Temporal Approach to Association Rule Mining Using T-tree and P-tree

Keshri Verma, O. P. Vyas, Ranjana Vyas

School of Studies in Computer Science
Pt. Ravishankar Shukla university Raipur Chhattisgarh – 492010, India
seema2000_2001@yahoo.com, opvyas@rediffmail.com,
vyasranjana@yahoo.com

Abstract. The real transactional databases often exhibit temporal characteristic and time varying behavior. Temporal association rule has thus become an active area of research.A calendar unit such as months and days, clock units such as hours and seconds and specialized units such as business days and academic years, play a major role in a wide range of information system applications. The calendar-based pattern has already been proposed by researchers to restrict the time-based associationships. This paper proposes a novel algorithm to find association rule on time dependent data using efficient T tree and P-tree data structures. The algorithm elaborates the significant advantage in terms of time and memory while incorporating time dimension. Our approach of scanning based on time-intervals yields smaller dataset for a given valid interval thus reducing the processing time. This approach is implemented on a synthetic dataset and result shows that temporal TFP tree gives better performance over a TFP tree approach.

1 Introduction

Efficient algorithm for finding frequent patterns has been one of the key success stories of data mining. The Apriori algorithm [1] by Agrawal et al is based on support-confidence framework. It begins with counting the support of each and every item of transaction table. It comprises of two phases; (1) Candidate generation in all possible combinations. (2) The database scanning and counting of all transactions for each itemset. The process continues as long as frequent item is available. There are some pertinent questions which can be raised like - What about mining the -

- Associationship in all patterns of a certain type during a specific time interval.
- Associationship in all patterns of a certain type with a specific periodicity.
- Associationship in all patterns of certain type with a specific periodicity during a specific time interval.

All these statements indiates that data is dependent on time. Time plays an important role in real dataset. In various business setups such as stock market and share market time is most important dimension In existing algorithms like Apriori [1] and Fp-growth[7] when time aspect is involved in the dataset, it provides useful infor-

mation for business, but it increases the time complexity because it is needful to scan the database for every valid specific interval. Though Fp-growth is one of the well-known approaches for finding frequent itemset, but it performs miserably when the nature of data is sparse. Sparse data creates large number of nodes and if temporal dimension is also incorporated on dataset, it again increases the branches of tree and is thus difficult to fit in memory. In such case Apriori outperforms Fp-growth approach.

Our proposed approach is aimed at devising an efficient algorithm for mining association rule in time-based dataset by using efficient data storage mechanism- Temporal T-tree.

Definition 1: A temporal pattern is a triplet <pattern , periodicexp, interval exp>, where pattern is a general pattern which may be a trend, a classification rule, an association a causal relationship etc. Periodicexp is a periodic time expression or a special symbol p_null with φ (p_null) being {T} and intervalexp is a general interval expression or a special symbol I_null with φ (p_null) being {T}. It expresses that pattern hold during each interval in φ (periodicexp ^{0} intervalexp). T is the time domain. [10]

Given a time-stamped dataset D over a time domain T, the problem of mining temporal pattern of a certain type is to discover all pattern of the form <pattern, periodicexp, intervalexp> in D which satisfy all the user defined threshold with respect to described minimum frequency min_f% with some given condition.

Three properties of the algorithm are of interest when considering its performance:

- The number of database access required
- The no. of computational step required in counting subset of records
- The memory requirements.

For small values of n these set of algorithm are appropriate, but large dataset these algorithms are computationally infeasible.

The rest of the paper is organized as follows: Section 2, discusses some related works. In section 3 defines temporal association rule in term of calendar schema. In Section 4 elaborate the proposed work, section 5 shows the experimental study and section 6 elaborates conclusion and future works and section 6 provides application of above investigation.

