ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20180218021747
Original scientific paper

Mining Weighted Frequent Closed Episodes over Multiple Sequences

Guoqiong LIAO, Xiaoting YANG, Sihong XIE, Philip S. YU, Changxuan WAN

Abstract - Frequent episode discovery is introduced to mine useful and interesting temporal patterns from sequential data. The existing episode mining methods mainly focused on mining from a single long sequence consisting of events with time constraints. However, there can be multiple sequences of different importance as the persons or entities associated with each sequence can be of different importance. Aiming to mine episodes in multiple sequences of different importance, we first define a new kind of episodes, i.e., the weighted frequent closed episodes, to take sequence importance, episode distribution and occurrence frequency into account together. Secondly, to facilitate the mining of such new episodes, we present a new concept called maximal duration serial episodes to cut a whole sequence into multiple maximum episodes using duration constraints, and discuss its properties for episode shrinking processing. Finally, based on the theoretical properties, we propose a two-phase approach to efficiently mine these new episodes. In Phase I, we adopt a level-wise episode shrinking framework to discover the candidate frequent closed episodes with the same prefixes, and in Phase II, we match the candidates with different prefixes to find the frequent close episodes. Experiments on simulated and real datasets demonstrate that the proposed episode mining strategy has good mining effectiveness and efficiency.

Keywords: closed episodes; episode mining; frequent episodes; multiple sequences

1 INTRODUCTION

Discovering frequent episodes was first introduced for discovering useful and interesting temporal patterns from sequential data in [1], where an episode was defined as a partially ordered collection of events occurring together as a subsequence within a given window in a sequence. Over the years, many episode mining methods, including frequent episode mining [2-13], closed episode mining [14-16], episode rule mining [17, 18] and utility episode mining [19], have been proposed to discover temporal correlations successfully in many application domains such as alarm sequence analysis in telecommunication networks [2], user-behaviour prediction for web applications [20], stock trend analysis based on financial events [7], etc.

To the best of our knowledge, there is no existing method for episode mining over multiple sequences of different importance. There are two reasons why it attracted less interest. Firstly, episode mining is initially suggested to mine the frequent subsequences from a single long sequence [2]. Therefore, the existing research on episode mining mainly focused on the single sequence. Secondly, the problem of episode mining in multiple sequences was simply reduced to sequential pattern mining approximatively as [21]. But the method may lose information such as episode distribution, sequence importance.

Given a scenario in an e-commerce website, according to historical order records and purchasing power, users can be divided into ordinary users, VIP users and gold VIP users. In order to facilitate product recommendation on the website, on the one hand, it is expected to find the frequent patterns which occurred close in time in the sequences of multiple users, i.e. finding the frequent purchase behaviour patterns from multiple sequences. On the other hand, it is desirable that the sequences from the users of more importance have higher impact on the analysis results than from the users of less importance, i.e., paying more attention on the important users.

Now assumed we have 3 sequences S_1, S_2 and S_3 from the ordinary users, VIP users and gold VIP users respectively. There are two cases: 1) X is an episode only occurring in S_1 10 times; 2) Y occurs 3, 3 and 4 times in S_1, S_2 and S_3 respectively. For the minimum support threshold of 3, both X and Y are identified as frequent episodes. However, the result from the second case is more significant for the recommendation applications, since Y is a frequent episode in all the three sequences, and some of the sequences come from the users of more importance.

As far as we can see, there are two issues to be addressed for episode mining over multiple sequences: 1) how to evaluate the importance of the episodes in the circumstances of multiple sequences with different importance? 2) how to find the desired episodes much more efficiently? However, they have been overlooked by existing episode mining algorithms:

- The existing algorithms used the various frequency definitions such as window-based frequency [2], minimal occurrence-based frequency [6, 10], non-overlapped frequency [11, 12], but they seldom consider the characteristics such as sequence importance and episode distribution.
- As sequential pattern discovery, the existing episode mining algorithms can be roughly classified into A priori-based breadth-first algorithms [2-4, 11-13] and projection-based depth-first algorithms [5, 7, 10, 14]. Since these two kinds of algorithms are *level-wise pattern growth methods*, i.e. mining longer sequential episodes by growing from shorter episodes, they are time-consuming in multiple sequences due to level-wise iteration, and the mining time increases in terms of sequence lengths.

To facilitate discussion of the mining method in multiple sequences, we focus on mining serial closed episode and adopt the window-based frequency [2, 9]. The contributions of the paper can be concluded as follows:

- We first define a new kind of episodes, i.e., the weighted frequent closed episodes, to take sequence importance, episode distribution and occurrence frequency into account.

- To facilitate mining the new kind of episodes, we present a new concept called MDSE (**M**aximal **D**uration **S**erial **E**pisodes) to cut a whole sequence into multiple maximum episodes, and discuss its theoretical properties for episode shrinking processing.
- Based on the properties, we propose a two-phase approach to efficiently mine these new episodes. In Phase I, we adopt a level-wise episode shrinking framework to discover the candidate frequent closed episodes with the same prefixes, and in Phase II, we match the candidates with different prefixes to find the frequent close episodes.
- Experiments on simulated and real datasets verify the efficiency and effectiveness of the proposed episode mining method.

The rest of the paper is organized as follows. The related works are introduced in Section 2. In Section 3, we introduce the basic concepts and problem definition. We put forward the concept of maximal duration serial episodes and discuss their theoretical properties in Section 4. In Section 5, we propose the details of the two-phase mining approach. Performance evaluation is conducted in Section 6. Finally, we conclude the paper.

