Rapid Update in Frequent Pattern form Large Dynamic Database to Increase Scalability

Abhay Mundra, Poonam Tomar, Deepak Kulhare

Abstract— Association rule mining is a popular data mining technique which gives us valuable relationships among different items in a dataset. In dynamic databases, new transactions are appended as time advances. This may introduce new association rules and some existing association rules would become invalid [1]-[2]. Thus, the maintenance of association rules for dynamic databases is an important problem. Several incremental algorithms, is proposed to deal with this problem. In this paper we proposed algorithm RUPF (Rapid Update in Frequent Pattern). This algorithm reduces a number of times to scan the database (old and new) to generate frequent pattern. As a result, the algorithm has execution time faster than that of previous Algorithms. This paper also conducts experiments to show the performance of the proposed algorithm. The result shows that the proposed algorithm has a good performance.

Index Terms—Association rule, frequent pattern for Dynamic maintenance, incremental algorithms.

I. INTRODUCTION

Database is dynamic when new transactions are inserted and deleted from the database. This may introduce new association rules and some existing association rules would become invalid [1]. As a brute force approach, apriori may be reapplied to mining the whole dynamic database when the database has been changed. However, this approach is very costly even if small amount of new transactions is inserted into a database. Thus, the association rule mining for a dynamic database is an important problem. Several research works have proposed several incremental algorithms to deal with this problem [2]-[3]-[5].

II. EXISTING WORKS

A. FUP [Cheung et al. (1996)]

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. 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 [4]-[6].

B. FUP2 [Cheung et al. (1997)]

This algorithm works on a dynamic dataset where new records may be inserted and some of the existing records may be deleted. It extracts the rules from the final dataset by considering both the deleted parts and the newly added part [3]-[4].

C. Borders [Feldman et al., (1999)]

This algorithm finds the frequent itemsets from the dynamic dataset, using the frequent itemsets already discovered from the old dataset. Here, an infrequent itemset is termed as border set if all the non empty proper subsets of it are frequent. Due to the insertion of new records to the dataset, some of the border sets may become frequent, and is termed as promoted border set. For that- the border sets of the old dataset also have to be maintained along with the frequent sets derived. Based on the promoted border set [7]-[8].

D. Modified borders [Das and Bhattacharyya, (2005)]

This algorithm is a modified version of the borders algorithm that minimizes unnecessary candidate generations. However, this algorithm uses an additional user parameter apart from the parameter support count which is sensitive. With proper tuning of these parameters only- a better performance of the algorithm is possible [9]. When this additional parameter's value is closer to the support count, the algorithm converges to the borders algorithm. Depending on this parameter, the border sets has been divided into four different sets B', B", B"' and B"". The probability of becoming promoted border set is the highest for the elements of B' and lowest for B"". Illustrate through example [10]-[11]

Table I. Transactions database

	TID	Items
D	T1	A B C
	T2	A F
	T3	A B C E
	T4	A B D F
	T5	C F
	T6	A B C
	T7	A B C E
	T8	C D E
	T9	B D E

Generate one item set with support count

Table II. One item set

Item	Support Count
A	6/9
B	6/9
C	6/9
D	3/9

Manuscript received on January, 2013.

Abhay Mundra, CSE Dept M.Tech IV Sem, Central India Institute of Technology, Indore M.P. India.
Poonam Tomar, Asst. Prof. CSE Dept., Central India Institute of Technology, Indore M.P. India.
Deepak Kulhare, Associate Prof. CSE Dept, Central India Institute of Technology, Indore M.P. India.

E	4/9
F	3/9

Generate frequent one item set with minimum support

Table III. Frequent one item set

Item	Minimum support t Count
A	6/9
B	6/9
C	6/9
E	4/9

Now for two items set uses joining in Table III each item is join with every other item. Now calculate support count for each two item set.

Table IV. Two item set

Item set	Support Count
AB	5/9
AC	4/9
AE	2/9
BC	4/9
BE	3/9
CE	3/9

Delete those item which support countless then the given support count

Table V. Frequent two item set

Item set	Minimum Support Count
A,B	5/9
A,C	4/9
B,C	4/9

Now for three item set use joining in Table VI Finally we got three frequent item set with minimum support.

Table VI. Support count

Item set	Support Count
ABC	4/9

Frequent one item set {A}, {B}, {C}

Frequent two item set {A, B}, {A, C}, {B, C}

Frequent three item set {ABC}

Based on our survey and experimental analysis, it has been observed that:

(1) The algorithms work on market basket encoded data, which causes information loss.

(2) They work in two phases, which causes them computationally expensive.

(3) Some of the algorithms work for incremented dataset only; they cannot handle the updated Dataset due to deletion;

(4) May need multiple scanning of the whole dataset.

