Efficient calendar based temporal association rule

Keshri Verma , O. P. Vyas
School of Studies in Computer Science
Pt. Ravishankar Shukla University Raipur Chhattisgarh
{keshriverma,opvyas}@rediffmail.com

Abstract

Associationship is an important component of data mining. In real world data the knowledge used for mining rule is almost time varying. The item have the dynamic characteristic in terms of transaction , which have seasonal selling rate and it hold time-based associationship with another item. It is also important that in database, some items which are infrequent in whole dataset but those may be frequent in a particular time period. If these items are ignored then associationship WVW200R3100221-398 result will no longer be accurate. To restrict the time based associationship calendar based pattern can be used [YPXS03]. A calendar unit such as months and days, clock units, such as hours and seconds & specialized units , such as business days and academic years, play a major role in a wide range of information system applications[BX00].

Most of the popular associationship rule mining methods are having performance bottleneck for database with different characteristics. Some of the methods are efficient for sparse dataset where as some are good for a dense dataset. Our focus is to find effective time sensitive algorithm using H-struct called temporal H-mine, which takes the advantage of this data structure and dynamically adjusts links in the mining process [PHNTY01]. It is faster in traversing & advantage of precisely predictable spaces overhead. It can be scaled up to large database by database partitioning, end when dataset becomes dense, conditionally temporal FP-tree. can be constructed dynamically as part of mining.

1. Introduction –

The Associationship is an important component of data mining. It indicates the co-relationship of one item with another. For example Egg ==> coffee (support 3%, confidence 80%) means that 3% of all transaction contain both egg & coffee, and 80% of transaction that have egg also have coffee in them. In real dataset time is one of the important factors. For example egg and coffee may be ordered together primarily between 7 to 11 AM in this interval the support & confidence is 40% but at another interval, support is as low .005% in other transaction [YPXS03]. This discussion suggests that different association rules may be discovered while considering different time intervals associated to it. Many items are introduced or removed form the database, that is items lifespan[ZMTW02] which means that item is valid on specific time interval. To discover such temporal intervals (with calendar information) together with the association rules that hold during the time interval may lead to useful knowledge. If calendar schema is applied in association it is called calendar based temporal association rule.

A hierarchy of calendar concepts determines a calendar schema. A calendar unit such as months and days, clock units, such as hours and seconds & specialized units, such as business days and academic years, play a major role in a wide range of information system applications [BX00]. A calendar schema defines a set of simple calendar – based patterns. Each calendar pattern defines a set of time intervals.

Recent researches in the field of temporal association rule mining are using Apriori based approach. Nevertheless, these proposed approaches may still encounters some difficulties for different datasets such as Sparse or dense dataset. The limitations in these approaches are two fold.

First, huge space is required to perform the mining in Apriori based temporal association rule [JG00]. It generates a huge number of candidates in case of a dataset, which is large and/or sparse. Our first part of the algorithm which if the database is huge and sparse, the temporal approach of FP –tree outperform over Apriori and the space requirement for recursion is a challenge. Thus necessitates improvement in the existing approach. Second part of the algorithm generates Fp-tree when the data have the dense characteristic, no algorithm can bits the performance of FP-tree.

Second, the mining approach should ideally have more scalability. Many existing methods are effective when the dataset are not large. The existing Apriori based temporal

association rule , may easily cause thrashing when dataset become large and sparse.

The approach of frequent pattern mining is to find the complete set of frequent patterns in a given transaction database with respect to a given support threshold. Our data-mining problem is to discover all temporal association rules w.r.t. Calendar schema from a set of time stamped transactions. This paper ,improves an existing frequent pattern tree approach to discover temporal association rule to increase the memory performance over existing one[PHNTY01] , it uses the data structure H-struct, and incorporating temporal aspects with the following progress:

First, a memory based efficient pattern growth algorithm, Temporal H-mine is proposed for mining time based frequent patterns for dataset that can fit in memory. H-struct is used for fast mining on time-based dataset. It has polynomial space complexity and is thus more space efficient than pattern growth method like FP-growth and Tree-Projection when mining sparse dataset, and more efficient than Apriori based frequent pattern mining[JG00].

