Constraint-Based Mining of Episode Rules and Optimal Window Sizes*

Nicolas Méger and Christophe Rigotti

INSA-LIRIS FRE CNRS 2672
69621 Villeurbanne Cedex, France
{nmeger,crigotti}@liris.cnrs.fr

Abstract. Episode rules are patterns that can be extracted from a large event sequence, to suggest to experts possible dependencies among occurrences of event types. The corresponding mining approaches have been designed to find rules under a temporal constraint that specifies the maximum elapsed time between the first and the last event of the occurrences of the patterns (i.e., a window size constraint). In some applications the appropriate window size is not known, and furthermore, this size is not the same for different rules. To cope with this class of applications, it has been recently proposed in [2] to specifying the maximal elapsed time between two events (i.e., a maximum gap constraint) instead of a window size constraint. Unfortunately, we show that the algorithm proposed to handle the maximum gap constraint is not complete. In this paper we present a sound and complete algorithm to mine episode rules under the maximum gap constraint, and propose to find, for each rule, the window size corresponding to a local maximum of confidence. We show that the extraction can be efficiently performed in practice on real and synthetic datasets. Finally the experiments show that the notion of local maximum of confidence is significant in practice, since no local maximum are found in random datasets, while they can be found in real ones.

1 Introduction

Many datasets are composed of a large sequence of events, where each event is described by a date of occurrence and an event type. Commonly mined descriptive patterns in these datasets are the so-called *episode rules*. Informally, an episode rule reflects how often a particular group G_1 of event types tends to appear close to another group G_2. A rule is associated with two measures, its frequency and its confidence, that have an intuitive reading similar to the one of frequency and confidence used for *association rules* [1]. These two measures respectively represent how often the two groups occur together (i.e., is the rule supported by many examples ?) and how *strong* is this rule (i.e., when G_2 occurs, is G_1 appearing in many cases close to G_2 ?).

* This research is partially funded by the European Commission IST Programme - Accompanying Measures, AEGIS project (IST-2000-26450).

Finding episode rules may provide interesting insight to experts in various domains. In particular, it has been shown to be very useful for alarm log analysis in the context of the TASA project [3]. More generally, it can also be applied, after an appropriated discretization to time series, and to spatial data (the temporal dimension is replaced by a spatial dimension) like for example in DNA sequences.

The standard *episode rule mining problem* is to find all episode rules satisfying given frequency and confidence constraints. There are two main approaches to find such rules. The first one, proposed and used by [7, 6] in the *Winepi* algorithm, is based on the occurrences of the patterns in a sliding window along the sequence. The second one, introduced in [5, 6] and supported by the *Minepi* algorithm, relies on the notion of *minimal occurrences* of patterns. Both techniques have been designed to be run using a maximum window size constraint that specifies the maximum elapsed time between the first and the last event of the occurrences of the patterns. More precisely, in the case of *Winepi*, the algorithm used a single window size constraint and must be executed again if the user wants to perform an extraction with a different window size. The other algorithm, *Minepi*, needs a maximal window size constraint to restrict reasonably the search space in practice, but can derive rules for several window sizes that are lesser than this maximal window size.

To our knowledge, no existing complete algorithm is able to extract episode rules without at least a maximum window size constraint (in addition to a frequency and a confidence constraint) on non-trivial datasets. In some applications the window size is not known beforehand, and moreover, the interesting window size may be different for each episode rule. To cope with this class of applications, it has been recently proposed in [2] to use a maximum gap constraint that imposes the maximal elapsed time between two consecutive events in the occurrences of an episode. This constraint allows the occurrences of larger patterns to spread over larger intervals of time not directly bounded by a maximum window size. This constraint is similar to the maximum gap constraint handled by algorithms proposed to find frequent sequential patterns in a base of sequences (e.g., [10, 11, 4]). A base of sequences is a large collection of sequences where each sequence is rather small, and the algorithms developed to mine such bases cannot be reused to extract episodes in a single large event sequence, because the notion of frequency of a pattern is very different in these two contexts. In a base of sequences the frequency of a pattern corresponds to the number of sequences in which the pattern occurs at least one time, and several occurrences of the pattern in the same sequence have no impact on its frequency. While in the case of an episode in an event sequence, the frequency represents the number of occurrences of the pattern in this sequence.

