Time-Profiled Association Mining under the Constraints of Similarity

1Miss. S. R. Bhangde, 2Prof. Dr. S.P. Sonavane
Department of computer science and Engineering
Walchand college of Engineering
Sangli, India
email: sonud12@gmail.com¹, shefali.sonavane@gmail.com²

Abstract - This paper presents the algorithm to discover all associated item sets whose prevalence variation over time are similar to the given query sequence of interest over time under a threshold from time stamped transaction database. Traditional association mining uses simple numeric interest measure (e.g. Support) and simple subset specification such that support is greater than a certain threshold. In contrast the time-profiled association mining under the constraints uses a composite interest measure that describes a discrepancy degree between its n-dimensional prevalence value sequence from a temporally divided data set and a given query sequence. The proposed algorithm substantially reduces the item set search space by exploiting interesting properties such as upper and lower bound support time sequences and lower bounding distance for early pruning candidate item sets.

Keywords: temporal data mining; temporal association patterns; support time sequence; similarity

I. INTRODUCTION

The Time-profiled association mining under the constraints of similarity discovers all associated item sets whose prevalence variations over time are similar to the given query sequence of interest over time under a threshold from time stamped transaction database. The dissimilarity degree of the sequence of support values of an item set to the query sequence is used to capture how well its temporal prevalence variation matches the query pattern. Time-profiled association mining under the similarity constraints can be used to discover interacting relationships consistent with the query prevalence sequence over time. It can reveal interesting relationships of data items that co-occur with a particular event over time.

Following the work of Agarwal and Srikant [1], association mining has been extensively studied in [2], and [3]. In particular, in [4],and [5], they have paid attention to temporal information, which is implicitly related to transaction data, e.g., the time that a transaction is executed, and discovered association patterns that vary over time. However, current methods for temporal mining cannot reveal the similarity –based temporal patterns that are based on actual prevalence similarity[6].

Traditional association analysis uses a simple subset specification such that support is greater than a given threshold. In contrast, the time-profiled association mining under similarity constraints can use a richer subset specification, which consists of a query time sequence, a time sequence similarity function and a dissimilarity threshold.

Application examples: Some applications for similarity based temporal patterns include

- Climate data analysis: In the scientific application domain, Earth scientists have been interested in the behavior of climates in a region that are often influenced by the El Niño phenomenon, an abnormal warming in the eastern tropical Pacific Ocean [7]. If we consider the El Niño related index values over the last 10 years, e.g., the Southern Oscillation Index (SOI) [7], as a reference sequence, one example of a time-profiled association under similarity constraint is a climate event pattern of low precipitation and low atmospheric carbon dioxide in Australia whose co-occurrence over time is similar to the fluctuation of the El Niño index sequence.

- Website click stream data analysis: Let us consider the online website Weather.com, which offers weather-related lifestyle information including travel, driving, health, home and garden, fashion and beauty ,and sporting events, as well as weather information [8]. The website reports that almost 40 percent of weather.com visitors shop home improvement products when temperatures rise [8]. The website may attract more advertisers if it can analyze the relationships of visited websites through weather.com with changes of weather.

Figure 1 shows an example of time-profiled association mining under similarity constraints with a simple time stamped transaction database with time slots t₁ and t₂.

II. PROBLEM STATEMENT

Input:
1. A set of items E = {e₁,…., eₙ}.
2. A time stamped transaction database D = D₁∪……∪Dₙ, Dᵢ ∩ Dⱼ = φ , i ≠ j, wherein each
Time-Proﬁled Association Mining Under the Constraints of Similarity

Fig. 1. An example of Time-proﬁled association mining under the constraints of similarity: (a) input data, (b) generated prevalence (support) time sequence and sequence search, and (c) output item sets.

Transaction \( d \in D \) is a tuple \(<\text{timestamp}, \text{item set}>\), where timestamp is a time that the transaction \( D \) is executed and item sets is the sets of items which is subset of \( I \). \( D_t \) is a set of transactions in time slot \( t \).

An interest time period \( T \) which is divided into set of discrete time slots, \( T = t_1 \cup \ldots \cup t_n \), \( t_i \cap t_j = \emptyset \), \( i \neq j \).