(5) Huge number of rules may be generated, based on the user parameters [7]-[8]-[12]-[13]-[14].

III. PROPOSED METHOD

Add sum records and also delete some record in original database

Table VII. Transaction are added and deleted from DB

	TID	Item set
D-	T1	A B C
	T2	A F
	T3	A B C E
D	T4	A B D F
	T5	C F
	T6	A B C
	T7	A B C E
	T8	C D E
	T9	A, B D E
D+	T10	A,B,D
	T11	D,F
	T12	A,B,C,D

So after adding some new transaction the updated data base D' is now for frequent pattern support count 40%

Table VIII. Updated database

TID	Item set
T4	A B D F
T5	C F
T6	A B C
T7	A B C E
T8	C D E
T9	A, B D E
T10	A,B,D
T11	D,F
T12	A,B,C,D

Now we convert the Table VIII into vertical format resulted Table.

Table IX. Vertical data format

Item	TID
A	T4, T6, T7, T9, T10, T12
B	T4, T6, T7, T9, T10, T12
C	T5, T6, T7, T8, T12
D	T4, T8, T9, T10, T11, T12
E	T7, T8, T9
F	T4, T5, T11

Now assign a code to each row now use Transaction Id Intersection over vertical transaction.

Table X. Item with support count

Item	TID	RID	Support Count	S/D
A	T4, T6, T7, T9, T10, T12	R1	6	S
B	T4, T6, T7, T9, T10, T12	R2	6	S

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

C	T5, T6, T7, T8, T12	R3	5	S
D	T4, T8, T9, T10, T11, 12	R4	6	S
E	T7, T8, T9	R5	3	D
F	T4, T5, T11	R6	3	D

Delete those row which has support count less than the given support count

Table XI. Two item set

Item
AB
AC
AE
BC
BE
CE

Table XII. One item minimum support count

Item	TID	Row ID	Support Count
A	T4, T6, T7, T9, T10, T12	R1	6
B	T4, T6, T7, T9, T10, T12	R2	6
C	T5, T6, T7, T8, T12	R3	5
D	T4, T8, T9, T10, T11, T12	R4	6

Now perform intersection between Row ID

Table XIII. Two item support count

Row ID Intersection	Transaction sets	Support count	S/D
R1 ∩ R2 (A,B)	{T4, T6, T7, T9,T10, T12}	5	S
R1 ∩ R3 (A,C)	{T6, T7, T12}	3	D
R1 ∩ R4 (A,D)	{T4, T9, T10, T12}	4	S
R2 ∩ R3 (B,C)	{T6, T7, T12}	3	D
R2 ∩ R4 (B,D)	{T4, T9, T12}	3	D
R3 ∩ R4 (C,D)	{T8, T12}	2	D

Delete those row which support count less then the given support count so resulted table

Table XIV. Two item set with minimum support count

Row ID Intersection	Transaction sets	Support count
R1 ∩ R2 (A,B)	{T4, T6, T7, T9,,T10 T12}	5
R1∩ R4 (A,D)	{T4, T9, T10, T12}	4

Now finally take intersection between R1, R2 and R3

Table XV. Three item set with minimum support count

Row ID Intersection	Transaction sets	Support count
R1 ∩ R2 ∩ R4 {A,B,D}	{T4,T9,T10,T12}	4

Frequent one item set {A}, {B}, {D}
Frequent two item set {A, B}, {A, D}, {B, D}
Frequent three item set {ABD}

IV. PROPOSEDALGORITHM (RUPF)

Initialize: K: = 1, C_1 = all the 1- item sets;
Read the database to count the support of C1 to
Determine L1.
L_1:= {frequent 1- item sets};
K: =2; //k represents the pass number//
While ($L_{k-1} \neq \Phi$) ***do***
Begin
C_k: = gen_candidate_itemsets with the given L_{k-1}
Prune (C_k)
For all candidates in C_k do
Count the number of transactions that are common in each item Є C_k
L_k: = All candidates in C_k with minimum Support;
K: = K + 1;
End

V. EXPERIMENT

To evaluate the performance of promising frequent Algorithm, 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 real life dataset. The technique for generating the dataset is proposed by Agrawal and etc. The dataset comprises over 10,000 transactions Firstly, the proposed algorithm is used to find frequent item set from an original database. Then, several sizes of incremental databases, i.e. 10%, 20%, and 30% of the original database, are added to the original database. For comparison purpose, FUP algorithm is also used to find association rules from the same original database and the same incremental databases. The experimental results with various minimum support thresholds are shown in Table and Figure. From the results, the proposed algorithm has better running time than that of FUP algorithm, for Scalability When support count is 50%

