Secular Form to Affiliation Rule Mining Employing P-tree and T-tree

Aparna Agrawal  
Lecturer Dept of Computing,  
Middle East College, Muscat, Oman

Deevyankar Agarwal  
Lecturer, Dept of Engineering, Higher College of Technology, Muscat, Oman

DOI: 10.23956/ijarcsse/V7I8/0146

Abstract: The real commercialism information often demonstrates temporal feature and time varying behavior. Temporal affiliation rule has thus got an active area of explore. A calendar part such as months and days, clock parts such as hours and seconds and differentiated units such as business days and academic years, act a major role in a wide range of information system applications. The calendar-based form has already been proposed by explorers to restrict the time-based association ships. This paper advises a novel algorithmic program to determine association rule on time dependent data employing effective T tree and P-tree data structures. The algorithm complicates the significant advantage in terms of time and memory while comprising time dimension. Our approach path of scanning based on time-intervals yields litter information set for a given valid interval thus cutting down the processing time. This approach is enforced on a synthetic data-set and result shows that temporal TFP tree collapses better performance over a TFP tree access.

Keywords: temporal, P-Tree and T-Tree

I. INTRODUCTION

Effective algorithm for determination frequent forms has been one of the key success stories of data mining. The Apriori algorithm [1] by Agrawal et al is founded on support-confidence example. It commences with counting the affirm of each and every item of transaction table. It constitutes of two phases; (1) Candidate contemporaries in all possible compounding. (2) The information scanning and counting of all deals for each item set. The process covers as long as frequent item is available. There are some apt questions which can be raised like - What about mining the

- Associationship in all forms of a certain type throughout a specific time interval.
- Associationship in all forms of a certain type with a specific periodicity.
- Associationship in all forms of certain type with a particular periodicity throughout a specific time interval.

All these arguments indicate that data is contingent upon time. Time acts an significant role in real information set. In various business concern setups such as stock market and share market time is most significant dimension In existing algorithmic programs like Apriori [1] and Fp-growth [7] when time expression is demanded in the information set, it provides useful information for business, but it increases the time complexity because it is needful to scan the information for every valid specific interval. Though Fp-growth is one of the well-known approaches for detecting frequent item set, but it performs miserably when the nature of data is sparse. Sparse data produces large number of clients and if temporal dimension is also comprised on information set, it again increases the branches of tree and is thus unmanageable to fit in memory. In such case Apriori outperforms Fp-growth approach.

Our suggested approach is aimed at devising an effective algorithm for mining as-association rule in time-based information set by using effective data storage mechanism- Temporal T-tree.

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

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

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

- The number of information access required
- The no. of computational step commanded in counting subset of records
- The memory necessities.

For small measures of n these set of algorithm are reserve, but large information set these algorithms are computationally unworkable.
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 outline. In Section 4 elaborate the proposed work, section 5 demonstrates the experimental study and section 6 elaborates conclusion and future works and section 6 furnishes application of above investigation.

II. RELATED WORK

The conception of association rule was brought in as Apriori algorithm [1]. Its execution was improved by spreading frequent-form growth approach [7]. In paper [6] the omission of the time dimension in association rule was very clearly observed. A temporal aspect of association rule was given by Juan [5]. According to this the transactions in the information are time stamped and time interval is assigned by the user to divide the data into disjoint segments, like month, days, and years. Further the cyclic association rule was acquainted by Ozden [6] with lower limit support and high confidence. Using the definition of cyclic association rule, it may not have high support and assurance for the entire transactional information. A nice bibliography of temporal data mining can be detected in the Roddick literature [8]. Rainsford and Roddick presented extension to association rules to adapt temporal semantics. According to [9] logic the technique first explores the associationship than it is wont to comprise-rate temporal semantics. It can be applied in point based and interval based example of time simultaneously [9]. A Frequent formaccess for mining the time sensitive data was brought in in [4] where the format history under a tilted-time window example is used to answer time-sensitive queries. A collection of item forms along with their ratio histories are compacted and stored using a tree structure similar to FP-tree are updated incrementally with entering transactions [4]. Li et. al. addresses the calendar based association rule problem [11], the result demonstrates 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.