A subset speciﬁcation

a. query sequence \( Q = \{ q_1, \ldots, q_n \} \) over time slots \( t_1, \ldots, t_n \)

b. a time series similarity function \( f_{\text{similarity}}(\overline{A},\overline{B}) \rightarrow IR^p \), where \( A \) and \( B \) are numeric sequences and

c. a dissimilarity threshold \( \theta \)

**Expected output:** A result set of item sets \( I \subseteq E \) that satisfy the given subset speciﬁcation, i.e.,

\[ f_{\text{similarity}}(\overline{S_t},\overline{Q}) \leq \theta, \]  

where \( \overline{S_t} = \{ s_1, \ldots, s_n \} \) is the sequence of support values of item sets over time slots \( t_1, \ldots, t_n \).

**Objectives:**

1. To ﬁnd complete and correct result set
2. To minimize the computational cost

**III. PROPERTIES OF TIME-PROFILED ASSOCIATION MINING ALGORITHM**

a) Basic Concepts.

1) Choice of Similarity measure: Different dissimilarity measures have been discussed in the time series database literature [9]. This paper proposes using a \( L_p \) norm (\( p=1,2,\ldots,\infty \)) based similarity function since it is the most popular class of dissimilarity measures and is used in many applications.

**Definition 1**: For two time sequences \( \overline{A} = <a_1, \ldots, a_n> \) and \( \overline{B} = <b_1, \ldots, b_n> \), the \( L_p \) norm between \( \overline{A} \) and \( \overline{B} \) is defined as

\[
L_p(\overline{A}, \overline{B}) = \left( \sum_{t=1}^{n} |a_t - b_t|^p \right)^{1/p} \quad (1)
\]

When \( p=1 \), it is called city block or Manhattan norm. \( L_1 \) norm is optimal when measurement errors are additive, i.i.d. Laplacian (or Double Exponential), hence more robust against impulsive noise [10].

When \( p=2 \), it is called as Euclidean distance. The work proposed here uses the \( L_2 \) norm based similarity measure. It is known that Euclidean distance is optimal (in the Maximum Likelihood sense) when measurement differences are additive, i.i.d. (independent, identically distributed) Gaussian [11]. It has been the most popular similarity measure in similar time sequence matching [11]. It can be reformulated as

\[
L_2(\overline{A}, \overline{B}) = \left( \sum_{t=1}^{n} (a_t - b_t)^2 \right)^{1/2} \quad (2)
\]

The disadvantage of \( L_p \) norm distances is that there is no obvious bound value of the maximum dissimilarity distance. It is hard to infer from their value whether the degree of dissimilarity is small or large.
Time-Profiled Association Mining under the Constraints of Similarity

Thus, the normalized $L_p$ norm distances that are divided by the number of time points can be used.

2) Support time sequence: Given a time-stamped transaction database $D = D_1 \cup \ldots \cup D_n$, where $D_i$ is a set of transactions executed in $i$th time slot, $1 \leq i \leq n$, the support of an item set $I$ at a time slot $t_i$ is the fraction of transactions $d$ in $D_i$ that contain the item set $I$ such that

$$\text{Support}(D_i, t_i) = \frac{|\{d \in D_i | I \subseteq d\}|}{|D_i|} \quad (3)$$

The support time sequence of an item set $\bar{S} = \{s_1, \ldots, s_n\}$ is the time sequence of support values of an item set $I$ over time slots $t_1, \ldots, t_n$ such that

$$\bar{S} = \langle \text{Support}(D_1, t), \ldots, \text{Support}(D_n, t) \rangle$$

For example, Fig. 1b shows the support time sequences of possible item sets from the data set in Fig. 1a.

b) Upper bound and lower bound support time sequence

Generating the support time sequences of item sets is the core operation in time-profiled association mining algorithm under the constraints of similarity. This paper proposes a method for estimating support time sequences without examining an input data set. The upper bound and lower bound of the support time sequence of the item set is defined using the support time sequence of its subsets.

Theorem 1: Let $D$ be a transaction data set and $I$ be an item set, $\text{support}(I, D) \in [L(I, D), U(I, D)]$ with

$$L(I, D) = \max \{ \sigma_I(J, D) : 0 \leq |J| \leq |I| \text{ and } |J| \text{ is even} \},$$

$$U(I, D) = \min \{ \sigma_I(J, D) : 0 \leq |J| \leq |I| \text{ and } |J| \text{ is odd} \},$$

where $\sigma_I(J, D) = \sum_{I \subseteq J} (-1)^{|I-J|} \cdot \text{support}(J, D)$.

$L(I, D)$ means a lower bound of $\text{support}(I, D)$, and $U(I, D)$ means an upper bound of $\text{support}(I, D)$. The proof of tight bound is described in [12]. The proposed algorithm uses this set of rules to derive the tight upper bound and lower bound of the support time sequence of an item set.