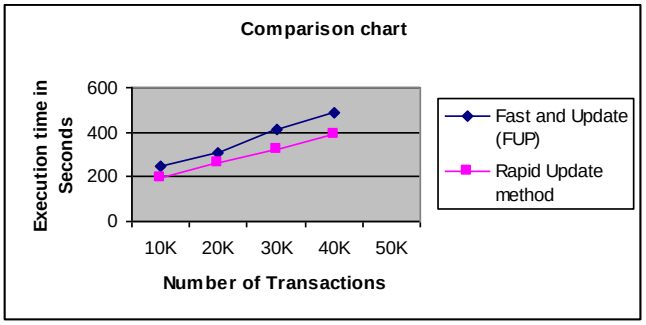

Figure 1. Comparison FUP & RUPF using number of record

For Efficiency When number of records is 30k

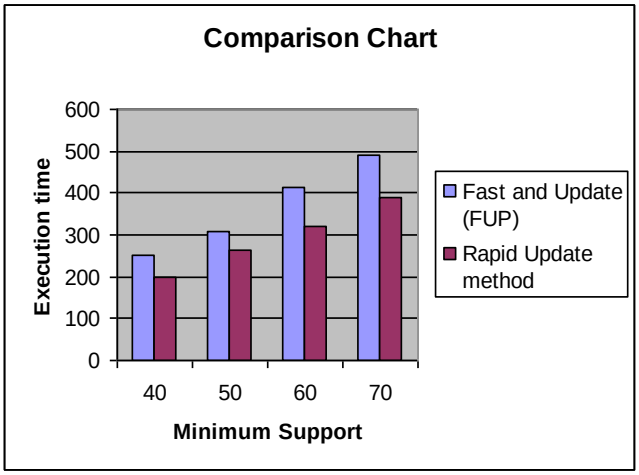

Figure 2. Comparison FUP and RUPF support count number of record

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

VI. CONCLUSION

From the above experiment it is clear that PUPF takes less execution time as compared to the fast and Update algorithm We compare RUP and FUP by using different number of record and by different number of minimum support value .So RUPF is more efficient as compared to fast and update algorithm

REFERENCES

1. Rahman, Mohammad.M AL-Widyan "Reduce Scanning Time Incremental Algorithm (RSTIA) of Association rules" Academic Research International2, September Volume 1, Issue 2, September 2011 University, Amman, JORDAN
2. N.L. Sarda N. V. Srinivas "an Adaptive Algorithm for Incremental Mining of Association Rules "Computer Science and Engineering Indian Institute of Technology Bombay Mumbai.
3. Siddharth Shah, N. C. Chauhan, S. D. Bhander "Incremental Mining of Association Rules: A Survey "International Journal of Computer Science and Information Technologies, Vol. 3 , 2012,
4. Wei-Guang Teng and Ming-Syan Chen "Incremental Mining on Association Rules" Department of Electrical Engineering National Taiwan University Taipei, Taiwan.
5. Animesh Tripathy, Subhalaxmi Das "An Association Rule Based Algorithmic Approach to Mine Frequent Pattern in Spatial Database System" International Journal of Computer Science & Communication Vol. 1, No. 2, July-December 2010,
6. Sandhya Rani Jetti, Sujatha D "Mining Frequent Item Sets from incremental database: A single pass approach "International Journal of Scientific & Engineering Research, Volume 2, Issue 12, December-2011,
7. Chelliah Balasubramanian*, Karuppaswamy "A mining method for tracking changes in temporal association rules from an encoded database" International Journal on Computer Science and Engineering Vol.1(1), 2009,
8. Nibedita Panigrahi "An Efficient Algorithm for Mining Of frequent items using incremental model" Konark Institute of Science and Technology International Journal of Computer Science & Informatics, Volume-1,
9. Wuzhou Dong, Juan Yi, Haitao He, and Jiadong Ren "An incremental algorithm for frequent pattern mining based on bit-sequence (IJACT)" Volume3, Number9, October 2011
10. Ahmed Taha1, Mohamed Taha1, Hamed Nassar2, and Tarek F. Gharib3 "DARM: Decrement Association Rules Mining "Journal of Intelligent Learning Systems and Applications,
11. Romans Tumasonis, Gintautas Dzemyda "A probabilistic algorithm for mining frequent sequences "
12. Jia-Dong Ren and Xiao-Lei Zhou Yanshan "A New Incremental Updating Algorithm for Mining Sequential Patterns "University, , China Journal of Computer Science 2
13. F.A. Dafa-Alla, Ho Sun Shon and Khalid E.K. Saeed "IMTAR: Incremental Mining of General Temporal Association Rules " Journal of Information Processing Systems, Vol.6, No.2, June 2010
14. Vasile Parvan 2, Timisoara, Romania "A Comparative Study of Association Rules Mining Algorithms"Computer & Software Engineering Department, Politehnica University of Timisoara, Bd.