2 RELATED WORKS

Episodes mining was first introduced in [1]. The related works can be classified into the following aspects:

Frequent episode mining. In general, the existing frequent episode mining algorithms can be classified into Apriori-based breadth-first algorithms [2-4, 11-13] and projection-based depth-first algorithms [5, 7, 10]. [2] presented two kinds of Apriori-based frequent episode mining methods, WINEPI and MINEPI. [3-4] proposed two kinds of probabilistic frequent serial episode mining methods from uncertain sequences. [11] proposed a new notion for episode frequency based on the non-overlapped occurrences. [12] presented two fast non-overlapped occurrences counting algorithms for the serial episodes and parallel episodes. In terms of episode mining in complex sequences, [13] proposed two frequent episodes mining algorithms, MINEPI+ and EMMA. [5] presented an algorithm that can find all frequent partite episodes satisfying a partwise constraint in an input sequence. [7] used the event tree structure to represent the sets of event types with paths and nodes, and to mine the frequent episodes by a recursive procedure. [10] proposed two episode mining approach based on a prefix-growth-Episode Prefix Tree (EPT) and Position Pairs Set (PPS). Both kinds of methods are time-consuming in multiple sequences due to level-wise iteration, and the mining time increase in terms of sequence lengths.

Closed episodes mining. [14] proposed *Clo-episode* to mine the closed serial episodes following a breadth-first search order and integrating the pruning techniques using a prefix tree. To solve the problem that a non-closed frequent episode can have more than one closure in general episodes, [15] introduced the concept of strict episodes and defined instance-closed episodes. [16] extended the definition of an episode in order to be able to represent the cases where events often occur simultaneously. It proposed an efficient depth-first search method algorithm using a directed acyclic graph.

Utility episode mining. [19] incorporated the concept of utility mining into episode mining and proposed a novel framework for mining high utility episodes in complex event sequences, which considers the external utility and internal utility of events to measure the utility of episodes. To some extent, our work is also a kind of a utility episodes mining. But in that model, each event has a weight associated with it. Here, we associate a weight with each sequence.

Episode mining over event streams. By introducing the concept of last episode occurrence within a time window, [22] proposed a method to mine the online frequent episodes by detecting new minimal episode occurrences. [23] proposed the episode matrix and frequent episode tree based mining method over event streams for episode mining. [24] proposes an efficient algorithm for the discovery of frequent and periodic episodes in streams.

Episode mining in multiple sequences. To the best of our knowledge, there is no existing work for episodes mining over multiple sequences. Although [21] presented a frequent episode mining method from multiple electronic medical records, the method reduces the episodes mining into sequential pattern mining. This method may lose information such as episode distribution, sequence importance, etc.

3 BASIC CONCEPTS AND PROBLEM DEFINITON

This section gives the basic concepts and the problem to be solved in this work.

Definition 1 (**Event sequence**). Let E be a set of event types. An event is defined by the pair (e, t) where $e \in E$ and t is the time at which the event occurs. An ordered event sequence is denoted as S=<(e_1, t_1), (e_2, t_2), …, (e_n, t_n)>, such that $\forall i \in \{1, \ldots, n\}, e_i \in E$, and $t_i < t_j$, $1 \le i < j \le n$.

Definition 2 (Super sequence and subsequence). Given two event sequences S={(e_1, t_1), (e_2, t_2), …, (e_n, t_n)}and S'={(e_1', t_1'), (e_2', t_2'), …, (e_m', t_m')}, we say that S is a super sequence of S' (i.e., S'is a sub-sequence of S), if and only if each e_j' (e_j'$\in S$') can be mapped by e_{ij} (e_{ij} $\in S$ and e_{ij} corresponds to the j^{th} event in S'), i.e., e_j'= e_{ij} ,and all e_{ij} preserve the temporal order, i.e., $1 \le i_1 < i_2 < \ldots < i_m \le n$.

Definition 3 (Serial episode). A serial episode is a non empty partial ordered set of events P=<e_1, e_2, …,e_k>, where each e_i is a nonempty event set and e_i occurs before e_j for all $1 \le i < j \le k$.

Definition 4 (Occurrence). Given a sequence S, a serial episode P=<e_1, e_2, …,e_k>and a window size z, it is said that P occurs in a sliding window Z_i=(e_{i1}, e_{i2}, …, e_{iz-1}) $\in S$, if and only if, e_1= e_{i1} and <e_2, …, e_k>is a subsequence of (e_{i2}, …, e_{iz-1}). Z_i is also called a match of P in S.

Definition 5 (Episode support). The number of sliding windows in sequence S that match episode P is called the support of P in S, denoted as $sup(P)$. In particular, $sup_i(P)$ represents the support of P in the i^{th}

sequence of multiple sequences.

Definition 6 (Frequent episode). An episode *P* is a frequent episode in *S*, if and only if the support of *P* is at least the user-specified minimum support, denoted as *minsup*, that is, $sup(P) \geq minsup$.

Definition 7 (Subepisode and super episode). Given two episodes $P = < e_{i1}, e_{i2}, \ldots, e_{in} >$ and $P' = < e_{j1}, e_{j2}, \ldots, e_{jm} >$, $m \leq n$, if *P*' is a subepisode of *P* (i.e., *P* is a super episode of *P*'), if and only if there are *m* integers, $1 \leq i_1 \leq i_2 \ldots \leq i_m \leq n$, for all *k* ($1 \leq k \leq m$), $e_{jk} = e_{ik}$, denoted as $P' \subseteq P$.

Definition 8 (Closed episodes). *P* is a closed episode in *S*, if and only if there is no any episode *P*' in *S*, $P \subseteq P'$ and $sup(P) = sup(P')$.

Definition 9 (Frequent Closed episodes). If *P* is both a closed and a frequent episode, *P* is called a frequent closed episode.

Definition 10 (Weighted supports). Let $D = \{S_i\}$ ($i = 1, 2, \ldots, N$) be a collection of $N(N > 1)$ sequences, $W = \{w_i\}(i = 1, 2, \ldots, N)$ be the set of the weights (i.e., importance) of all sequences in *D*, and *P* be a frequent closed episode at least in one sequence of *D*. A weighted support of *P*, denoted as *sup_wa*(*P*), is calculated according to Eq. (1):

$$sup\_wa(P) = \frac{N_I}{N} \times \sum_{i \in I} (\omega_i \times sup_i(p)), \tag{1}$$

where, *I* is the set of the sequences in *D* where *P* is a frequent closed episode, and N_I is the number of sequences in *I*.

Definition 11 (Weighted frequent closed episodes). An episode *P* is a weighted frequent closed episode in *D*, if and only if the weighted support of *P* is at least the user-specified minimum weighted supports, denoted as *minsup_wa*, i.e., $sup\_wa(P) \geq minsup\_wa$.