Second based on H-mine data structure we propose Temporal based H-mine association rule mining algorithm.

Third for dense datasets. H-mine is integrated with Temporal FP-growth dynamically by detecting the swapping condition and constructing FP-tree for effective mining.

The Temporal base H-mine algorithm is scalable in both large and medium size dataset and in both the cases dense and sparse dataset.

The rest of the paper is organized in five section In Section 2, we discuss some related works. In section 3 we define temporal association rule in term of calendar schema. In Section 4 elaborate the extended algorithm of frequent pattern approach ,section 5 shows conclusion & future works and section 6 provides application of above investigation.

2. Related work –

The concept of association rule was introduced as Apriori algorithm [AS94]. Its performance was improved by deploying frequent-pattern growth approach [PH02]. In paper [ORS98] the omission of the time dimension in association rule was very clearly mentioned. Fp-growth algorithm is best if data is dense, & Apriori algorithm performs better if data is sparse, H-mine [PHNTY01] algorithm which is used to mine the dataset from both the cases sparse & dense. A temporal aspect of association rule was given by Juan [JG00]. According to this transaction in the database are time stamped and time interval is specified by the user to divide the data into disjoint segments, like month , days & years. Further The cyclic association rule was introduced by Ozden [ORS98] with minimum support & high confidence. Using the definition of cyclic association rule, It may not have high support & confidence for the entire transactional database. A nice bibliography of temporal data mining can be found in the Roddick literature[RHS00].Rainsford & Roddick presented extension to association rules to accommodate temporal semantics. According to [RR99] logic the technique first search the associationship than it is used to incorporate temporal semantics. It can be used in point based & interval based model of time simultaneously[RR99]. A Frequent pattern approach for mining the time sensitive data was introduced in[CJJX03] Here the pattern frequency history under a tilted-time window framework in order to answer time-sensitive queries. A collection of item patterns along with their frequency histories are compressed and stored using a tree structure similar to FP-tree and updated incrementally with incoming transactions [CJJX03].

3. Problem definition :

3.1 Association Rule:

The concept of association rule, which was motivated by market basket analysis and originally presented by Agrawal. [AS94]. Given a set of T of transaction, an association rule of the form X==> Y is a relationship between the two disjoint itemsets X & Y. An association rule satisfies some user-given requirements. The support of an itemset by the set of transaction is the fraction of transaction that contain the itemset. An itemset is said to be large if its support exceeds a user-given threshold minimum support. The confidence X ==> Y over T is a transaction containing X and also containing Y. Due to complex candidate generation in the data set Jiewai Han invented a new technique of FP-growth method for mining frequent pattern without candidate generation [PH02]. Efficiency of this mining technique is better than all most all algorithm like Apriori, AprioriTid, Apriori Hybrid when data is dense because (1). a large dataset is compressed into a condensed smaller data structure which avoids costly & repeated data scan (2). FP-tree-based mining adopts a pattern-fragment growth method too avoid the costly generation of a large number of candidate generation sets and,(3). A partitioning-based divide- and-conquer method is used to decompose the mining task into a set of similar tasks for

conditional database which dramatically reduce the search space.

In our opinion this mining associationship will be become more useful if we include the time factor in to it.

3.2 Temporal association rule

Definition 1 : The frequency of and itemset over a time period T is the number of transactions in which it occurs divided by total number, of transaction over a time period. In the same way , confidence of a item with another item is the transaction of both items over the period divided by first item of that period.

Support(A) = Frequency of occurrences of A in specified time interval / Total no of Tuples in specified time interval

Confidence(A => B[Ts,Te]) = Support_count(A U B) over Interval / occurrence of A in interval

T_s indicates the valid start time & T_e indicate valid time according to temporal data.