Thus [2] has proposed a new algorithm, but as we will show in Section 2.3 this algorithm is not complete. However, the contribution of [2] remains interesting because this work suggests that mining episode rules in practice could be done using a maximum gap constraint.

In this paper our contribution is twofold. Firstly, we present a sound and complete algorithm to extract episode rules satisfying frequency, confidence and maximum gap constraints. And secondly, we propose a way to find if it exists, for each rule, the smallest window size that corresponds to a local maximum of confidence for the rule (i.e., confidence is locally lower, for smaller and larger windows).

From a quantitative point of view, we present experiments showing that mining episode rules under the maximum gap constraint and finding the local maximums of confidence can be done in practice at reasonable extraction thresholds. From a qualitative point of view, these experiments advocated the fact that local maximums of confidence can be interesting suggestions of possible dependencies to the expert, because no local maximum have been found on synthetic random datasets, while they exist in real data. Finally, the experiments show that the number of rules satisfying the frequency and confidence constraints and having a local maximum of confidence is orders of magnitude lesser than the number of rules satisfying the frequency and confidence constraints only. So, in practice the expert has to browse only a very limited collection of extracted patterns.

This paper is organized as follows. The next section gives preliminary definitions and shows that the algorithm presented in [2] is incomplete. Section 3 introduces the algorithm *WinMiner* that handles the maximum gap constraints and finds the local maximums of confidence of episode rules. Section 4 presents experiments performed, and we conclude with a summary in Section 5.

2 Episode Rules and Local Maximum of Confidence

2.1 Preliminary definitions

In this section we follow the standard notions of event sequence, episode, minimal occurrences and support used in [6] or give equivalent definition, when more appropriated to our presentation. The only noticeable difference is that our notion of occurrence incorporates the necessity for the occurrences to satisfy a maximum gap constraint.

Definition 1. ***(event, ordered sequence of events)*** Let E be a set of *event types*. An *event* is defined by the pair (e, t) where $e \in E$ and $t \in \mathbb{N}$. The value t denotes the time at which the *event* occurs. An *ordered sequence of events* s is a tuple $s = \langle (e_1, t_1), (e_2, t_2), ..., (e_n, t_n) \rangle$ such that $\forall i \in \{1, ..., n\}, e_i \in E \wedge\ t_i \in \mathbb{N}$ and $\forall i \in \{1, ..., n-1\}, t_i \leq t_{i+1}$.

Definition 2. ***(operator $\sqsubseteq$)*** Let α and β be two ordered sequences of events, then α is a subsequence of β, denoted $\alpha \sqsubseteq \beta$ iff α can be obtained by removing some elements of β or $\alpha = \beta$.

Definition 3. ***(event sequence)*** An *event sequence* S is a triple (s, T_s, T_e), where s is an ordered sequence of events of the form $\langle (e_1, t_1), (e_2, t_2), ..., (e_n, t_n) \rangle$ and T_s, T_e are natural numbers such that $T_s \leq t_1 \leq t_n \leq T_e$.

T_s and T_e respectively represent the starting time and the ending time of the event sequence. Notice that t_1 may differ from T_s and that t_n may differ from T_e.

Definition 4. ***(episode)*** An *episode* is a tuple α of the form $\alpha = \langle e_1, e_2, \ldots, e_k \rangle$ with $e_i \in E$ for all $i \in \{1, \ldots, k\}$. In this paper, we will use the notation $e_1 \rightarrow e_2 \rightarrow ... \rightarrow e_k$ to denote the episode $\langle e_1, e_2, ..., e_k \rangle$ where '$\rightarrow$' may be read as 'followed by'. We denote the empty episode by $\emptyset$.