For example, consider the bounds of support of an item set $I=\{A,B\}$. Suppose the supports of all subsets of $I$ are $\text{support}()=1$, $\text{support}({A})=0.5$ and $\text{support}({B})=0.2$. By Theorem 1,

$$\sigma_I(\{\}) = (-1)^3 \cdot \text{support}(\{} + (-1)^2 \cdot \text{support}(\{A\}) + (-1)^2 \cdot \text{support}(\{B\})$$

$$\text{support}(\{A\})=1.05 = 0.5, \text{ and } \sigma_I(\{B\}) = (-1)^2 \cdot \text{support}(\{A\})=1.02 = 0.2.$$ Among all subsets of $I$, $\{A\}$ and $\{B\}$ are the subsets whose size is odd, thus

$$U(I, D)=\min(\sigma_I(\{\}), \sigma_I(\{B\}))=\min(0.5,0.2)=0.2$$

Since, $\{\}$ is a subset of $I$ whose size is even

$$L(I, D)=\max(\sigma_I(\{\}), 0) = \max(-0.3,0)=0$$

Thus, $0 \leq \text{support}(\{A,B\}) \leq 0.2$.

Definition 2: Let $D = D_1 \cup \ldots \cup D_n$ be a time-stamped transaction data set. The lower bound support time sequence of an item set $I$, $\bar{L}_I$, and the upper bound support time sequence of $I$, $\bar{U}_I$, are defined as follows

$$\bar{L}_I = \langle l_1, \ldots, l_n \rangle = \langle U(I, D_1), \ldots, U(I, D_n) \rangle,$$

$$\bar{U}_I = \langle u_1, \ldots, u_n \rangle = \langle U(I, D_1), \ldots, U(I, D_n) \rangle.$$
c) **Lower Bounding Distance**

If the lower bounding distance of an item set does not satisfy the dissimilarity threshold, its true distance also does not satisfy the threshold. Thus, the lower bounding distance can be used to prune item sets early without the computation of true distance.

**Definition 3:** For a query sequence \( \overline{Q} \) of an item set, let \( \overline{Q}^U = <q_1, \ldots, q_k> \) be a subsequence of \( \overline{Q} \) and \( \overline{U}^L = <u_1, \ldots, u_k> \) be a subsequence of \( \overline{U} \), where \( q_i > u_i, 1 \leq t \leq k \). The **upper lower bounding distance** between \( \overline{Q} \) and \( \overline{U} \), \( D_{ULB}(\overline{Q}, \overline{U}) \), is defined as \( D(\overline{Q}^U, \overline{U}^L) \).

**Definition 4:** For a query sequence \( \overline{Q} \) of an item set, let \( \overline{Q}^U = <q_1, \ldots, q_k> \) be a subsequence of \( \overline{Q} \) and \( \overline{U}^t = <u_1, \ldots, u_k> \) be a subsequence of \( \overline{U} \), where \( q_i > u_i, 1 \leq i \leq k \). The **lower lower bounding distance** between \( \overline{Q} \) and \( \overline{U} \), \( D_{Llb}(\overline{Q}, \overline{U}) \), is defined as \( D(\overline{Q}^U, \overline{U}^t) \).

**Definition 5:** For a query sequence \( \overline{Q} \), the upper bound support sequence \( \overline{U}^U \), and the lower bound support sequence \( \overline{U}^L \) of an item set, the lower bounding distance \( D_{LB}(\overline{Q}, \overline{U}^U, \overline{L}^U) \) is defined as \( D_{ULB}(\overline{Q}, \overline{U}) + D_{Llb}^{LB}(\overline{Q}, \overline{L}) \).

**Lemma 1:** For the upper bound support time sequence \( \overline{U} = <u_1, \ldots, u_n> \), the lower bound support time sequence \( \overline{L} = <l_1, \ldots, l_n> \), support time sequence \( \overline{S} = <s_1, \ldots, s_n> \) of an item set \( I \), and a query sequence \( \overline{Q} = <q_1, \ldots, q_n> \), the lower bounding distance \( D_{LB}(\overline{Q}, \overline{U}, \overline{L}) \) and the true distance \( D(R, S) \) hold the following inequality.

\[
D_{LB}(R, U, L) \leq D(R, S).
\]

d) **Monotone Property of Upper Lower Bounding Distance**

**Observation 1.** The \( l_p \) norms between the support time sequence of an item set and a query sequence do not show any monotonicity with the size of the item set.