3.3 Simple calendar based Pattern :

When temporal information is applied in terms of date, month , year & week form the term calendar schema. It is introduced in temporal data mining. A calendar schema is a relational schema (in the sense of relational databases) R = (f_n : D_n, F_{n-1} : D_{n-1},……F_1 :d_1) together with a valid constraint. A calendar schema (year : {1995,1996,1997…..} , month : {1,2,3,4,……12}, day : {1,2,3…..31} with the constraint is valid if that evaluates (yy, mm, dd) to True only if the combination gives a valid date. For example <1955,1,3> is a valid date while ,<1996,2,31> is not.

In calendar pattern , the branch e cover e' in the same calendar schema if the time interval e' is the subset of e and they all follow the same pattern. If a calendar pattern <d_n, d_{n-1}, d_{n-2}……..d_1> covers another pattern <d'$_n$, d'$_{n-1;}$, d'$_{n-2}$ ……..d_1> if and only if for each I, 1<=i<=n or d_i = d'$_i$.

Now Our task is to mine frequent pattern over arbitrary time interval in terms of calendar pattern schema.

4 Proposed work-

4.1 Temporal H-Mine (Mem) : Memory -Based Hyper Structure Mining using time dimension.

The problem of Temporal Frequent mining is to find the complete set of item which frequently occurred in valid time interval, for a given support threshold.

This section elaborates how Temporal H-mine process is applied for temporal association rule.

Example 1 Let the first three column of the table be our running transaction id, transaction item and date in which the transaction happened.

Trans Id	Items	Date	Frequent items Projection
100	c,d,e,f,g,i	<*,01,04>	{c,d,e}
200	a,c,d,e,m,b	<*,01,04>	{a,c,d,e,b}
500	a,c,d,e,b	<*,01,04>	{a,c,d,e,b}
400	a,c,d,h	<*,06,04>	{a,d,h}
600	a,b,d,h,i	<*,06,04>	{a,b,d,h}
300	a,b,d,e,g,k	<*,06,04>	{a,b,d,h}

Table 1: Transaction Database

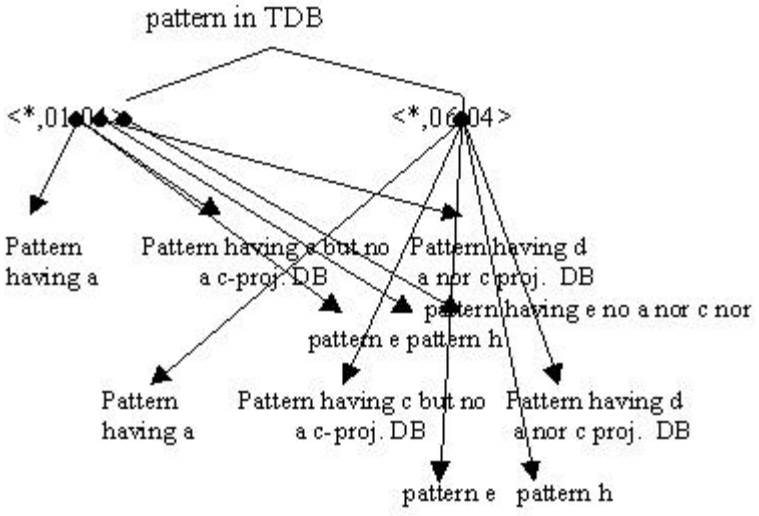

Figure 1: Divide and Conquer method on the basis of time Database

Only the frequent items play roles in frequent pattern mining. A item with lifespan is frequent for some period of time may be infrequent in whole dataset. Due to the pattern change the probability of finding relationship will also be vary in real dataset. By scanning TDB once, the complete set of frequent items on <*,01,04> are {a:2,c:3,d:2,e:3,f:1,g:1,b:2,i:1} and <*,06,01> are{ a:3,b:2,d:3,h:2,e:1,g:1,e:1,k:1 }

A header table H is created separately for each interval, where each frequent item entry has three field : an item-id, a support count and a hyper-link. When the first item projections are loaded into memory, those with the same first item (from DB list) hyper-links as a queue, and entries in header table H act as the head of the

queues. To mine a-projected database ,an a-header table Ha is created. In Ha , every frequent item except for a itself has entry with three field item id, support count and hyper-links & time also [PHNTY01].