2 Related Work

The concept of association rule was introduced as Apriori algorithm [1]. Its performance was improved by deploying frequent-pattern growth approach [7]. In paper [6] the omission of the time dimension in association rule was very clearly mentioned. A temporal aspect of association rule was given by Juan [5]. According to this the transactions in the database are time stamped and time interval is specified by the user to divide the data into disjoint segments, like month, days, and years. Further the cyclic association rule was introduced by Ozden [6] with minimum support and high confidence. Using the definition of cyclic association rule, it may not have high support and confidence for the entire transactional database. A nice bibliography of temporal

data mining can be found in the Roddick literature[8]. Rainsford and Roddick presented extension to association rules to accommodate temporal semantics. According to [9] logic the technique first searches the associationship than it is used to incorporate temporal semantics. It can be used in point based and interval based model of time simultaneously[9]. A Frequent pattern approach for mining the time sensitive data was introduced in [4] where the pattern frequency history under a tilted-time window framework is used 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 are updated incrementally with incoming transactions [4]. Li et. al. addresses the calendar based association rule problem [11],the result shows temporal Apriori is 5 to 22 times faster than direct Apriori, for fuzzy match temporal Apriori is 2.5 to 12 times faster than direct Apriori and the execution time extremely decreases with respect to precise match or fuzzy match.

3 Problem Definition

3.1 Association Rule

The concept of association rule, which was motivated by market basket analysis and was originally presented by Agrawal [1]. Given a set of T of transaction, an association rule of the form X Y is a relationship between the two disjoint itemsets X and 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 contains 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 [7]. In our opinion this mining associationship will become more useful if we include the time factor in to it.

3.2 Temporal Association Rule

Definition 2: The frequency of an 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 specifiedtime 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 and 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 and week they 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

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

The calendar schema is implemented by applying Apriori algorithm [11]. It follows the candidate generation approach in order to mine the frequent item. We assist here that total tree construction from partial tree is an efficient approach for mining time based associated items. It first constructs a partial tree (P tree). A P-tree is a set enumeration tree structure in which to store partial counts for item sets. The top, *single attribute*, level comprises an array of references to structures of the form shown to the right, one for each column [12]. Each branch indicates the association ship of item. It reduces the size of dataset and increases the performance and 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.

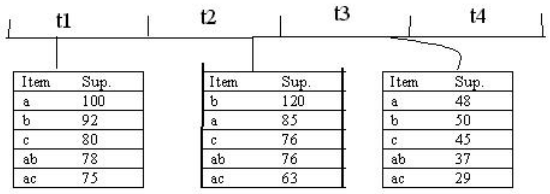

Fig. 1. Frequent pattern in different interval

4.1 Partial Support [2]

Most of the existing methods described above proceed essentially on each database pass by defining some candidate set and then examining each record to identify all the members of the candidate set that are subsets of the record, incrementing a support-count for each. The computational cost of this increases with the density of information in database records, i.e. when the average number of attributes present in a record is high, leading to an exponential increase in the number of subsets to be considered, and when candidate sets are large. In principle, however, it is possible to reduce this cost of subset- counting by exploiting the relationships between sets of items. For example, in the simplest case, a record containing the attribute set *ABD* will cause incrementation of the support-counts for each of the sets *ABD*, *AB*, *AD*, *BD*, *A*, *B* and *D*. Strictly, however, only the first of them is necessary, since a level of support for all the subsets of *ABD* can be inferred subsequently from the support-count of *ABD*.

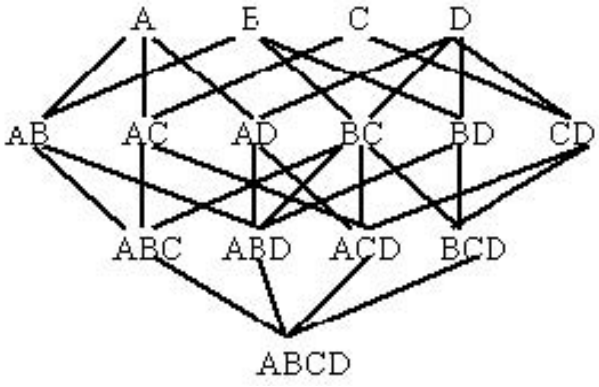

Fig. 2. Lattice of item {A,B,C,D}

Let *i* be a subset of the set *I* (where *I* is the set of *n* attributes represented by the database). We define *Pi*, the *partial support* for the set *i*, to be the number of records whose contents are identical with the set *i*. Then *Ti*, the *total support* for the set *i*, can be determined as:

$$T_i = \sum P_j (\forall j, j \supseteq i) \qquad (1)$$

This allows us to postulate a general algorithm for computing total supports. Let P be the set of partial support counts Pi corresponding to sets i which appear as records in the database, and T be the set of total support counts in which we are interested (however this is defined). With the members of P and T initialized to zero.