For example, there are 3 sequences (N=3), $minsup$=3, $minsup\_wa$=5, and the weights of S_1, S_2 and S_3 are 1, 2, and 3respectively.

Case 1): *P*=<*CDE*> is an episode, and only occurs in sequence S_1 10 times, namely $sup_1(CDE)$=10. By Definition 10, the weighted support of *CDE* is calculated as Eq. (2):

$$sup\_wa(CDE) = \frac{1}{3} \times (1 \times 10) = 3.3\,. \tag{2}$$

Since $sup\_wa(CDE) < minsup\_wa$, *CDE* is not a weighted frequent closed episode.

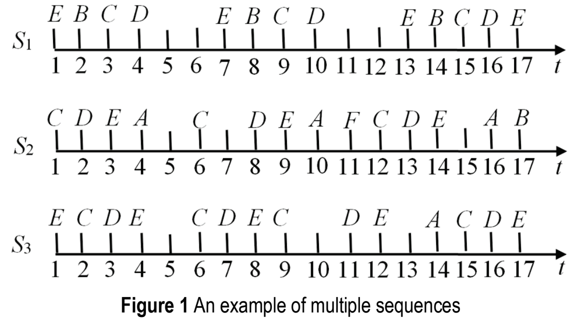

E B C D E B C D E B C D E
S1
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 t
C D E A C D E A F C D E A B
S2
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 t
E C D E C D E C D E A C D E
S3
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 t

Figure 1 An example of multiple sequences

Case 2): *CDE* occurs respectively in S_1, S_2 and S_3 3, 3 and 4 times (see Fig. 1). As we know, *CDE* is a frequent closed episode in all three sequences, so its weighted support can be calculated as Eq. (3):

$$sup\_wa(CDE) = \frac{3}{3} \times (1 \times 3 + 2 \times 3 + 3 \times 4) = 21\,. \tag{3}$$

Therefore, *CDE* is a weighted frequent closed episode in Case 2 due to $sup\_wa(CDE) > minsup\_wa$, which is in accordance with our mining expectation.

Definition 12 (Problem definition). Let *D* be a collection of multiple sequences, and given a user-specified maximal window width *maxwin*, *minsup* and *minsup_wa*, the task of this work is to discover all weighted frequent closed episodes in *D*, where each $P = <e_1, e_2, \ldots, e_k>$must satisfy the following conditions:

1) $t_k - t_1 \leq maxwin$;

2) There is at least a sequence $S_i \in D$, $sup_i(P) \geq minsup$, and there is no any $P' \in S_i$, $P \subseteq P'$and$sup_i(P') = sup_i(P)$;

3) $sup\_wa(P) \geq minsup\_wa$.

4 MAXIMAL DURATION SERIAL EPISODES

4.1 Maximal Duration Serial Episodes

According to Definition 4, if the difference of the occurrence time of a pair of adjacent events in a sequence is more than the window size *z*, they cannot be the elements of any episode occurrence. We make use of this feature to extract the frequent adjacent events from the sequences as candidate episodes.

Definition 13 (2-neighboring episode, 2NE). A pair of adjacent events $<e_j, e_{j+1}> \in S$ is a 2-neighboring episode, if and only if $t_{j+1} - t_j \leq maxwin$.

Definition 14 (Frequent 2-neighboring episode, F-2NE). A 2-neighboring episode$<e_j, e_{j+1}>$ is a frequent 2-neighboring episode, if and only if $sup(e_j, e_{j+1}) \geq minsup$.

Definition 15 (Index point). If $<e_j, e_{j+1}>$is an F-2NE in *S*, $<sid, t_j, pos_j>$ is said to be an index point of $<e_j, e_{j+1}>$in *S*, where *sid* is the identification of *S*, t_j is the occurring time of e_j, and pos_j is the position of e_j in *S*.

To avoid scanning sequences again after getting the F-2NEs, we record all index points of each F-2NE into an inverted list, called F2NE_Index. It is obvious that the number of the index points of an F-2NE in the sequence is equal to its support.

Supposed minsup=3 and maxwin=4, <*EB*>, <*BC*>, <*CD*> and <*DE*> are the F-2NEs in S_1. Their supports and index points are listed in Tab. 1. For example, the index point <1, 7, 5> of *EB* represents *EB* occurs in S_1 at time 7, and the position of first event E in S_1 is 5.

Table 1 The supports and index points of the F-2NEs in S_1

F-2NE	Support	Index point
<EB>	3	<1, 1, 1>, <1, 7, 5>, <1, 13, 9>
<BC>	3	<1, 2, 2>, <1, 6, 6>, <1, 13, 10>
<CD>	3	<1, 3, 3>, <1, 9, 7>, <1, 15, 11>
<DE>	3	<1, 4, 4>, <1, 10, 8>, <1, 16, 12>

The purpose of extracting the F-2NEs is to kick out the non-frequent 2NEs from the sequences. The generation of F-2NEs consists of two steps:

1) For each sequence, it scans and counts the number of each adjacent event which time interval is not greater than *maxwin* in the sequence, to obtain 2-neighboring episode set *2NEp*;

2) It gets all F-2NEs from *2NEp*. If the support of the 2NE is not less than *minsup,* to obtain the candidate episode set *CanEp*.

Since there is a time constraint for each episode, we extend the F-2NEs into the maximum episodes by *maxwin* as far as possible.

Definition 16 (F-2NE prefix). Φ=<e_1, e_2> is called the F-2NE prefix of P=<e_1, e_2, …, e_k>(k≥2).

For the sake of simplicity, we use "prefix" to represent the F-2NE prefix in the paper.

Definition 17 (Maximal duration serial episode, MDSE**).** P=<e_1, e_2, …,e_k>(k≥2) is a maximal duration serial episode with prefix Φ=<e_1, e_2>in S, if and only if t_k-t_1≤*maxwin*, and there is no super episode P'=<e_1, e_2, …, e_k, e_{k+1}> with the same prefix of P in S, t_{k+1}−t_1≤*maxwin*. Also, we call P has length k.