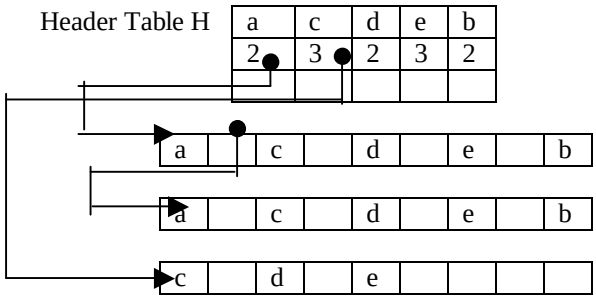

Figure 2 (a). H-struct , the hyper structure for storing frequent -item projection in specific first interval <*,01,04>

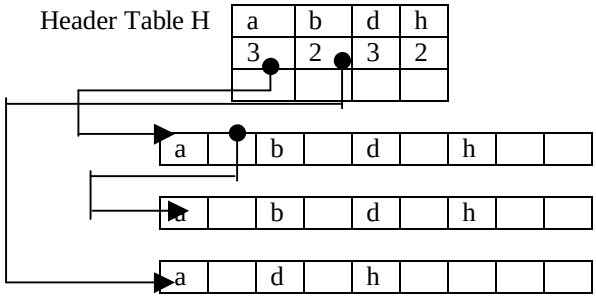

Figure 2 (b): H-struct , the hyper structure for storing frequent -item projection in specific second interval <*,06,04>

Algorithm
Temporal H-mine : The algorithm for memory-based hyper-structure mining.

Input : transaction database TDB with time interval , Support threshold min_sup.

Output : Set of frequent itemset on different time interval.

Method :

1. Scan transaction database TDB once to find L, the complete set of frequent items.
2. Partition TDB into k parts , TDB_1, TDB_2..... TDB_k such that, each TDB_i (i <i <= k) the frequent item projection held in main memory..
3. Check the itemset valid time interval e cover e_0
4. for i = 1 to k, use H-mine (mem) to mine frequent patterns in TDBi with respect to min_sup .
5. Let $F = U^{k}_{i=1} F_i$. Scan TDB one more time, Collect support for frequents in F. Output those patterns which pass the min. supp. on specific time interval.

Although H-mine perform better than Fp-tree in specific cases but Fp-growth method have several advantages over Mining on H-struct since FP-tree shares common prefix path among different transaction., which leads to saving the spaces & time as well. Temporal aspect of data is more important because if a part of data is dense on specific interval may be sparse if we consider the whole dataset. H-mine algorithm is extended when data is dense it call Fp-tree approach for frequent pattern growth method.

4.2 **Handling Dense data sets : Dynamic integration of H-struct and Fp-tree-based mining**

Finding time based frequent pattern in dense database is a challenging task. The FP-growth [PH02] works well known when data is dense. In comparison with FP-growth, Temporal H-mine does not generate physical projected database and conditional Fp-trees and thus save space as well as time in may cases. Temporal Fp-tree based mining has its advantages over mining on H-struct since Temporal Fp-tree shares common prefix paths among different transaction., which may lead to space and time.

The support of dataset in the data warehouse can be maintained by dividing it in different intervals. The support of a item in interval t1 can not be the same in interval t2. A infrequent or less support item in interval t1 can be frequent item in interval t2.