Algorithm A

```
Inputs: Transaction  DS, countset P
Output: Returns P and T counting sets in DS
Method:
          A1:   ∀ Records j in DS do
                begin add 1 to Pj
                   insert j to P
                end;
          A2:   ∀ j in P do
                begin . i in T, i ⊆. j do
                   begin add Pj to Ti
                   end;
                end;
```

For a database of m records, stage 1 of the algorithm (A1) performs m support-count incrementations in a single pass, to compute a total of m partial supports, for some $m' \leq m$. The second stage of the algorithm (A2) involves, for each of these, the examination of subsets which are members of the target set T . In an exhaustive version of the method, T will be the full set of subsets of I. Computing via summation of partial supports, however, offers three potential advantages. Firstly, when n is small ($2^n << m$), then A2 involves the summation of a set of counts, which is significantly smaller than a summation over the whole database. Secondly, even for large n, if the database contains a high degree of duplication ($m' << m$) then the stage 2 summation will again be significantly faster than a full database pass, especially if the duplicated records are densely-populated with attributes. Finally, and most generally, we may use the stage A1 to organize the partial counts in a way which will facilitate a more efficient stage 2 computation, exploiting the structural relationships inherent in the lattice of partial supports.

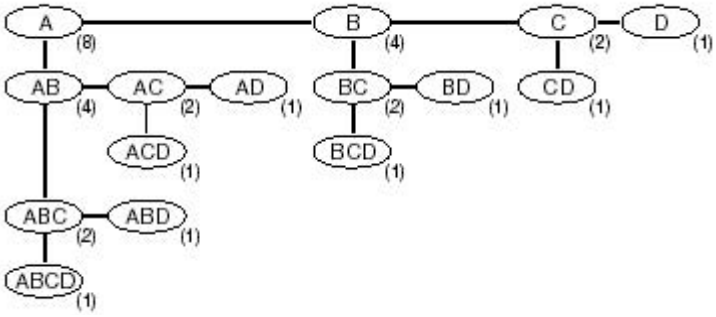

Fig. 3. Tree Storage of Subset (A,B,C,D)

Figure 3 shows an alternative representation of the sets of subsets of I , for I = {*A, B,C, D*}, in the form of Rymon's [13] set enumeration tree. In this structure, each subtree contains all the supersets of the root node, which follow the root node in lexicographic order.

4.2 Algorithm for Generating Calendar Based Temporal Association Rule Using TFP Tree

The proposed algorithm first extract the data of particular interval from whole data set and apply the TFP Mining approach to find frequent itemset on that specific intervals.

*Input:*A Transaction Database D, Specified calendar pattern <dd,mm,yy>
Output: Frequent item set, Temporal database table TDB
Method:

Step (1) Set pointer to first record of database
Step (2) Scan the Database one by one and follow the Step(3)
Step (3) {
Step (4) If data ∈ <dd,mm,yy>
Step (5) Send the data into Temporal database Table TDB
Step (6) }
Step (7) set K = 1
Step (8) Build level *K* in the *T* -tree.
Step (9) "Walk" the *P*-tree, applying algorithm TFP to add interim supports associated with individual *P*-tree nodes to the level *K* nodes established in (2).
Step (10) Remove any level *K* T-tree nodes that do not have an adequate level of support.
Step (11) Increase K by 1.
Step (12) Repeat steps (8) through (11); until a level K is reached where no nodes are adequately Supported.

In above algorithm step (1) to step (5) used to find out the itemset, which occurs on valid time period specified by calendar schema. Step (7) to step (12) used for mining frequent itemset from TFP tree.

5 Experimental Observation

In this section we present the experimental result showing the performance of TFP tree approach and temporal TFP tree approach. The experiments were performed on the synthetic data set based on KDD cup T20I10D250kN500. The Pentium III with 128 MB Main Memory, 20 GB hard disk having Microsoft windows was used. Algorithms were implemented in C++ and Java. Figure 4(a) and 4(b) shows the comparative graph for different time intervals. The execution time taken by CPU in TFP algorithm is almost three times more than temporal TFP algorithm, which is a significant

improvement and also shows that the performance of temporal TFP algorithm steadies as the support of itemset grows.

