Discovery of Frequent Episodes in Event Sequences

Alfiya Akhmetova

June 24, 2006

1 Introduction

Discovery of frequent episodes is a data mining method for temporal data. Temporal data is a type of the data which refer to the time. There are several different tasks of the temporal mining as temporal clustering and other.
The main idea of discovery of the frequent episodes is to discover a frequently occurs sequence of events.
There are several different areas where data to be analyzed consist of sequence of the event. One of the problem, which can be analyzed in the sequence of events, is a discovery of the frequent episodes [MTV97]

An episode is a sequent of events which have some dependences between each other. This dependences appears in the order of occurring of events. Events in episodes occur close to each other. Proximity is shown in a order of the occurring. We can say that events close to each other if they are appears within of some time measure. However we should pay attention for the natural of the events.
So, we need to define such ideas as event, sequent of the events for introduce episodes. I this section I'll only introduce all ideas without formal definitions. Follow example represent these ideas.

Example 1 *Let us assume that we have go follow types of the events: A = "sun is shining", B = "it is cloudy", C= "it is rainy", D = "it is snowing", E = "it is hailing'. Let us define time moment when the types of the events can happen. Result of this operation will be these events which occur in some period of time. This set of events is event sequence S. For example sequent S= ((A,1),(B,2),(C,3)) demonstrate that at time step 1 sun was shining, at time step 2 was cloudy and at time step 3 was rain.*

Some of the events can occur after certain events very often, but some of them can not be next to each other in the time axis. For example, it is natural that event C occurs after event B, but it is almost impossible to find a sequent of events S= ((A, 1), (E, 2)).

Thus from the set of events it is possible to define collections of events occurring frequently together. These collections of events are called episodes. Following list represent episodes for Example 1 The follow events can form these episodes:

- A,B,C;
- A,C;
- B,C;
- B,D;
- B,E;
- B,D,E and others

The main types of the episodes are: serial, parallel, mixed depending on the order of events. pictures In the previous example it was very easy to find out episodes, but in case of large database of the events and another more difficult environment it is not so easy. The main algorithms for searching frequently episodes are described in the paper.

In this paper I will give an overview of the episodes. Definitions of an event, a sequence of events, an episodes, main type of the episodes are given in the section 2; the main principles of the discovery frequent episodes are described in the section 3; in the section 4 practical application of the discovering frequent episodes are reflected.

2 Event sequences and episodes

In this section I will give definition of the basic term in formal way. This part is continuation of previous which contains general description. First and the most important definition of the section is a definition of an event.

An *event* is something that happens at a given place and time. So it is possible to define event as a pair (A, t), where $A \in E$ is an event type and

t is an integer, the occurrence time of the event, and E is predefined set of event types.

According to [MTV97]:An *event sequence* s on E is a triple (s, T_s, T_e), where $s = ((A_1, t_1), (A_2, t_2), ..., (A_n, t_n)$, is an ordered sequence of events such that $A_i \in E$ for all $i \in 1, ..., n$, and $t_i \leq t_{i+1}$ for all $i = 1, ..., n-1$. Further on, T_s and T_e are integers: T_s is called the starting time and T_e the ending time, and $T_s \leq t_i < T_e$ for all $i \in 1, ..., n$.

Example 2 *Let us suppose that first of April was cloudy (B), second of April it was snowing (D), third of April was sun shining (A). So we have got the following ordered set of event* $s = ((B, 1), (D, 2), (A, 3))$ *and event sequence is follow triple* $(s, T_s, T_e) = (((B, 1), (D, 2), (A, 3)), 1, 3)$.

An episode is a set of the ordered events which occur close to each other. So to define an episode it is necessary to define some measure which describes the closeness of event which belong to one episode. Such a measure is called are *window*. The user can restrict time border of the episode by defining width of the window.

Formally [MTV97] ,a window is a part of an event sequence. A window on an event sequence $s = (s; T_s; T_e)$ is an event sequence $w = (w; t_s; t_e)$, where $t_s < T_e$ and $t_e > T_s$, and w consists of those pairs $(A; t)$ from s where $t_s \leq t < t_e$. The time span $t_e - t_s$ is called the *width of the window* w, and it is denoted $Width(w)$.

So after definitions of event sequent and window it is possible to define clear meaning of the episode. As it was mentioned above, an episode is a set of ordered events which occur close to each other. It is possible to define an ordered set of events as an event sequence, and closeness of occurring of events can be defined by using a time window. According to this an episode is a sequence of events which occur within the width of the window.