The calendar schema is implemented by applying Apriori algorithm [YPXS03]. It follows the candidate generation approach in order to mine the frequent item. We assist here that instead of candidate generation H-mine and divide & conquer approach is more efficient than Apriori approach. It construct queues for maintains the list of items a tree & each branch indicate the association ship of item. It reduces the size of dataset and increases the performance & efficiency of algorithm. It can solve following queries (1) What are the frequent set over the interval t_1 and t_2 ? (2) what are the period when (a,b) item are frequent ?

(3) Item which are dramatically change from t4 to t1.

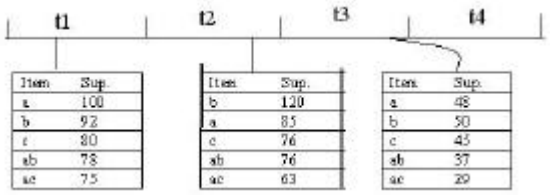

Figure 3 : Frequent pattern in different interval

Lemma 1 :- During transaction in database the association of a item over the support ? can be obtained by projection of the branch of FP-tree.

Rationale : Based on the TFP-tree construction process its frequent item can be projected into a single branch.

For a path a_1, a_2 a_3.....a_k from the root to a node in a FP-tree. Suppose a_{ak} be the count at the node labeled $a_{k\ and}$ c'_{ak} be the sum of the count of the branch of the node.

Tid	Item bought	Date	Calendar pattern	Item in Descending order of Frequency
100	f,a,c,d,g,i,m,p	01/01/2004	<*,01,04>	f,c,a,m
200	a,b,c,f,l,m,o	01/01/2004	<*,01,04>	f,c,a,b,m
300	b,f,h,j,o	02/01/2004	<*,01,04>	f,c,b,o
500	a,f,c,e,l,p,m,n	03/06/2004	<*,06,04>	f,c,a,e,m,l
600	f,a,c,d,e	04/06/2004	<*,06,04>	f,c,a,e
700	b,f,h,j,o,l,o	06/06/2004	<*,06,04>	f,o,l

Table 2. Transaction database in running example

Definition (Temporal FP-Tree) – A Temporal frequent pattern (FP) is tree structure defined below.

It consists of root labeled as "null".

It consist of a set of item-prefix sub trees as the children of the root, and a frequent item header table.

Each node in the item prefix sub tree consists of four fields :

(a) Item name - Item name represents the name of item which is registers on that node (b) count - count registers the number of transactions represented by the portion of the path reaching this node (c) node link - node-link links to the next node of temporal FP-tree (d) calendar pattern time calendar pattern represent the time in which the item transaction appeared.

Each entry in the Frequent –item header table consists of two fields (1) Item name (2) Head of node link

Algorithm : (FP- Tree construction)

Input : A transaction database DB and a minimum support threshold

Output : FP Tree, Fp Tree, Frequent item

Method : The FP tree is constructed ad follows :

1. Scan the database DB once. Collect f, the set of frequent item and support of each item. Sort F from support in descending order as Flist, the list of frequent items
2. Create the root of Temporal FP tree and label it as "Null". For each transaction in DB and do the following

Select the frequent items in Trans and sort them in descending order of Flist. Let the sorted frequent –item list in the Trans be p[P] where p is first element and P is the remaining list. Cal insert_tree(p[P],T).

```
Procedure insert_tree(p[P],T).
{
Step(1) If T has a child N such that
Step(2) if (N.time = P.time) then
// For checking interval phase I
Step(3) if (N.itemname = P. itemname) then
Step(a) N.count = N.count +1
//  Increment the count by 1
Step(4) else create a new node // Node created
//  on the same branch
Step(5) Link to its parent P.count = 1  // Initialize
the counter by 1.
Step(6) else create a new Branch link from the
root.
        Call insert_tree(P,N) recursively

} // End of Function
```

Temporal Frequent pattern Tree : Design & Construction

Let I = {a_1, a_2, a_3.. a_m} be a set of items , and a transaction database DB {T_1, T_2, T_3.....T_n } where T_i {i ? [1..n] } is a transaction which contains a set of items in I.