For *minsup* of 3 and *maxwin* of 4, Fig. 2 shows the maximal duration serial episodes extended from different prefixes in S_1 (not including the F-2NEs).

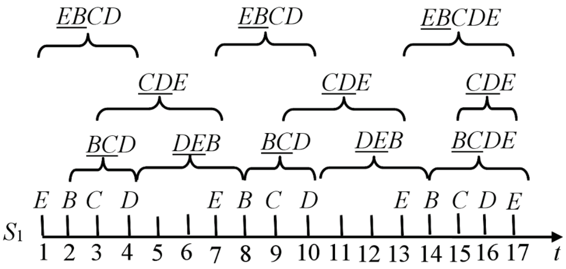


Figure 2 The generation of the maximal duration serial episodes in S_1

The prefixes and their corresponding MDSEs are listed in Tab. 2, where the underlined events mean the prefixes, and the numbers in the parentheses are the supports. For example, $\underline{EB}CDE$(1) represents that the prefix of $\underline{EB}CDE$ is *EB* and its support is 1.

Table 2 The MDSEs with different prefixes in sequence S_1

F-2NE prefix	MDSE
EB(3)	$\underline{EB}CDE$(1), $\underline{EB}CD$(2)
BC(3)	$\underline{BC}DE$(2), $\underline{BC}D$(1)
CD(3)	$\underline{CD}EB$(1), $\underline{CD}E$(2)
DE(3)	$\underline{DE}BC$(1), $\underline{DE}B$(1)

A maximal duration serial episode is extended from an F-2NE <e_j, e_{j+1}>, which should guarantee the occurring time of each event must be in the time interval [t_i, t_i+*maxwin*], so t_{max}=t_i+*maxwin* is the maximal occurring time of the event in an episode. The construction of MDSEs is shown in Algorithm 1.

Algorithm 1. ***MDSE_Construction*( )**

```
Input:   X=<$e_j$, $e_{j+1}$>;maxwin;
Output:  X.MDSE; // The Set of MDSE of X
Begin:
X.MDSE =∅;
GetIndexpoint(X); //get the index points of X
for each index point i of X do
   tmp_mdse=NULL;
   tmp_mdse=tmp_mdse+$e_j$;
   tmp_mdse= tmp_mdse+$e_{j+1}$;
   $t_{start}$=X[i].$t_j$; //get the occurring time
      p=X[i].$pos_j$; //get the position of the $i^{th}$ index point
   $t_{max}$= $t_{start}$+maxwin;
      p=p+2; //start from the second event after p
   while $t_p$≤$t_{max}$
   tmp_mdse=tmp_mdse+$e_p$; //add $e_p$ into the episode
   p=p+1;
   end while
   X.MDSE=X.MDSE∪tmp_mdse
end for
return X.MDSE
End
```

4.2 Candidate frequent closed episodes

Based on the maximal duration serial episodes, we define the candidate frequent closed episodes.

Definition 18 (Episodes with the same prefix). P=<e_{i1}, e_{i2}, …, e_{in}> and P'=<e_{j1}, e_{j2}, …, e_{jm}> are two different episodes with the same prefix in a sequence, if and only if $e_{i1}=e_{j1}$, *and* $e_{i2}=e_{j2}$.

Definition 19 (Super episode and subepisode with the same prefix). P'=<e_{j1}, e_{j2}, …, e_{jm}>is a subepisode of P=<e_{i1}, e_{i2}, …, e_{in}> with the same prefix (i.e., P is the super episode of P with the same prefix), if and only if e_{j1}=e_{i1}, e_{j2}=e_{i2}, and there is m integers, 1≤i_3≤i_4…≤i_m≤n, for all k(3≤k≤m), e_{jk}=e_{ik}, denoted as $P'\underline{\subseteq}\underline{\subseteq} P$.

Let P=<e_1, e_2, …, e_k> be a maximal duration serial episode based on prefix Φ=<e_1, e_2> in a sequence, we have the following definition and properties.

Definition 20 (Candidate frequent closed episode). P is a candidate frequent closed episode (CFCE) based on prefix Φ, if and only if *sup*(P)≥*minsup*, and there is no P' with prefix Φ, $P\underline{\subseteq}\underline{\subseteq}P'$ and *sup*(P)=*sup*(P').

Property 1. If s*up*(P)≥*minsup*, the maximal duration serial episode P is a candidate frequent closed episodes based on its prefix.

Proof: 1) If there is another maximal duration serial episode P' with prefix Φ and $P\underline{\subseteq}\underline{\subseteq}P'$, the total support of P should be equal to the sum of *sup*(P) and *sup*(P'). Thus, P cannot be closed by P'.

2) If there is no any maximal duration serial episode P' with prefix Φ and $P\underline{\subseteq}\underline{\subseteq}P'$, there is not any super episode with the same prefix can close P.

By Definition 20, if *sup*(P)≥*minsup*, P is a candidate frequent closed episode based on Φ.

Property 1 shows, when doing closure relation judgment, we only need to determine the closure relations between the maximal duration serial episodes and their subepisodes.

Property 2. If *sup*(P)=*sup*(Φ), all subepisodes with the same prefix of P are closed by P.

Proof: As we know, in a sequence, if *sup*(P)=*sup*(Φ), Φ can't be the prefix of any other maximal duration serial episode in the sequence.

Therefore, the supports of all P'*s* subepisodes with prefix Φ are the same as the P's support, so that they are closed by P.

For example, in Fig. 1, <$\underline{CD}EA$> is a maximal duration serial episode in S_2, and its support is 3, which is the same as the support of its prefix *CD*. By Property 2, all subepisodes with prefix *CD* of *CDEA*, i.e., *CDE*, *CDA*, and *CD*, are closed by *CDEA*.

Therefore, if a maximal duration serial episode has the same support as its prefix, it is not necessary to further judge the closure relationships between the episode and its subepisodes. This is an important termination condition of level-wise episode shrinking.

Thus, we can redefine the frequent closed episodes based on Definition 10.

Definition 21 (Frequent closed episodes). *P* is a frequent closed episode, if and only if there is not any candidate frequent closed episode *P*' with different prefix differing from *P* in the sequence, $P \subseteq P'$and *sup*(*P*)=*sup*(*P*').