Depending on order of the events episodes can be:

- Serial episode;
- Parallel episode;
- non-serial and non-parallel episode.

Episodes can be described as directed acyclic graph [Woj00]. An episode α is a triple $(V; \leq; g)$.where V is a set of nodes, $\leq$ is a partial order on V, and

$g : V \rightarrow$E is a mapping associating each node withan event type [Woj00].
The interpretation of an episode is that the events in g(V) have to occur in the order described by $\leq$. The size of α, denoted $|\alpha|$, is $|V|$. Episode α is parallel if thepartial order $\leq$ is trivial (i.e.,$x \not\leq y$ for all $x, y \in V$ such that $x \neq y$). Episode α is serial if the relation $\leq$ is a total order (i.e., $x \leq y$ or $y \leq x$ for all $x, \in V$) [Woj00].
Serial episode is a sequence of events that occurs in the specified order. In the graph representation a serial episode corresponds to a single path from the first event of the episode to the last one.
Parallel episode is an unordered collection of events. In the graph representation a parallel episode corresponds to a single node containing all events of the episode. Formally, a parallel episode corresponds to the set of all permutations of symbols of the episode. [GAS05]

3 Algorithms

Discovering frequent episodes can be a difficult process, especially in the case with large database of events. In this part of paper I describe algorithm for discovering frequent episodes in a database of events.
There is description of the algorithm [MTV97] in this section. It is basic algorithm for discovering frequent episode. Another algorithms can be more appropriate in the different cases. For example, efficient algorithm which discovers rare episodes [LJD04]; algorithm for discovering frequent episodes in sequences of complex events [Woj00]
The problem formulation is following: "given an event sequent S, a desired class of episodes, a user defined time window size *win* and a minimal required number of the events occurrences for an episode l to be called frequent min_fr, discover all frequent episodes from S".
The algorithm implement *breadth first* search among the episodes. The method is based on dividing each class into a set of smaller subclasses. In the breadth-first search the all a child classes are processed before moving on to the next level.
The discovery of frequent episodes is the process, which require generation of the frequent episodes. Episodes containing one event description and then iteratively generate and verify candidate episodes of larger sizes. In k-th iteration candidate episodes of size k (containing k event descriptions) are generated from frequent episodes of size k-1. In each iteration, occurrences of candidate episodes in the event sequence are counted. Candidates that do not occur frequently enough are filtered out. The process stops when no candidates can be generated or no candidates of a certain size are found to

be frequent.
The discovery of the frequent episodes requires two steps:

- generation of candidate episodes;
- pruning of an infrequent episodes;

Let us describe both of the steps .
Candidate generation.

Let F_l=$\alpha_1, ..., \alpha_k$ be a set of the episodes,$k = |F_l|$. If $\alpha_i = s$ are parallel then each $\alpha_i = A_1, ..., Al$. If $\alpha_i = s$ are serial then each α_i is ordered set $\alpha_i =< A_1, ..., Al >$
In candidate generation we are given F_l and want to generate F_{l+1}. It is performed by combining two episodes from F_l, which contain $l-1$ common elements.

Example 3 *Two episodes α_i and α_j are combining if $\exists\alpha : \alpha < \alpha_i$ and $\alpha < \alpha_i, |\alpha| = l-1$*
The new parallel episode is $B = (A_1, A_2, ..., A_{l-1}, \alpha_i, A_l, \alpha_j, A_{l+1})$
In case when episode is serial we generate two new episodes B_1 and B_2
$B_1 = (A_1, A_2, ..., A_{l-1}, \alpha_i, A_l, \alpha_j, A_{l+1})$
$B = (A_1, A_2, ..., A_{l-1}, \alpha_j, A_l, \alpha_i, A_{l+1})$

Here is pseudo algorithm which represent candidate generation for parallel and serial episodes.

After episodes generation it is necessary to *prune infrequent episode.*
To define which episodes are infrequent it is necessary to use some measure. This measure is *time window.*
So we need to check that the candidate episode occurs in min_fr of window.
In the simple pruning strategy we check each $l - subset$ of the potential $(l+1) - element$ candidates one-by-one. If all subsets are found to be frequent, then the potential candidate becomes a real candidate.
We need to shift window through the sequence, step by step and check whether episode α occurs in the window. If does we will update the frequency counter for α. This step we need to make for all episodes $\alpha \in F_l$.
Both parallel and serial episodes can be checked in this way.