4.1 TEMPORAL FREQUENT- PATTERN TREE

To design the Temporal FP-tree for frequent pattern mining , let's first examine example from table1.

1. Since time is the most important feature of real world data set, so arrange the item in according to time and define the calendar pattern or interval in calendar unit form.

2. In calendar pattern <*> is used to define that any day or month for example if it used <dd,mm,yy> calendar pattern <*,01,04>

represents any day of Month January & year 2004.

2. Since only the frequent item will play a role in the frequent pattern mining. So first scan is used to identify the set of frequent items
3. If the set of frequent items of each transaction can be stored in some compact data structure , it may be possible to avoid repeatedly scanning the original transaction database.
4. If multiple transaction share a set of frequent items, it may be possible to merge the shared sets with the number of occurrences registered as count. It is easy to check whether two sets are identical if the frequent items in all of the transaction are listed according to a fixed order.
5. If two transaction share a common prefix , according to some sorted order of frequent items, the shared part can be merged into one prefix structure as long as the count is registered properly.

With the above observation , a Temporal frequent pattern tree can be constructed as follows :

First , a scan of DB drives a frequent list of items in schema <*,01,04> are {(f:3),(C:2),(b:2), (a:2),(m:2} same for calendar pattern <*,06,04> frequent items are {(f:3),(c:2),(a:2),(e:2),(m:2)(l:2)} and remaining items are infrequent so skip those item .

Second, the root of the tree is created and labeled with "null". The FP- tree is constructed as follows by scanning the transaction database DB in second time.

1. The scan of the first transaction leads to construction of the first branch of the tree {(f:1),(c:1),(a:1),(m:1),(p:1)}& it follow the calendar pattern <*,01,04>.
2. For the second transaction its temporal period is same so it follow the same branch & the frequent item list {f,c,a,b,m,o} shares a common prefix <f,c,a> the count of each node along the prefix is incremented by 1. and a new node (b:1) is created and linked to child of (a:2), another new one (m:1) is created and linked to as the child of (b:1) and another new one (o:1) is created and linked to as the child of (m:1),
3. For the third transaction its time period is same as previous the transactio is <f,c,b,o> shares common prefix <f,c> so f & c's count is incremented by 1, and a new node b is created although b is already existing but it not go that branch it is not common prefix of the node. Node b is linked as a child of (c:2) and a new node o is created with intialize the count because o is first time introduce on Temporal FP-tree and linked as a child of node <b:1>
4. The scan of forth transaction leader to construct another branch because its time period <*,06,04>does not match with existing branch's node time period. New nodes are created with <(f:1),(c:1),(a:1),(e:1),(m:1),(l:1)>
5. The scan of fifth transaction which follow the time interval of forth transaction so if follow the same branch if the item prefix match, It can share the common prefix <f,c,a,e> with existing path <f,c,a,e,m,l> the count of each node incremented by 1.
6. For the last transaction ,<f,m,l> its time time interval match with second branch so it follow the second branch of FP-tree here it share common prefix f , its count is incremented by 1 , a new node is created for item m , & it is linked to node f by initializing the counter value to1. for next item l again a new node will be created by initializing its counter value