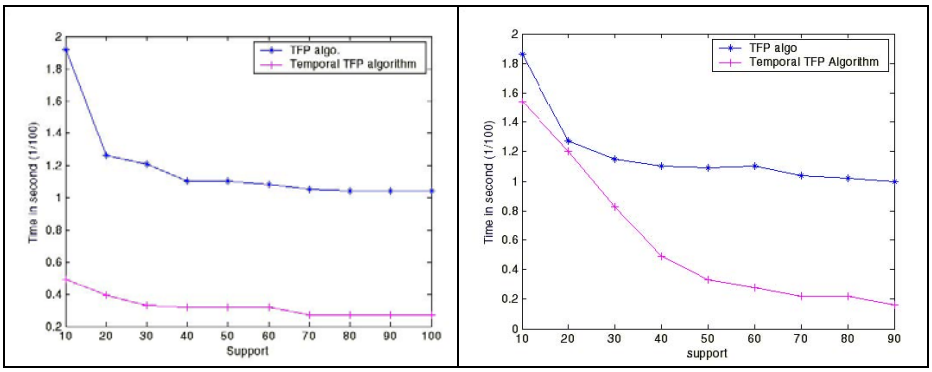

Fig. 4. Comparative graph of TFP tree and Temporal TFP tree

6 Conclusion and Future Work

In real world data the knowledge used for mining is always time varying. Real transactional databases specially exhibit temporal characteristic and time varying behavior. For example, in Telecommunication Data Analysis, the calling pattern may vary with time. Similarly for Market Basket Analysis the associationships between various items may change with time and also that this transaction Database may have either a sparse or dense database. In this paper we have adopted time sensitive approach and presented an algorithm for mining frequent itemset on specified time interval using TFP approach. Frequent items generated for specific time interval have great impact from the databases. It has the capability to affect all aspects of doing business in today's world. It will empower decision makers with realistic results and that too with more accuracy and reduced time lag. It will thus help in more realistic and relevant decision-making. An inherent advantage of using P- tree and T- tree structure is in having branches which can be considered independently and therefore the structures can be rapidly adapted for use in parallel / distributed Association Rule Mining.

References

1. R. Agrawal and R. Srikant , R : Fast algorithm for mining association rule. In VLDB'94 Chile , Sept (1994) ,pp –487-499.
2. Coenen, F.P., Goulbourne, G. and Leng, P.H. (2001). Computing Association Rules Using Partial Totals. In de Raedt, L. and Siebes, A. (Eds), Principles of Data Mining and Knowledge Discovery, Proc PKDD 2001, Spring Verlag LNAI 2168, pp 54-66.

3. Frans Coenen, Paul Leng, and Shakil Ahmed, Data Structure for Association Rule Mining : T-tree and P- Tree, IEEE transaction on Knowledge Discovery and Data Engineering, Vol 16, No 6 ,(2004).
4. 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).
5. Juan M .Ale , Gustavo H. Rossi R: An approach to discovering temporal association rules, ACM SIGDD March 1..21,(2002).
6. Banu Ozden , Sridhar Ramaswamy , Avi Silberschatz R: "Cyclic Association Rule" ,In Proc. Of forteenth International conference on Data Engineering (1998), pp 412-425.
7. Jian Pei, Jiawei Han, Yiwen Yin and Running Mao R : Mining Frequent Pattern without Candidate GenerationProc. ACM-SIGMOD Int'l Conf. Management of Data, pp. 1-12, (2000).
8. John F. Roddick, Kathleen Hornsby, Myra Spiliopoulou: An Updated Bibliography of Temporal, Spatial, and Spatio-temporal Data Mining Research. TSDM 2000: pp147-164.
9. Chris P. Rainsford, John F. Roddick R: Adding Temporal semantics to association rule, 3rd International conference KSS Springer 1999, pp 504-509.
10. Xiandong Chen, Iiias Petrounian, Book "Knowledge Discovery and Data Mining" Chapter 5 " A Development Framework of Temporal data Mining", pp 93,2001.
11. Yingjiu Li, Peng Ning, X. Sean Wang , Sushil Jajodia. Discovering calendar-based temporal association rules, Data and Knowledge Engineering volume 4,Elesvier publisher, Volume 44 pp– 193-214 ,(2003).
12. Frans Coenen, Paul Leng, and Shakil Ahmed , Data Structure for Association Rule Mining : T-tree and P- Tree, IEEE transaction on Knowledge Discovery and Data Engineering, Vol 16, No 6 ,(2004).
13. R. Rymon,.Search Through Systematic Set Enumeration, Proc. Third Int'l Conf. Principles of Knowledge and Reasoning, pp. 539-550, (1992).