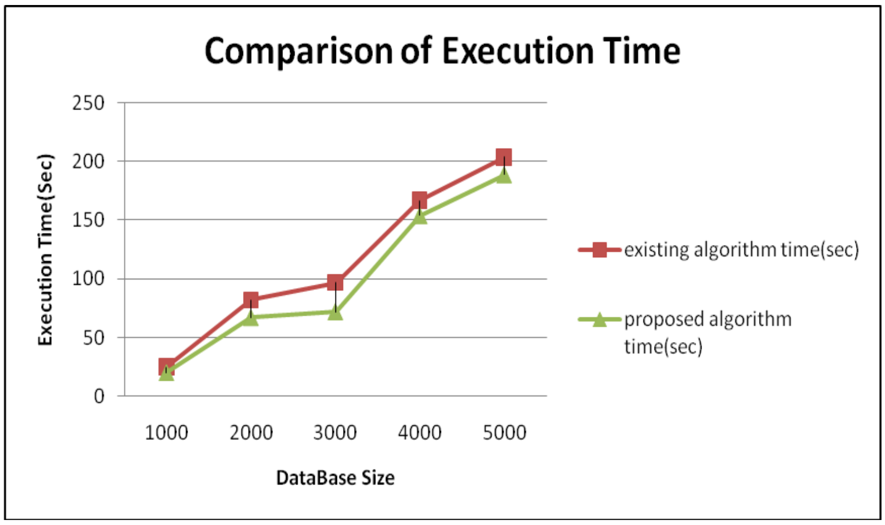

FIG 4. COMPARISON OF EXECUTION TIME OF EXISTING AND PROPOSED ALGORITHM

## VI. CONCLUSIONS

In the real world, databases are continually updated. Therefore, mining must be repeated. Valid patterns and rules must to be efficiently generated. Incremental mining must usually involve the original database scanning and the new added transactions. Scanning the original database is very expensive, so the proposed method avoids rescanning of original database it only scans updated database. Proposed approach for mining, takes less time to execute large database than existing algorithm. Proposed algorithm executes faster than existing algorithm.

There are many scopes to enhance proposed algorithm. One way you can enhance this algorithm is in space complexity. Enhancement in terms of less memory space and get more accurate result. Another is that proposed algorithm is only work with incremented database one can enhance this algorithm that it can work with decremented database also.

## ACKNOWLEDGMENT

We would like to thanks my research guide Mrs. Anita Zala for many discussions and for providing a guidelines throughout my research work.

## REFERENCES

[1] Jiawei Han & Michelline Kamber, Data Mining – Concepts & Techniques, Morgan Kaufmann Publishers (Academic Press), 2001.
[2] Chapter-6: Association Analysis: Basic Concepts and Algorithms - Tan, Steinbach Kumar.
[3] Issues in Data Stream Management [Lukasz Golab and M. Tamer Özsu SIGMOD Record, Vol. 32, No. 2, June 2003]
[4] An Efficient Algorithm for Incremental mining of association Rules [chin-chen chang, yu-chiang Li 15th international workshop IEEE (2005)]
[5] Research Issues in Data Stream Association Rule Mining. [Nan Jiang and Le Gruenwald, SIGMOD 2006]
[6] Incremental Association Rule Mining Using Promising Frequent Itemset Algorithm [Ratchadaporn Amornchewin, Worapoj Kreesuradej IEEE 2007]
[7] Verifying and Mining Frequent Patterns from Large Windows over Data Streams. [Barzan Mozafari, Hetal Thakkar, Carlo Zaniolo, IEEE-2008]
[8] Incremental Maintenance of Association Rules Over Data Streams [Jun Tan, Yingyong Bu and Haiming Zhao, IEEE 2010]
[9] Probability-based Incremental Association Rules Discovery Algorithm with Hashing Technique. *[Ratchadaporn Amornchewin, International Journal of Machine Learning and Computing, Vol.1, No. 1, April 2011]*

[10] Incremental Mining of Association Rules: A Survey [Siddharth Shah ,N. C. Chauhan S. D. Bhanderi (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 3 (3) , 2012]
[11] A Recent Survey on Incremental Temporal Association Rule Mining [Pradnya A. Shirsath, Vijay Kumar Verma International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-3, Issue-1, June 2013]
[12] IBM Quest Synthetic Data Generator: http://sourceforge.net/projects/ibmquestdatagen/