5 MINING WEIGHTED FREQUENTCLOSED EPISODES

In this section, we discuss the details of the two-phase closure strategy: discover the candidate frequent closed episodes and frequent closed episodes

5.1 Phase I Closure: Generation of Candidate Frequent Closed Episodes

Differing from the level-wise pattern growth methods, we use the idea of "from longer to shorter, level-wise shrinking" to find the candidate frequent closed episodes.

The shrinking procedure works like this. Assumed *X* is a maximal duration serial episode with length *L*, and *Φ* is its prefix. We first get all its length-(*L*-1) subepisodes with prefix *Φ*. For each one, it will be matched with the maximal duration serial episodes of the same prefix except *X*. If its support is not less than *minsup* and not equal to *sup*(*X*), it is determined as a candidate frequent closed episode; otherwise, as a non-candidate frequent closed episodes. Then, it is shrunk into length-(*L*-2) subepisodes, and so on, until one of the following two termination conditions is held:

TC1: Property 2 is held, that is, the support of the episode or the subepisode is equal to that of its F-2NE prefix;

TC2: The episode or the subepisode is the F-2NE prefix itself.

We use the prefix *EB* in Tab. 2 as an example to explain the shrinking procedure. *EBCDE* is a maximal duration serial episode and its support is 1. For sup(*EBCDE*) is less than *minsup*, it is certainly a non-candidate frequent closed episode. Since *sup*(*EBCDE*) is not equal to *sup*(*EB*) and its length is 5, it needs to be shrunk further. Starting from its 4-subepisodes, each one will be matched with the other maximal duration serial episodes with prefix *EB*. For instance, *EBCD* is a 4-subepisodes of *EBCDE*. After matching, sup(*EBCD*)=3. For *sup*(*EBCD*)≠*sup*(*EBCDE*) and *sup*(*EBCD*)=*minsup*, *EBCD* is marked as a candidate frequent closed episode. As a result of *sup*(*EBCD*)=*sup*(*EB*), satisfying Property 2, the shrinking of *EBCD* terminates. In the same way, we can find the 4-subepisodes *EBCE* and *EBDE* are non-candidate frequent closed episodes. They will be further shrunk till one termination condition is satisfied.

There is a special case in the shrinking procedure. *EBC* is a 3-subepisode of the 4-subepisode *EBCE*. After matching, its support is 3. Since its support is not equal to that of *EBCE* and equal to *minsup*, it should be a candidate frequent closed episode by the closure rule. It can be found, however, *EBC* is also a subepisode of the mined candidate frequent closed episode *EBCD*, and their supports are the same, so it should be closed by *EBCD*. The same cases include the 3-subepisode *EBD* of *EBDE*, and the F-2NE prefix *EB*.

We find that these episodes have the same characteristic, that is, their supports are equal to that of their F-2NE prefix. Therefore, in order to guarantee mining accuracy, besides matching with their parent episodes, they should be matched with the mined candidate frequent closed episodes which have the same support of the F-2NE prefix.

Rule 1. If a subepisode has the same support with its F-2NE prefix, and its support is different with its parent episode, it should check whether there is a super episode which has the same support with the F-2NE prefix mined. If so, the subepisode is regarded as a non-candidate frequent closed episode.

In order to guarantee that Rule 1 can work, we order the MDSEs with the same prefix according to episode lengths by descending order, and discover the candidate frequent closed episode starting from the longest one.

Fig. 3 is the shrinking procedure of *EBCDE* integrating Rule 1. In the figure, "Y" represents the episode is a candidate frequent closed episode, "N" represents it is a non-candidate frequent episode, and "-||" is the termination symbol.

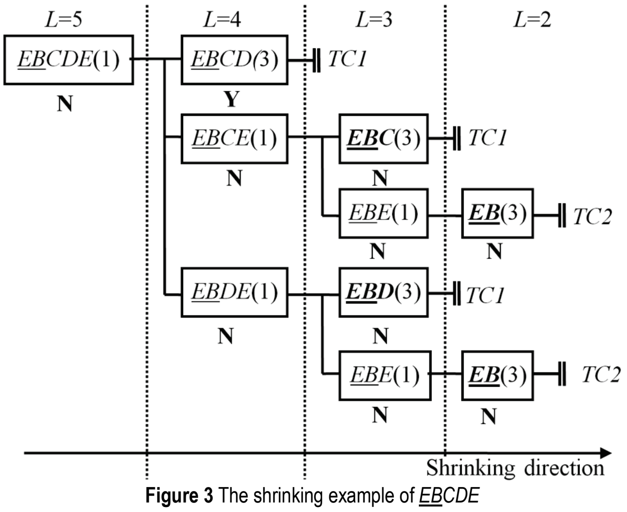


Figure 3 The shrinking example of *EBCDE*

But, we can find that there are some repetitive matches in the shrinking processing. For example:

1) The 3-subepisode *EBE* of 4-subepisode *EBDE,* which has been matched in 4-subepisode *EBCE.*
2) The F-2NE prefix *EB*, which support has been counted when generating the F-2NE.

To avoid repetitive matching, for case 1), besides the candidate frequent closed episodes, it also needs to keep the non-candidate frequent closed episodes. Before matching, it will check whether it has been matched. For case 2), we have the following rule:

Rule 2. If an episode or subepisode is the F-2NE prefix, its support is obtained directly from the index table of the sequence, instead of matching with the MDSEs.

5.2 Phase II closure: Generation of Final Frequent Closed Episodes

It is relatively easier to mine the final frequent closed episodes.

The closure relation only occurs between the episodes with the same support. When we have obtained the candidate closed episodes with the same F-2NE prefixes in Phase I, it only needs to judge the closure relationships among the episodes with different prefixes having the same supports.

After getting the frequent closed episodes of each sequence, we can calculate the weighted supports of each episode over multiple sequences according to Definition 10. If the support is not less than *minsup_wa*, the episode is our expected finally.