Definition 5. ***(size, suffix and prefix of an episode)*** Let $\alpha = \langle e_1, e_2, ..., e_k \rangle$ be an episode. The *size* of α is denoted $|\alpha|$ and is equal to the number of elements of the tuple α, i.e., $|\alpha| = k$. The $suffix$ of α is defined as an episode composed only by the last element of the tuple α, i.e., $suffix(\alpha) = \langle e_k \rangle$. The $prefix$ of α is the episode $\langle e_1, e_2, ..., e_{k-1} \rangle$. We denote it as $prefix(\alpha)$.

Definition 6. ***(occurrence)*** An episode $\alpha = \langle e_1, e_2, ..., e_k \rangle$ *occurs* in an event sequence $S = (s, T_s, T_e)$ if there exists at least one *ordered sequence of events* $s' = \langle (e_1, t_1), (e_2, t_2), ..., (e_k, t_k) \rangle$ such that $s' \sqsubseteq s$ and $\forall i \in \{1, ..., k-1\}, 0 < t_{i+1} - t_i \leq gapmax$ with $gapmax$ a user-defined threshold that represents the maximum time gap allowed between two consecutive events.

The interval $[t_1, t_k]$ is called an *occurrence* of α in S. The set of all the occurrences of α in S is denoted by $occ(\alpha, S)$.

These episodes and their occurrences correspond to the *serial* episodes of [6], up to the following restriction: the event types of an episode must occur at different time stamps in the event sequence. This restriction is imposed here for the sake of simplicity, and the definitions and algorithms can be extended to allow several event types to appear at the same time stamp. However, it should be noticed that this constraint applies on occurrences of the patterns, and not on the dataset (i.e., several events can occur at the same time stamp in the event sequence).

Definition 7. ***(minimal occurrence)*** Let $[t_s, t_e]$ be an occurrence of an episode α in the event sequence S. If there is no other occurrence $[t'_s, t'_e]$ such that $(t_s < t'_s \wedge t'_e \leq t_e) \vee (t_s \leq t'_s \wedge t'_e < t_e)$ (i.e., $[t'_s, t'_e] \subset [t_s, t_e]$), then the interval $[t_s, t_e]$ is called a *minimal occurrence* of α. The set of all minimal occurrences of α in S is denoted by $mo(\alpha, S)$.

Intuitively, a minimal occurrence is simply an occurrence that does not contain another occurrence of the same episode.

Definition 8. ***(width of an occurrence)*** Let $o = [t_s, t_e]$ be an occurrence. The time span $t_e - t_s$ is called the *width* of the occurrence o . We denote it as $width(o)$. The set of all occurrences (resp. minimal occurrences) of an episode α in an event sequence S having a width equal to w is denoted $occ(\alpha, S, w)$ (resp. $mo(\alpha, S, w)$).

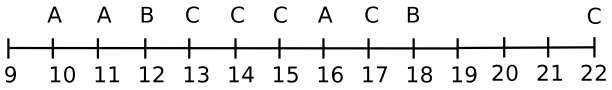

Fig. 1. Example of event sequence.

Definition 9. ***(support of an episode)*** The *support* of an episode α in an event sequence S for a width w is defined as $Support(\alpha, S, w) = \sum_{0 \leq i \leq w} \mid mo(\alpha, S, i) \mid$. We also define the *general support* of α in S as $GSupport(\alpha, S) = \mid mo(\alpha, S) \mid$

The notions of occurrence and support of an episode incorporate the satisfaction of the maximum gap constraint. However, for the sake of simplicity, the *gapmax* parameter does not appear explicitely in the notational conventions $mo(\alpha, S, i), Support(\alpha, S, w), GSupport(\alpha, S)$ and $mo(\alpha, S)$.

To illustrate some of the previous definitions, we consider the event sequence $S = (w, T_s, T_e)$ of Figure 1.