4 Discovery episodes from web logs

Episode discovery has been applaed in different areas. Process of discovery episode can be used in management, prediction, classification. One

Alg. 1 : An algorithm of candidate generating for parallel episodes

Input: $|F_l|$
Output: F_{l+1}

```
1    begin
2    initialize F_{l+1}
3    for i = 1 to |F_l|
4        for j = i to |F_l|
5            search such α_i, α_j ∈ F_l that α_i.A_1 = α_j.A_1, ..., α_i.A_{l-1} = α_j.A_{1-1}
6            create new episode B = A_1, A_2, ..., A_{l-1}, α_i.A_l, α_j.A_{l+1}
7            F_{l+1} = F_{l+1} ∪ B
8    return F_{l+1}
9    end
```

Alg. 2 : An Algorithm for candidate generating of serial episodes

Input: $|F_l|$
Output: F_{l+1}

```
1    begin
2    initialize F_{l+1}
3    for i = 1 to |F_l|
4        for j = 1 to |F_l|
5            search such α_i, α_j ∈ F_l that α_i.A_1 = α_j.A_1, ..., α_i.A_{l-1} = α_j.A_{1-1}
6            create new episode B =< A_1, A_2, ..., A_{l-1}, α_i.A_l, α_j.A_{l+1} >
7            F_{l+1} = F_{l+1} ∪ B
8    return F_{l+1}
9    end
```

of the application of the discovery episodes is a analysis of information in WWW. [PHMAZ00]
The rapid expansion of the World Wide Web has created an unprecedented opportunity to disseminate and gather information online. As more data are becoming available, there is much need to study web-user behaviors to better serve the users and increase the value of enterprises. One important data source for this study is the web-log data that traces the user's web browsing. Web mining is a application of data mining technologies to the data reposi-

tories. It is possible to discover Web content and Web mining. The process of the extracting knowledge about content of the site are known as Web log mining.
Web servers register a Web log entry for every single access they get, in which important pieces of information about accessing are recorded, including the URL requested, the IP address from which the request originated, and a timestamp.
Web log can be regarded as a sequence of pairs of user identifier and event. Web log files are divided into pieces. Preprocessing can be applied to the original Web log files, and pieces of Web logs can be obtained. Each piece of Web log is a sequence of events from one user or session. The events are recordered in time order.
For example, let E be a set of events. A Web log piece or (Web) access sequence $S = e_1, e_2, ..., e_n (e_i \in E)$ for $(1 \leq i \leq n)$ is a sequence of events, whole n is called the length of the access sequence. An access sequence with length n is also called an n-sequence.
Access sequence $\acute{S}a = \acute{e}_1\acute{e}_2...\acute{e}_n$ is called a *subsequence* of access sequence $S = e_1, e_2, ..., e_n (e_i \in E)$, and S a super-sequence of $\acute{S}$, denoted as $\acute{S}a \subset S$,
Given a set of access sequence $WAS = (S_1; S_2; ...; S_m)$, called *Webaccess* sequence database, in which $S_i(1\ leqi \leq m)$ are access sequences. The support of access sequence S in WAS is dented as $sup_{WAS}(S)$
So, the process of web access possible to represent as episode of events. Therefore it is possible to apply discovery frequent algorithm for searching frequent episodes.

5 Conclusions

In this paper we have represented an overview of discovering episodes in the sequent data. In the second chapter was represented basic definitions and basic types of the episode.
Another importance of the paper that main principles of the algorithms for discovering frequent episodes was represented here. These principles can be used for searching the episodes in a different environments.
Discovery of frequent episodes can be applied for analysis in different areas where a data can be represented as a sequence of the events. Discovery episodes from web logs can be such area.

References

[GAS05] Robert Gwadera, Mikhail J. Atallah, and Wojciech Szpankowski. Markov models for identification of significant episodes. In *SDM*, 2005.

[LJD04] Dan Li, Liying Jiang, and Jitender S. Deogun. Temporal knowledge discovery with infrequent episodes. In *dg.o '04: Proceedings of the 2004 annual national conference on Digital government research*, pages 1–2. Digital Government Research Center, 2004.

[MTV97] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1(3):259–289, 1997.

[PHMAZ00] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, and Hua Zhu. Mining access patterns efficiently from web logs. In *PADKK '00: Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Current Issues and New Applications*, pages 396–407, London, UK, 2000. Springer-Verlag.

[Woj00] Marek Wojciechowski. Discovering frequent episodes in sequences of complex events. In *ADBIS-DASFAA Symposium*, pages 205–214, 2000.