For example, Fig. 2a shows the Euclidean distances between the support time sequences of \( \{C\}, \{A, C\} \) and \( \{A, B, C\} \); \( \overline{S}_C \), \( \overline{S}_{AC} \), \( \overline{S}_{ABC} \) and a query sequence \( \overline{Q} \). \( D(\overline{S}_C, \overline{Q}) = 0.64, D(\overline{S}_{AC}, \overline{Q}) = 0.2 \) and \( D(\overline{S}_{ABC}, \overline{Q}) = 0.32 \). Thus it can be noticed that \( D(\overline{S}_C, \overline{Q}) > D(\overline{S}_{AC}, \overline{Q}) \) but \( D(\overline{S}_{AC}, \overline{Q}) < D(\overline{S}_{ABC}, \overline{Q}) \). However, this paper presents an interesting property related to the upper lower bounding distance.

**Lemma 2:** The upper lower bounding distance between the (upper bound) support time sequence of the item set and a reference time sequence is monotonically nondecreasing with the size of the item set.

Fig. 2b. shows the monotonically nondecreasing property of the upper lower bounding distance.

e) **Data Scan Methods**

This paper proposes two data scan methods.

1) **Lattice-dominant scan method:** This method scans a whole transaction database from time \( t_1 \) to time \( t_n \) at each level of lattice item sets and generate the support time sequences of item sets over all.

2) **Snapshot dominant scan method:** This method repeats the scanning of transactions at a time slot, e.g., from the first time slot, with counting the supports of item sets until it finds all candidate item sets of different sizes. It then moves to the next time slot and repeats the process.

IV. ALGORITHM FOR TIME–PROFILeD ASSOCIATION MINING

1. **Generate the support time sequences of single items and find similar items.** The algorithm presented uses lattice dominant scan method to generate support time sequences. In the first scan of a whole time-stamped database, the supports of single items (\( k = 1 \)) are counted per each time slot and their support sequences \( \overline{S}_i \) are generated. If the distances between the support time sequences and a given reference sequence do not exceed a given dissimilarity threshold, the single items are added to a result set. If the upper lower-bounding distances of the support time sequences is greater than the given dissimilarity threshold, the single item is pruned from the candidate set. The others are used to generate the next size candidate sets.

2. **Generate the candidate item set.** All size \( k \) (\( k \geq 1 \)) candidate item sets are generated using size \( k-1 \) item sets whose upper lower-bounding distance satisfy the dissimilarity threshold. If any subset of size \( k-1 \) of a candidate item set does not satisfy the dissimilarity threshold the candidate is eliminated using the monotonically nondecreasing property of upper lower-bounding distance.

3. **Generate the upper and lower bound support sequence of candidate item set.** The upper and lower bound support sequence of size \( k \) candidate item set are generated using the support sequence of their size \( k-1 \) subsets.

4. **Prune candidate item sets using their lower bounding distance.** Calculate the lower bounding distance of the upper and lower bound support sequences of a candidate item set. If the lower bounding distance is greater than the similarity threshold, the candidate is eliminated from the candidate set.

5. **Scan the database and find the similar item sets.** The supports of candidate item sets are computed during the scan of the transaction data set from time slot \( t_1 \) to \( t_n \) and their support time sequences are generated. The
true similarity value between the support time sequence of an item set and the query sequence are calculated. If the value satisfies the threshold, the item set is included in the result set. The size of examined item sets is increased to \( k = k + 1 \) and the above procedures are repeated until no candidate item set remains in the previous pass.

V. EXPERIMENTAL RESULTS

The experiments were performed using the synthetic data sets. The experimentation was carried on Intel Core I3 Machine with 3GB of memory. Synthetic data sets were generated using a general transaction dataset generator used in [1]. The experiments were performed on TD100_D1_L10_I20_T100 data set. Here TD is the total number of transaction (\( \times 1,000 \)), D is the number of transactions per time slot (\( \times 1,000 \)), I is the number of distinct items, L is the average size of transactions, and T is the number of distinct time slots. A reference time sequence was generated by choosing randomly a support sequence of an item set.

The effect of similarity measure was examined with different similarity thresholds using the synthetic data set. Reference sequences were chosen near the 0.5 quantile in the data set. Fig. 3a shows the execution time required by the proposed algorithm for different threshold values. The proposed algorithm showed a little effect with the small increase in the similarity threshold in this dataset.

The effect of number of items was examined on the synthetic data set TD10_D1_L6_I*_T10 data set. The threshold value was fixed at 0.1. Fig. 3b shows the execution time required by the proposed algorithm for different number of items.

VI. REFERENCES