4.3 Mining the frequent item from FP-tree –

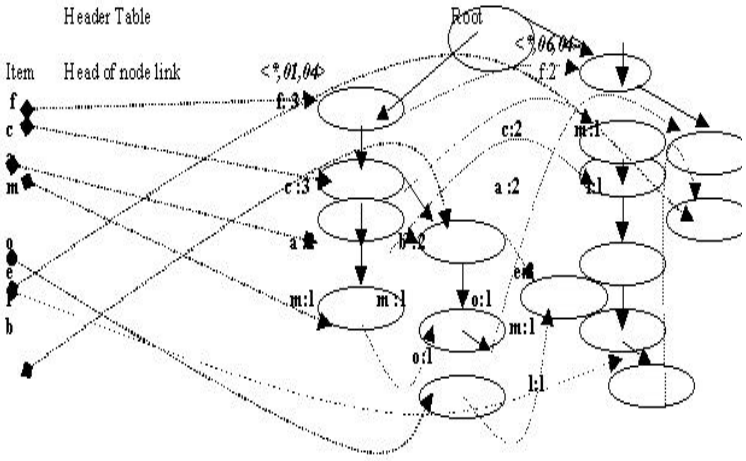

Figure 4 FP tree with different time interval

& it is linked as a child of node m.

In Step(2) if (N.time = P.time) in phase I , its meaning that when a new node p appears in the FP-tree we check the time of transaction , is inside the time of transaction of N item, defined as below table, then it follows the same branch

otherwise a new branch will be created in FP-tree.

Item name	N.Time	P. Time	Branch
F	<01,01,04>		First
C	<01,01,04>	<01,01,04>	First
A	<01,01,04>	<01,01,04>	First
B	<01,06,04>		Second
M	<01,06,04>	<01,06,04>	Second

Table 3. Shows the path depend on the time of transaction in itemset.

Property [PH02] Node Link property . For any frequent item a_i , all the possible patterns contining only frequent items and a_i can be obtained by following a_i, s node link's , starting from a_i 's head in the FP-tree header.

Mining process from the constructed Temporal FP-tree shown in figure1. We examine the mining process by starting from the bottom of node-link header table.
For calendar pattern <*,01,04> for node m ,its intermediate frequent pattern is (m:3), and its path are <f:2,c:2,a:1,m:1,p:1> , <f:1,c:1,b:1,m:1> and <f:1, m:1>. Thus to study which appears together with m at time period <*,01,04> only m prefix {(fca:1) ,(fcb:1),(f:1) }, m's sub -pattern base , which is called m's pattern conditional pattern base.(which is called m's conditional Fp tree) leads to two different branch (fc:3) & (f:2).
For node a , its immediate frequent pattern is (a:4)and it is in two different path one for<*,01,04> & second for <*,06,04>. Calendar pattern <*,01,04> consist of <f:3,c:3 a:2> and <*,06,04> consists of <f:2,c:2>

Item	Time Interval	Conditional Pattern base	Conditional FP-tree
m	<*,01,04>	{(fca:1) ,(fcb:1),(f:1) }	{f:2 c:2|m)
	<*,06,04>	{(fcae:1),(f:1)}	(f:2|m)
a	<*,01,04>	{(fc:2)}	(f:2|a)
	<*,06,04>	{(fc:2)}	(f:2|a)
0	<*,01,04>	{(fcabmo:1),(fcbo:1)}	(f:2,c:2,b:2|o)
	<*,06,04>	ϕ	ϕ
1	<01,*,04>	ϕ	ϕ
	<*,06,04>	{(fcaeml:1),(fml:1)}	(f:2|l)
e	<01,*,04>	ϕ	ϕ
	<*,06,04>	{(fca:2)}	(f:2,c:2 a:2|e)
c	<01,*,04>	{(fc:3)}	(f:2|c)
	<*,06,04>	{(fc:2)}	(f:2|c)
b	<*,01,04>	{(fcb:2)	(f:2,c:2|b)
	<*,06,04>	ϕ	ϕ
f	<01,*,04>	ϕ	ϕ
	<*,06,04>	ϕ	ϕ

Table 3 Mining Temporal frequent patterns by creating conditional (sub) pattern base

From the Temporal FP-tree the conditional frequent pattern tree can be generated by calling the section 3.3 procedure of Frequent pattern–growth method conditionally for every valid interval[PH02].

5. Conclusion & Future work-

In this paper, we have proposed algorithm gives an efficient time sensitive approach for mining frequent item in the dataset. Discovered rule is easier to understand. Temporal H-mine , which takes advantage of H-struct data structure and dynamically adjust link in the mining process.