III. PROBLEM DEFINITIONS

3.1 Association Rule

The conception of association rule, which was prompted by market basket psychoanalysis and was originally demonstrated by Agrawal [1]. Given a set of T of deal, an association rule of the form X Y is acquaintanceamong the two disjoint item sets X and Y. An association rule meets some user-given demands. The support of an item set by the set of deal is the fraction of transaction that contains the item set. An item set is said to be large if its support exceeds a user-given threshold minimum support. The confidence X Y over T is a transaction comprising X and also comprising Y. Due to complex candidate generation in the data set Jiwei Han invented a new proficiency of FP-growth method for mining frequent form without candidate generation [7]. In our opinion this mining association ship will become more utile if we include the time factor in to it.

3.2 Temporal Association Rule

Definition 2: The ratio of an item set over a time period T is the number of transactions in which it happens 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) = Ratio 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 indicate the valid start time and T_e indicate valid time according to temporal data.

3.3 Simple Calendar Based Form

When temporal information is applied in terms of date, month, year and week they form the term outline. It is introduced in temporal data mining. A calendar outline is a relational outline (in the sense of relational information’s) R = (T : D, F : D,…….F : d) together with a valid constraint. A calendar outline (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 form, the branch e cover e’ in the same calendar outline if the time interval e’ is the subset of e and they all follow the same form. If a calendar form<d_1, d_2,……d_i> covers another form<d’_1, d’_2,…….d’_i> if and only if for each i, 1 <= i <= n or d_i = d’_i. Now our task is to mine frequent form over arbitrary time interval in terms of calendar form outline.

IV. PROPOSED WORK

The support of information set in the data warehouse can be maintained by dividing it into dissimilar intervals. The support of a item in interval t1 cannot be the same in interval t2. An infrequent or less support item in interval t1 can be frequent item in interval t2.

The calendar outline 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 effective 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 information set 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 changed from \( t_4 \) to \( t_1 \).

<table>
<thead>
<tr>
<th>Item</th>
<th>Sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>100</td>
</tr>
<tr>
<td>b</td>
<td>92</td>
</tr>
<tr>
<td>c</td>
<td>80</td>
</tr>
<tr>
<td>ab</td>
<td>78</td>
</tr>
<tr>
<td>ac</td>
<td>75</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Item</th>
<th>Sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>120</td>
</tr>
<tr>
<td>b</td>
<td>85</td>
</tr>
<tr>
<td>c</td>
<td>76</td>
</tr>
<tr>
<td>ab</td>
<td>76</td>
</tr>
<tr>
<td>ac</td>
<td>63</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Item</th>
<th>Sup.</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>48</td>
</tr>
<tr>
<td>b</td>
<td>50</td>
</tr>
<tr>
<td>c</td>
<td>45</td>
</tr>
<tr>
<td>ab</td>
<td>37</td>
</tr>
<tr>
<td>ac</td>
<td>29</td>
</tr>
</tbody>
</table>

Fig. 1. Frequent form in different interval

4.1 Partial Activity

Most of the existent methods described above proceed fundamentally on each information pass by defining some candidate set and then analyzing 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 these gains with the density of information in information records, i.e. when the average number of attributes demonstrate 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 information). We fix \( P_i \), the partial support for the set \( i \), to be the number of records whose capacities are identical with the set \( i \). Then \( T_i \), the total support for the set \( i \), can be determined as:

\[
T_i = \sum_{j} P_j \left( \forall j, j \supseteq i \right)
\]