In this example, $T_s = 9, T_e = 22, w = \langle (A, 10), (A, 11), (B, 12), (C, 13), (C, 14), (C, 15), (A, 16), (C, 17), (B, 18), (C, 22) \rangle$. If we consider the episode $\alpha = A \rightarrow B$, and the constraint $gapmax = 3$, then $occ(\alpha, S) = \{[10, 12], [11, 12], [16, 18]\}$. It should be noticed that $[10, 18]$ does not belong to $occ(\alpha, S)$ since the constraint *gapmax* is not satisfied. The minimal occurrences of α are $mo(\alpha, S) = \{[11, 12], [16, 18]\}$. The occurrence $[10, 12]$ does not belong to $mo(\alpha, S)$ because it contains the occurrence $[11, 12]$. In the same way, in the case of episode $\beta = A \rightarrow B \rightarrow C$, we have $occ(\beta, S) = \{[10, 13], [10, 14], [10, 15], [11, 13], [11, 14], [11, 15]\}$ and $mo(\beta, S) = \{[11, 13]\}$. Some examples of support values are $GSupport(\alpha, S) = 2$, $Support(\alpha, S, 1) = 1$, $Support(\alpha, S, 2) = 2$ and $Support(\beta, S, 2) = 1$. It should be noticed that the support increases in a monotonic way, with respect to the width value.

2.2 Episode rule and local maximum of confidence

Definition 10. ***(episode rule)*** Let α and β be episodes such that $prefix(\beta) = \alpha$. An *episode rule* built on α and β is the expression $\alpha \Rightarrow suffix(\beta)$.

For example, if $\alpha = e_1 \rightarrow e_2$ and $\beta = e_1 \rightarrow e_2 \rightarrow e_3$, the corresponding episode rule is denoted $e_1 \rightarrow e_2 \Rightarrow e_3$. It should be noticed that the episode rules used in this paper are restricted to rules having a single event type in their right hand sides, but that the definitions and algorithms proposed can be extended in the case of right hand sides containing several event types.

Definition 11. ***(support and confidence of an episode rule)*** The *support* of an episode rule is defined by $Support(\alpha \Rightarrow suffix(\beta), S, w) = Support(\beta, S, w)$.

The *confidence* of an episode rule is defined as follows:

$$Confidence(\alpha \Rightarrow suffix(\beta), S, w) = \frac{Support(\beta,S,w)}{Support(\alpha,S,w)}$$

Let γ be a user defined confidence threshold such that $0 \leq \gamma \leq 1$, and let σ be a user defined support threshold such that $0 < \sigma \leq 1$. Then, if $Confidence(r, S, w) \geq \gamma$ (resp. $Support(r, S, w) \geq \sigma$) the rule is said to be $confident$ (resp. $frequent$) for the width w.

It should be noticed that, as for episodes, the support and confidence are defined with respect to a given width. The definition of confidence can be illustrated by the previous example (Figure 1). Knowing that $mo(A \rightarrow B, S) = \{[11, 12], [16, 18]\}$ and $mo(A \rightarrow B \rightarrow C, S) = \{[11, 13]\}$, we have $Confidence(A \rightarrow B \Rightarrow C, S, 2) = 1/2$.

Definition 12. ***(LM and FLM)*** A rule r is said to have a LM ($Local\ Maximum$) for a given width i on event sequence S iff the three following properties are satisfied:

- $Confidence(r, S, i) \geq \gamma \wedge Support(r, S, i) \geq \sigma$
- $\forall j,\ j < i \wedge Support(r, S, j) \geq \sigma \ \Rightarrow\ Confidence(r, S, i) > Confidence(r, S, j)$
- $\exists j,\ i < j \wedge Confidence(r, S, j) \leq Confidence(r, S, i) - (decRate * Confidence(r, S, i))$ with $decRate$ a decrease threshold defined by the user, and $\forall\ k,\ i < k < j \ \Rightarrow\ Confidence(r, S, k) \leq Confidence(r, S, i)$

The rule r has a $FLM(First\ Local\ Maximum)$ for width i iff r has a LM for width i, and r has no LM for width strictly lesser than i. A rule having at least one LM, and thus also a (single) FLM, is called a $FLM - rule$.