Temporal Fp-tree, uses divide & conquer technique for construction & traversing of tree which is used to decompose the mining task into a set of smaller task for mining confined pattern in conditional database which dramatically reduce the search space on specific time interval when the data is sparse. H-mine algorithm not need to physically construct memory structures of projected database. In fact Data mining concepts are applied where there are huge set of data available in data warehouse. It requires more scanning & processing time. Hence after applying our logic of the scanning this valid time & processing time can be decreases for mining the frequent set of items. It is very useful for retailer to create its own market strategy as per the requirement of time.

The work can be further extended for designing good classifier and performance can be increases.

Applications:

?? ***Business Application :*** This technique is most useful in Business mining. Most of the real world data have the time varying features. The retailer can change their business policy with time to time for maximize the output Example some model of vehicle are not available from 1980s , suppose is currently appears in the market. Its history indicate no associationship but the fact is that product is not available on that period , so its associationship is started from the interval where it was valid.

?? ***Web Mining :*** The concept can be applicable in web mining , In WWW the site which is no longer available so its associationship also be no longer.

?? ***Clustering Problem :*** This approach can be useful to solve the clustering problem, the cluster can be designed on the basis of period of data., that will reduce the size of data & processing time also.

References –

[AS94] R. Agrawal & R. Srikant , R :" Fast algorithm for mining association rule.In VLDB'94 Chile , Sept 1994,pp –487-499.

[BX00] Claudio Bettini, X. Sean Wang R: " Time Granularies in databases , Data Mining , and Temporal reasoning 2000. pp 230, ISBN 3-540-66997-3, Springer-Verlag, July 2000. 230 pages. Monograph.

[CJJX03] Chris Giannella_, Jiawei Hany, Jian Peiz, Xifeng Yany, Philip S. Yu R: Mining Frequent Patterns in Data Streams at Multiple TimeGranularities, pg 191 – 210, H. Kargupta, A. Joshi, K. Sivakumar, and Y. Yesha (eds.), Next Generation Data Mining, 2003.

[JG00] Juan M .Ale , Gustavo H. Rossi R: " An approach to discovering temporal association rules", ACM SIGDD March 1..21,2002

[JM01] Jiawei Han, Micheline Kamber , Book : " Data Mining Concept & Technique",2001

[ORS98] Banu Ozden , Sridhar Ramaswamy , Avi Silberschatz R : "Cyclic Association Rule" ,In Proc. Of forteenth International conference on Data Engineering 1998, pp 412-425

[PH02] Jian Pei, Jiawei Han, Yiwen Yin and Running Mao R : Mining Frequent Pattern without Candidate Generation", Kluwer online Academy 2004.

[RHS00] John F. Roddick, Kathleen Hornsby, Myra Spiliopoulou: An Updated Bibliography of Temporal, Spatial, and Spatio-temporal Data Mining Research. TSDM 2000: pp147-164.

[RMS98] S. Ramaswamy, S. Mahajan, and A. Silberschatz. On the discovery of interesting patterns in association rules. In Proc. of the 1998 Int'l Conf. on Very Large Data Bases, pp 368–379, 1998.

[RR99] Chris P. Rainsford, John F. Roddick R: " Adding Temporal semantics to association rule", 3rd International conference KSS Springer 1999, pp 504-509

[YPXS03]Yingjiu Li, Peng Ning, X. Sean Wang , Sushil Jajodia R :"Discovering calendar- based temporal association rules", Data & Knowledge Engineering volume 4,Elesvier publisher, Volume 44 pp– 193-214 ,2003

[PHNTY01] ***J. Pei**, J. Han, H. Lu, S. Nishio, S. Tang, and D. Yang, "H-mine : Hyper structure Mining of Freuqent patterns in Large database", In proc. of International conference on Data Mining , San Jose, California, Novermber 29-December 2, 2001.*