This allows us to contend a general algorithm for computing total supports. Let \( P \) be the set of partial support considers \( P_i \) corresponding to sets \( i \) which appear as records in the information, and \( T \) be the set of total support counts in which we are concerned (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:**

\[
A1: \forall \text{ Records } j \text{ in } DS \text{ do}
\begin{align*}
& \text{begin}\text{ add } 1 \text{ to } P_j \\
& \text{insert } j \text{ to } P
\end{align*}
\]

\[
A2: \forall \text{ } j \text{ in } P \text{ do}
\begin{align*}
& \text{begin } \forall i \text{ in } T, i \subseteq j \text{ do}
\end{align*}
\]

begin add \( P_j \) to \( T \)
end
end

© www.ijarcsse.com, All Rights Reserved
For an information of \( m \) records, stage 1 of the algorithm (A1) executes \( m \) support-count accusse in a single pass, to compute a total of \( m \) partial supports, for some \( m \leq m \). The second stage of the algorithm (A2) demands, for each of these, the investigation of subsets which are members of the target set \( T \). In an exhaustive interpretation of the method acting, \( T \) 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 accumulation of a set of counts, which is significantly smaller than a summation over the whole information. Secondly, even for large \( n \), if the information contains a high degree of duplication (\( m' \) \( < m \)) then the stage 2 summation will again be significantly faster than a full information 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 help a more effective stage 2 computation, exploiting the structural relationships inherent in the lattice of partial supports.

![Tree Storage of Subset (A,B,C,D)](image)

Figure 3 demonstrates an alternative representation of the sets of subsets of \( I \), for \( I = \{A, B, C, D\} \), in the form of Rymon’s [13] set numbering 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 Giving Calendar Based Temporal Association Rule Employing TFP Tree

The proposed algorithmic program first extracts the data of particular interval from whole data set and apply the TFP mining approach path to find frequent itemset on that specific intervals.

**Input:** A Transaction Information \( D \), Specified calendar form <dd, mm, yy>

**Output:** Frequent item set, Temporal information table TDB

**Method:**

1. Set pointer to first record of information
2. Scan the Information one by one and follow the Step(3)
3. Step (3) {
4. If data \( \in <dd, mm, yy> \)
5. Send the data into Temporal information Table TDB
6. }
7. Set K = 1
8. Build level K in the \( T \)-tree.
9. “Walk” the \( P \)-tree, applying algorithm TFP to add interim sup-ports associated with individual \( P \)-tree nodes to the level \( K \) nodes established in (2).
10. Remove any level \( K \) \( T \)-tree nodes that do not have an adequate level of support.
11. Increase K by 1.
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 outline. Step (7) to step (12) used for min-ing frequent item set from TFP tree.

### V. EXPERIMENTAL REFLECTIONS

In this section we demonstrate the experimental result showing the execution of TFP tree approach and temporal TFP tree approach path. 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 relative graph for different time intervals. The execution time taken by CPU in TFP algorithm is almost three times more than temporal TFP algorithmic program, which is a significant improvement and also shows that the functioning of temporal TFP algorithm steadies as the support of itemset grows.

© www.ijarcsse.com, All Rights Reserved
VI. CONCLUSIONS AND FUTURE WORK

In real-world data, the knowledge used for mining is always time changing. Real transactional information particularly exhibit temporal characteristics and time-varying behavior. For example, in Telecommunication Data Analysis, the calling form may vary with time. Similarly, for Market Basket Analysis, the associations among various items may alter with time and also that this transaction Information may have either sparse or dense information. In this paper, we have adopted a time-sensitive approach and demonstrated an algorithm for mining frequent itemsets on specified time intervals using the TFP approach. Frequent items generated for specific time intervals have great impact on the information's. It has the capability to affect all aspects of behaving 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 structures is in having branches which can be considered severally and therefore the structures can be rapidly adapted for use in parallel/distributed Association Rule Mining.

REFERENCES

[7] Jian Pei, Jiawei Han, Yiwen Yin and Running Mao R : Mining Frequent Form without Candidate GenerationProc. ACM-SIGMOD Int’l Conf. Management of Data, pp. 1-12, (2000).