6 PERFORMANCE EVALUATION

We use MDSE-I and MDSE-II to represent the methods that MDSE is using one-phase closure strategy and two-phase closure strategy, respectively. Differing from MDSE-II, in MDSE-I, each episode matches with all maximal duration serial episodes in the sequence, instead of generating the candidate frequent closed episodes. In default, the MDSE method is the MDSE-II method.

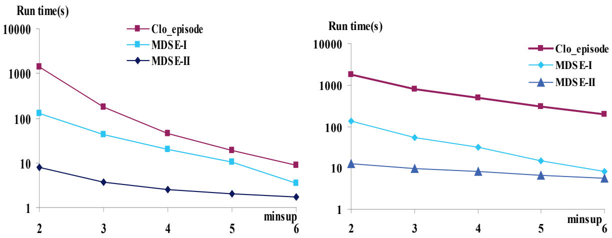
All experiments are performed both on the simulated and real datasets: 1) the simulated dataset T10I4D100K is generated from the IBM Almaden Quest (http://fimi.ua.ac.be/data/), which contains 100000 sequences; 2) The real dataset is the retail records from an anonymous retail store in Belgium (http://fimi.ua.ac.be/data/), which contains 88162 sequences. The test includes two parts:

1) Mining the frequent closed episodes in the single sequence. Since the first step of MDSE is to use the two-phase closed strategy to mine the frequent closed serial episodes in the single sequence, we choose the closed serial episode mining algorithm Clo–episode [14] for comparison;

2) Mining the weighted frequent closed episodes over multiple sequences, where we analyse the influence of the combination of different parameters on the number of the mined episodes.

6.1 Mining Closed Episodes in Single Sequence

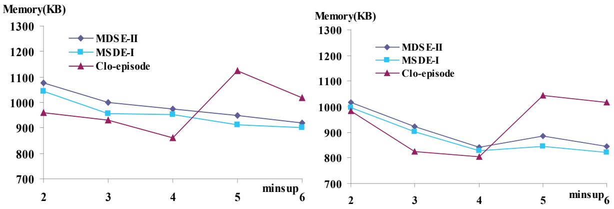
Fig. 4 is the running time comparison of MDSE-I, MSED-II and Clo-episode as the time window threshold is 10 under different support thresholds. It can be found, the running time of the two MDSE-based methods is obviously less than that of Clo-episode. The reason is MDSE can quickly generate the maximal duration serial episodes using index and the shrinking procedure can terminate quickly. We also can see the running time of MDSE-II is less than that of MDSE-I on the two datasets. This is because the two-phase closure strategy only matches the episodes with the same prefix.



(a) Simulated dataset (b) Real dataset

Figure 4 The running time comparison under different support thresholds

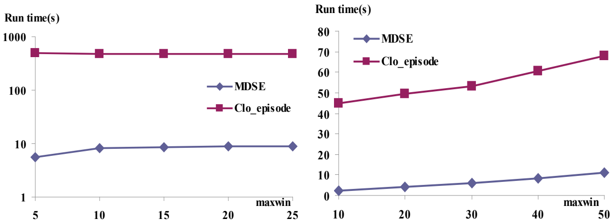
Fig. 5 is memory overhead comparison of the two MDSE-based methods and Clo-episode in the maximal time window of 10 under different support thresholds. As shown in the figures, the memory overhead of the two MDSE methods is slightly higher than that of Clo-episode when support threshold is less than 4. This is because when the support threshold is smaller, it needs more memory space to store the F-2NEs and index information. But when the support is up to 5, the memory required by Clo-episode becomes larger than the MDSE methods, since the number of subepisodes to be extended in Clo-episode increases suddenly. We can also see the memory overhead of MDSE-II is slightly higher than that of MDSE-I, since MDSE-II needs extra storage space to save the candidate frequent closed episodes.



(a) Simulated dataset (b) Real dataset

Figure 5 The memory overhead comparison under different support thresholds

Fig. 6 is the running time comparison of the two methods when the support threshold is 4 under different time window size. Overall, the running time of MDSE is less than that of Clo-episode. We can see from Fig. 7 (a), the running time of the two methods is little affected by the time window size in the simulated dataset. The reason is the simulated data is relatively sparse. But in Fig. 7 (b), as the growth of the window size, the running time of the two methods in a real dataset increases correspondingly.

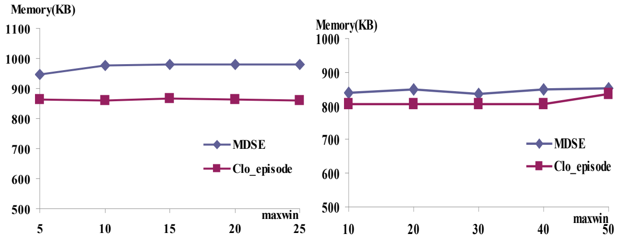


(a) Simulated dataset (b) Real dataset

Figure 6 The running time comparison under different time window sizes

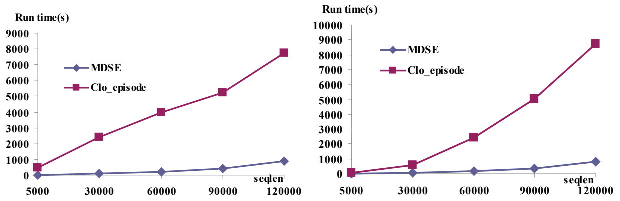
Fig. 7 is memory overhead change trends when the support threshold is 4 under different time window sizes. It can be found the space change of both methods is relatively stable, and the space required by MDSE is slightly higher than that of Clo-episode.

Fig. 8 is the running time comparison of the two methods when *minsup*= 4 and *maxwin*=10 with different sequence lengths. We can see from the figures, the growth ratio of Clo-episode is significantly higher than that of MDSE.