Intuitively, a LM for a rule r is a width w such that (1) r is frequent and confident for this width, (2) all lower width values such that r is frequent correspond to a strictly lower confidence, and (3) the next consecutive greater width values correspond also to a lower confidence until the confidence becomes lower than a given percentage ($decRate$) of confidence obtained for w. The figure 2 illustrates, for a given rule, possible variations of confidence with respect to the width values and the corresponding FLM are represented by a dot. The vertical axis represents the confidence of the rule and the horizontal dashed line indicates the confidence threshold γ. The horizontal axis represents the width values, and the vertical dashed line corresponds to a specific width, denoted w_σ, that is the width at which the rule turns out to be frequent. Two particular situations should be pointed out. Firstly, the bottom-left graphic where there is no FLM. And secondly, the bottom-right graphic, where the three first local maximums are not valid LM because there are not followed by a sufficient decrease of confidence.

2.3 Incompleteness of [2]

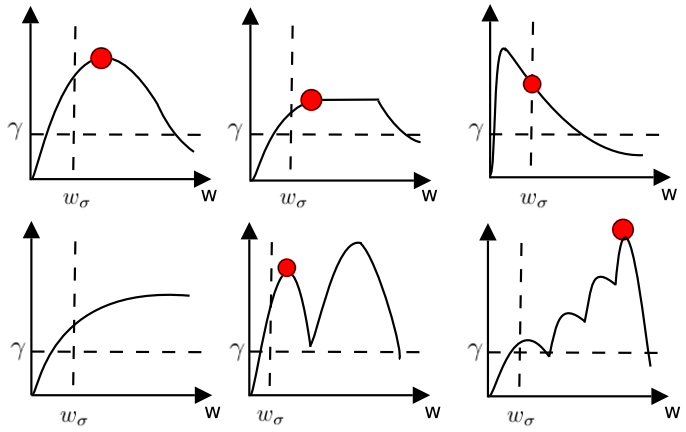

Fig. 2. Confidence vs width

In [2] an algorithm has been proposed to extract frequent and confident episode rules under a maximum gap constraint, but the formal correctness of the algorithm has not been established. Indeed, we have remarked that the algorithm is not complete. To show this, we need to recall the definition of *window* of [6] : *"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 $t_s \leq t < t_e$".*

Let $\mathcal{W}(S, win)$ be the set of all windows (w, t_s, t_e) on S such that $t_e - t_s = win$. Then [2] defines by $fr(\alpha) = \frac{|\{W \in \mathcal{W}(S,win) | \alpha \ occurs \ in \ W\}|}{|\mathcal{W}(S,win)|}$ where $win = (|\alpha| - 1) \times gapmax$, the frequency of an episode α under a maximum gap constraint. In [2] the episode $A \rightarrow B \rightarrow C$ is considered only if the episodes $A \rightarrow B$ and $B \rightarrow C$ are frequent. Let S be the event sequence $(\langle (A, 5), (B, 8), (C, 9) \rangle, 1, 10)$ and $gapmax = 4$. Then $fr(A \rightarrow B) = 1/12$ and $fr(A \rightarrow B \rightarrow C) = 4/16 = 1/4$. If the frequency threshold is $1/4$, then $A \rightarrow B$ is not frequent and thus $A \rightarrow B \rightarrow C$ is not considered by the algorithm even though it is frequent.

3 Extraction of all FLM-rules

The proofs of the theorems and lemmas of this section are given in the extended version of this paper [9].