(a) Simulated dataset (b) Real dataset

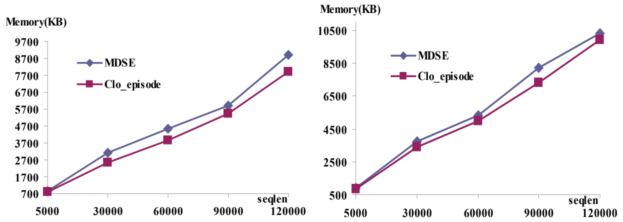
Figure 7 The memory overhead comparison under different time window thresholds



(a) Simulated dataset (b) Real dataset

Figure 8 The running time comparison under different sequence lengths

Fig. 9 is memory size comparison to mine the sequences with different lengths in *minsup*=4 and *maxwin*=10. We can see from the figures, the memory sizes of the two methods increase with the increase of the sequence length, and the memory space of MSDE is slightly higher than that of Clo-episode. This is because when the sequence length increases, the space storing the maximal duration serial episodes for MDSE also increases accordingly.



(a) Simulated dataset (b) Real dataset

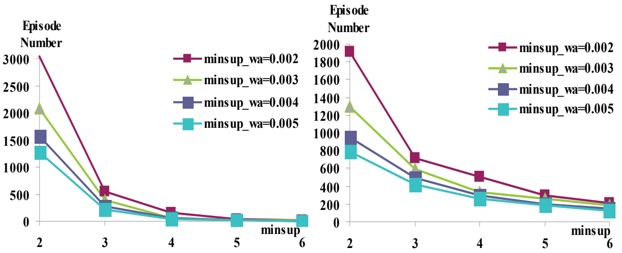
Figure 9 The memory overhead comparison under different sequence lengths

6.2 Mining Weighted Episodes in Multiple Sequences

For the simulated dataset, we combine each 100 short sequences into a long sequence, and give a timestamp to each event in the sequence in order to get 1000 long sequences. In the same way, we get 882 long sequences for the real dataset. Also, we use the sequence number as a seed, and generate a weight randomly for each sequence in the two datasets according to Eq. (4):

$$\omega_t(S_i) = (srand(i)*(4000)+3000)/(10000),\ S_i \in D \tag{4}$$

Fig. 10 is the change trend of the episode numbers in a fixed window size of 15 under different support thresholds and weighted support threshold. It can be found the episode number significantly decreased with the increase of *minsup_wa*. The reason is that with the increase of the weighted support threshold, the episode satisfying the constraint conditions in multiple sequences reduced significantly.

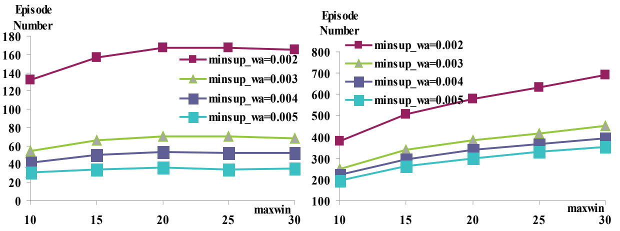


(a) Simulated dataset (b) Real dataset

Figure 10 The relationship between the episode number and the weighted support thresholds under different support thresholds

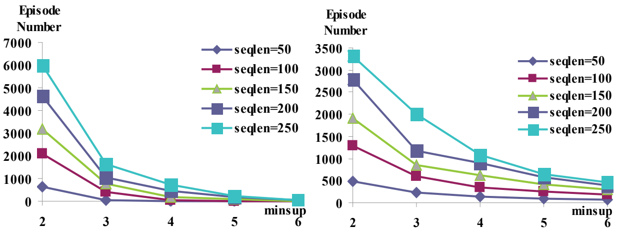
Fig. 11 is the comparison of the episode number under different weighted support thresholds in a fixed support threshold of 4 and different time window sizes. We can see the episode number decreases in general with the increase of the weighted support threshold.

Fig. 12 is the change trend of the episode number with the increase of the sequence length in a fixed window threshold of 15 and different support threshold. We can find, under the same support, the longer the sequence length is, the more the number of mined episodes is. But with the increase of the support threshold, the number of the mined episodes reduces, and the reduction ratio becomes small with the increase of the support threshold.



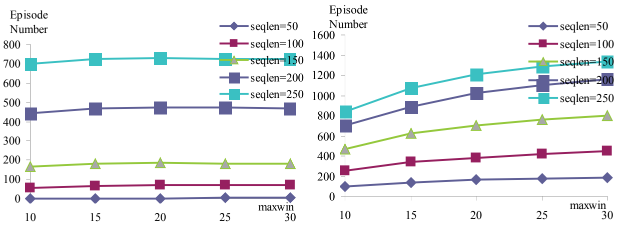
(a) Simulated dataset (b) Real dataset

Figure11 The relationship between the episode number and the weighted support thresholds under different time window sizes



(a) Simulated dataset (b) Real dataset

Figure 12 The relationship between the episode number and the sequence length under different support thresholds



(a) Simulated dataset (b) Real dataset

Figure 13 The relationship between the episode number and the sequence length under different time window sizes

Fig. 13 is the change trend of the episode number with the increase of the sequence length in a fixed support threshold of 4 and different window sizes. We can see, under the same *maxwin*, the longer the sequence length is, the more the number of the episodes is.

7 CONCLUSIONS

Episode mining is an important branch of sequential pattern mining. Focusing on frequent closed episode mining over multiple sequences, we put forward a novel episode mining strategy for the new kind of episodes – the weighted average frequent closed episodes. The main innovation of the paper includes two parts: the one is we propose a new kind of episodes and a new way to evaluate the importance of episodes in the background of multiple sequences; the other one is, we propose a novel level-wise episode shrinking framework to mine the frequent closed episodes based on the maximal duration serial episodes. Experiments verified that the proposed episode mining strategy has good mining effectiveness and efficiency.

How to integrate the level-wise episode shrinking framework with the closed parallel episode mining and complex episode mining is one of our future works.

Acknowledgement

This work is supported partly by NSF of China (No. 61772245, 61262009), NSF of Jiangxi Province, China (20151BBG70046), Science Foundation of Jiangxi Provincial Department of Education, China (GJJ160419), Building Project of Superior Innovation Team of Science and Technology of Jiangxi, China (No. 20113BCB24008).

8 REFERENCES

[1] Mannila, H., Toivonen, H., & Verkamo, A. I. (1995). Discovering Frequent Episodes in Sequences (Extended Abstract). *Proc. of 1st International Conference on Knowledge Discovery and Data Mining*, 210-215.