To extract the FLM-rules we propose an algorithm, called *WinMiner*, based on the so-called *occurrence list* approach, used in [6] and also in the context of mining sequential patterns in base of sequences (e.g., [11, 4]). The key idea of such methods is to store and use occurrences of patterns to compute the occurrences of longer patterns by means of a kind of temporal join operation. In our case, if

we only keep the information about the location of all the minimal occurrences of the episodes of size k, it is not possible to produce all the minimal occurrences of the episodes of size $k+1$. Looking back to the example given Figure 1, we can see that there exists a minimal occurrence $[11, 18]$ of $A \rightarrow B \rightarrow C \rightarrow B$. If we only know $mo(A \rightarrow B \rightarrow C, S) = \{[11, 13]\}$ and $mo(B, S) = \{[12, 12], [18, 18]\}$, we can not determine the minimal occurrence of $A \rightarrow B \rightarrow C \rightarrow B$. This is because the first B occurs too early (before time stamp 18) and the second one occurs too late (the maximum gap constraint of 3 is not satisfied between time stamps 13 and 18). To overcome this difficulty, *WinMiner* is based on a notion of *minimal prefix occurrence* introduced in the next section.

3.1 Minimal prefix occurrence

Definition 13. ***(minimal prefix occurrence)*** Let $o = [t_s, t_e]$ be an occurrence of an episode α in the event sequence S, then o is a *minimal prefix occurrence* (*mpo*) of α iff $\forall\ [t_1, t_2] \in mo(prefix(\alpha), S)$, if $t_s < t_1$ then $t_e \leq t_2$. We denote by $mpo(\alpha, S)$ the set of all *mpo* of α in S.

It is important to notice that $mpo(\alpha, S)$ is defined with respect to $mo(prefix(\alpha), S)$ and not with respect to $mo(\alpha, S)$. In the example depicted Figure 1, we have $mpo(A \rightarrow B \rightarrow C, S) = \{[11, 13], [11, 14], [11, 15]\}$ and $mpo(B, S) = \{[12, 12], [18, 18]\}$. Then, using these sets it is possible to built $mpo(A \rightarrow B \rightarrow C \rightarrow B, S) = \{[11, 18]\}$. As it can be intuitively noticed, the minimal occurrences are particular *mpo* and the minimal occurrences can be determined using the set of *mpo*. This is formally stated by the two following lemmas:

Lemma 1. *If* $[t_s, t_e] \in mo(\alpha, S)$ *then* $[t_s, t_e] \in mpo(\alpha, S)$ *and there is no* $[t_s, t'_e] \in mpo(\alpha, S)$ *such that* $t'_e < t_e$.

Lemma 2. *If* $[t_s, t_e] \in mpo(\alpha, S)$ *and there is no* $[t_s, t'_e] \in mpo(\alpha, S)$ *such that* $t'_e < t_e$, *then* $[t_s, t_e] \in mo(\alpha, S)$.

E/O-pair. The algorithm *WinMiner* handles an episode α and its *mpo* in a pair of the form $(episode, occurrences)$ called E/O-pair. For a E/O-pair x, we denote respectively $x.Pattern$ and $x.Occ$, the first and second element of the pair. The $x.Pattern$ part represents the episode itself and $x.Occ$ contains its *mpo* in a compact way. The $x.Occ$ part is a set of pairs of the form $(Tbeg, TendSet)$ where $Tbeg$ represents a *mpo* starting time and $TendSet$ is the set of the ending times of all *mpo* of $x.Pattern$ starting at $Tbeg$. Intuitively, the interest of this representation is that, according to lemma 2, if we consider a pair $(Tbeg, TendSet)$ then the interval $[Tbeg, min(TendSet)]$ represents a minimal occurrence of $x.Pattern$.

3.2 Algorithm *WinMiner*

This algorithm extract all $FLM - rules$ in a event sequence S, according to the following user-specified parameters: a support threshold σ, a confidence threshold γ, a maximum time gap constraint $gapmax$ and a decrease threshold $decRate$.

The algorithm is presented as Algorithm 1. First, it computes the *mpo* of all frequent episodes of size 1, using a function named *scan*, that is not detailed in this paper, but that simply determines the *mpo* of an episode by scanning S. The algorithm then calls the function *exploreLevelN* (Algorithm 2) to find in a depth-first way all episodes (of size greater than 1) such that their *GSupport* are greater or equal to the threshold σ. For a given episode $x.Pattern$ this function extends the pattern on the right side with a frequent episode $y.Pattern$ of size 1. This is performed by a call to the *join* function (Algorithm 3), that computes the new pattern $z.Pattern = x.Pattern \rightarrow y.Pattern$, and also the corresponding set of *mpo* in $z.Occ$ using the *mpo* of $x.Pattern$ and $y.Pattern$ stored respectively in $x.Occ$ and $y.Occ$. The core part of this join operation is the line 6, where it checks that the new interval generated ($[t_s, t'_s]$) is an occurrence of $z.Pattern$ satisfying the maximum gap constraint (condition $t'_s > t_s \wedge t'_s - t \leq gapmax$) and that this interval is a *mpo* of $z.Pattern$ (condition $\forall (t_1, T'') \in x.Occ\ t_s < t_1 \Rightarrow \forall t_2 \in T'', t'_s \leq t_2$). Then, after the call to *join*, if the *GSupport* of $z.Pattern$ is greater or equal to σ, the Algorithm 2 determines, if it exists, the FLM of the rule built from the prefix and suffix of $z.Pattern$, by means of the function $findFLM$. Finally, the algorithm 2 considers, in a recursive way, the episodes that can be obtained by adding frequent episodes of size 1 to $z.Pattern$ itself.

Algorithm 1 (*WinMiner*)
Input: *S an event sequence S and E the set of event types.*

```
1.  let L_1 := ∅
2.  for all e ∈ E do
3.      let x.Pattern := e
4.      let x.Occ := scan(S, e)
5.      if | x.Occ | >= σ
6.          let L_1 := L_1 ∪ {x}
7.      fi
8.  od
9.  for all x ∈ L_1 do
10.     exploreLevelN(x, L_1)
11. fi
```

Algorithm 2 (*exploreLevelN*)
Input: *x a E/O-pair, and L_1 the set of E/O-pairs of frequent episodes of size 1.*

```
1.  for all y ∈ L_1 do
2.    let z := join(x, y)
3.    if |z.Occ| >= σ
4.      findFLM(x.Pattern ⇒
        suffix(z.Pattern), x.Occ, z.Occ)
5.      exploreLevelN(z, L_1)
7.    fi
8.  od
```

Algorithm 3 (*join*) Input: *x and y, two E/O-pairs, containing an episode and its set of mpo, and where y corresponds to an episode of size 1.*

Output: *z, a E/O-pair containing the episode $x.Pattern \rightarrow y.Pattern$ and its set of mpo.*

```
1.  let z.Pattern := x.Pattern → y.Pattern
2.  let z.Occ := ∅
3.  for all (t_s, T) ∈ x.Occ do
4.    let L := ∅
```

5. **for all** $t \in T$ **do**
6. **let** $EndingTimes := \{t'_s \mid \exists (t'_s, T') \in y.Occ$ *such that*
$t'_s > t_s \wedge t'_s - t \leq gapmax \wedge \forall (t_1, T'') \in x.Occ,$
$t_s < t_1 \Rightarrow \forall t_2 \in T'', t'_s \leq t_2\}$
7. **let** $L := L \cup EndingTimes$
8. **od**
9. **if** $L \neq \emptyset$
10. **let** $z.Occ := z.Occ \cup \{(t_s, L)\}$
11. **fi**
12. **od**

Let us now consider the correctness of the approach.

Definition 14. Let S be an event sequence, then

- a E/O-pair x is sound iff $\forall (t_s, T) \in x.Occ, \forall t \in T, [t_s, t] \in mpo(x.Pattern, S)$.
- a E/O-pair x is complete iff $\forall [t_s, t_e] \in mpo(x.Pattern, S)$, $\exists (t_s, T) \in x.Occ$ s.t. $t_e \in T$.
- a E/O-pair x is non-redundant iff $\forall (t_s, T) \in x.Occ, \nexists (t_s, T') \in x.Occ$ s.t. $T \neq T'$.