[2] Mannila, H., Toivonen, H., & Verkamo, A. I. (1997). Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery, 1*(3), 259-289. https://doi.org/10.1023/A:1009748302351

[3] Wan, L., Chen, L., & Zhang, C. (2013). Mining Frequent Serial Episodes over Uncertain Sequence Data. *Proc. of the 16th International Conference on Extending Database Technology*, 215-226. https://doi.org/10.1145/2452376.2452403

[4] Wan, L., Chen, L., & Zhang, C. (2013). Mining Dependent Frequent Serial Episodes from Uncertain Sequence Data. *Proceedings of the 13th IEEE International Conference on Data Mining*, 1211-1216. https://doi.org/10.1109/ICDM.2013.35

[5] Katoh, T., Tago, S., Asai, T., Morikawa, H., Shigezumi, J., & Inakoshi, H. (2014). *Mining Frequent Partite Episodes with Partwise Constraints. New Frontiers in Mining Complex Patterns.* Lecture Notes in Computer Science, 117-131. https://doi.org/10.1007/978-3-319-08407-7_8

[6] Mannila, H. & Toivonen, H. (1996). Discovering generalized episodes using minimal occurrences. *Proc. of the 2nd International Conference on Knowledge Discovery and Data Mining*, 146-151.

[7] Ng, A. & Fu, A. W. C. (2003). Mining frequent episodes for relating financial events and stock trends. *The 7th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining*, 27-39. https://doi.org/10.1007/3-540-36175-8_4

[8] Casas-Garriga, G. (2003). Discovering unbounded episodes in sequential data. *Proc. of European conference principles and practice of knowledge discovery in databases*, 83-94. https://doi.org/10.1007/978-3-540-39804-2_10

[9] Iwanuma, K., Takano, Y., Nabeshima, H. (2004). On anti-monotone frequency measures for extracting sequential patterns from a single very-long sequence. *Proc. of IEEE Conference Cybernetics and Intelligent Systems*, 213-217.

[10] Ma, X., Pang, H., & Tan, L. (2004). Finding constrained frequent episodes using minimal occurrences. *Proc. of 4th IEEE International Conference on Data Mining*, 471-474.

[11] Laxman, S., Sastry, P. S., Unnikrishnan, K. P. (2005). Discovering frequent episodes and learning hidden Markov models: A formal connection. *IEEE Transaction on Knowledge and Data Engineering, 17*(11), 1505-1517. https://doi.org/10.1109/TKDE.2005.181

[12] Laxman, S., Sastry, P. S., & Unnikrishnan, K. P. (2007). A fast algorithm for finding frequent episodes in event streams. *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 410-419. https://doi.org/10.1145/1281192.1281238

Huang, K. & Chang, C. (2008). Efficient mining of frequent episodes from complex sequences. *Information Systems, 33*(1), 96-114. https://doi.org/10.1016/j.is.2007.07.003

[13] Zhou, W., Liu, H., & Cheng, H. (2010). Mining closed episodes from event sequences efficiently. *Proc. of 7th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining*, 310-318. https://doi.org/10.1007/978-3-642-13657-3_34

Tatti, N. & Cule, B. (2010). Mining closed strict episodes. *Proc. of the 10th IEEE International Conference on Data Mining*, 501-510. https://doi.org/10.1109/ICDM.2010.89

[14] Tatti, N. & Cule, B. (2011). Mining closed episodes with simultaneous events. *Proc. of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1172-1180. https://doi.org/10.1145/2020408.2020589

[15] Meger, N., Leschi, C., Lucas, N., et al. (2004). Mining episode rules in STULONG dataset. *Proc. of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, 1-12.

[16] Tatti, N. & Vreeken, J. (2012). The long and the short of it: summarising event sequences with serial episodes. *Proc. of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 462-470. https://doi.org/10.1145/2339530.2339606

[17] Wu, C., Lin, Y., Yu, P. S., & Tseng, V. S. (2013). Mining high utility episodes in complex event sequences. *Proc. of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 536-544. https://doi.org/10.1145/2487575.2487654

[18] Laxman, S., Tankasali V., & White, V. R. (2008). Stream prediction using a generative model based on frequent episodes in event sequences. *Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 453-461. https://doi.org/10.1145/1401890.1401947

[19] Patnaik, D., Butler, P., Ramakrishnan, N., et al. (2011). Experiences with mining temporal event sequences from electronic medical records: initial successes and some challenges. *Proc. of the 17th ACM International Conference on Knowledge Discovery and Data Mining*, 360-368. https://doi.org/10.1145/2020408.2020468

[20] Ao, X., Luo, P., Li, C., et al. (2015). Online Frequent Episode Mining. *Proc. of 31st IEEE Conference on Data Engineering,* 891-902. https://doi.org/10.1109/ICDE.2015.7113342

[21] Lin, S., Qiao, J., & Wang, Y. (2014). Frequent episode mining within the latest time windows over event streams. *Applied intelligence, 40*(1), 13-28. https://doi.org/10.1007/s10489-013-0442-8

[22] Soulas, J. & Lenca, P. (2015). Periodic episode discovery over event streams. *Proc. of Portuguese Conference on Artificial Intelligence*. Springer, 547-559. https://doi.org/10.1007/978-3-319-23485-4_54

Contact information:

Guoqiong LIAO
(Corresponding author)
School of Information Technology,
Jiangxi University of Finance and Economics,
Nanchang 330013, China
E-mail: liaoguoqiong@163.com

Xiaoting YANG
School of Information Technology,
Jiangxi University of Finance and Economics,
Nanchang 330013, China
E-mail: 523338968@qq.com

Sihong XIE
Computer Science and Engineering Department,
Lehigh University,
Bethlehem, PA 18015, USA
E-mail: xiesihong1@gmail.com

Philip S. YU
Department of Computer Science,
University of Illinois at Chicago,
Chicago, IL, 60607, USA
E-mail: psyu@uic.edu

Changxuan WAN
School of Information Technology,
Jiangxi University of Finance and Economics,
Nanchang 330013, China
E-mail: wanchangxuan@263.net