The following theorem states the correctness of the function *join* (Algorithm 3):

Theorem 1 (correctness of *join*). *If x.Occ and y.Occ in the input of join are sound, complete and non-redundant, then z.Occ in the output is sound, complete and non-redundant.*

Theorem 2 (correctness of support counting). *Let S be an event sequence and z be a E/O-pair outputted by Algorithm 3, for sound, complete and non-redundant x.Occ and y.Occ in the input, then:*

- *the number of minimal occurrences of $z.Pattern$ of width w is $|mo(z.Pattern, S, w)| = |\{(t_s, T) \in z.Occ | min(T) - t_s = w\}|$*
- $Support(z.Pattern, S, w) = \sum_{0 \leq i \leq w} |\{(t_s, T) \in z.Occ | min(T) - t_s = i\}|$
- $GSupport(\alpha, S) = |\{(t_s, T) \in z.Occ\}|$

Since $GSupport(\alpha, S) \geq \sigma \Rightarrow GSupport(prefix(\alpha), S) \geq \sigma$, then by the theorems 1 and 2, the depth-first enumeration of *WinMiner* is correct to find all episodes such that *GSupport* is greater or equal to σ. Furthermore, if an episode rule $prefix(\alpha) \Rightarrow suffix(\alpha)$ is frequent for a given width w then we also have $GSupport(\alpha, S) \geq \sigma$. Thus, by the theorems 1 and 2, we can also correctly find the support and confidence of a rule for a given width w, when the rule is frequent for this w.

So, we have at hand all the information necessary to determine if a rule such that $GSupport \geq \sigma$ is a FLM-rule and the width corresponding to its FLM. Due to space limitation, the corresponding algorithm (function $findFLM$) is not presented here, but can be found in the extended version of this paper [9].

4 Experiments

In this section we present experiments on a real dataset, using an implementation of *WinMiner* in C++, and performed on an Intel Pentium IV 2 GHz under a 2.4 Linux kernel (all the experiments were run using between 0.5 and 300 MB of RAM). Efficient implementation hints are given in [9]. Experiments on large random datasets are also presented in [9], and are not described here because of space limitation. These experiments show that the extractions can be done in practice in non-trivial cases and that no FLM-rule was found in these random datasets. Other experiments on atherosclerosis risk factors (atherosclerosis is the main cause of cardio-vascular diseases) are described in [8].

The experiments reported here were performed within the European Project AEGIS (IST-2000-26450) in collaboration with geophysicists to help them to find dependencies between earthquakes. In this paper, we only present experiments on a subset of the ANSS Composite Earthquake Catalog[1], that contains a series of earthquakes described by their locations, occurrence times and magnitudes. As the FLM-rules obtained in these experiments have suggested to the geophysicists some possible dependencies that are not at that time published in the geophysics literature, we cannot give the precise magnitudes and locations of the earthquakes considered in this subset of the catalog. After an appropriated discretization[2] the resulting dataset was built on 368 event types and contained 3509 events spread over 14504 time units of one day (about 40 years).

For the sake of conciseness, we only present, in Figure 3, extractions performed with $\sigma = 10$, $\gamma = 0.9$, $decRate = 30\%$ and $gapmax$ ranging from 100 to 140. The left graphic indicates that the running time is reasonable in practice (from less than 100 seconds to about 6700 seconds for the largest one). The right graphic presents, for each of the experiments, (1) the number of rules considered by *WinMiner* during the extraction (rules such that $GSupport \geq \sigma$), (2) the number of rules for which there exists a width w such that the rule is frequent and confident for w, and finally, (3) the number of FLM-rules. This graphic shows in particular that the collection of rules that are frequent and confident (for some width) is too huge to be handled by an expert, while the size of the collection of FLM-rules is several orders of magnitude smaller.

5 Conclusion

In this paper we presented a sound and complete algorithm to extract episode rules satisfying a maximum gap constraint. We also proposed to determine the window sizes corresponding to a local maximum of confidence. The experiments

[1] Public world-wide earthquake catalog available at http://quake.geo.berkeley.edu/cnss/

[2] This preprocessing has been performed by Francesco Pacchiani of Laboratoire de Geologie at Ecole Normale Supérieure of Paris in the context of the AEGIS project.

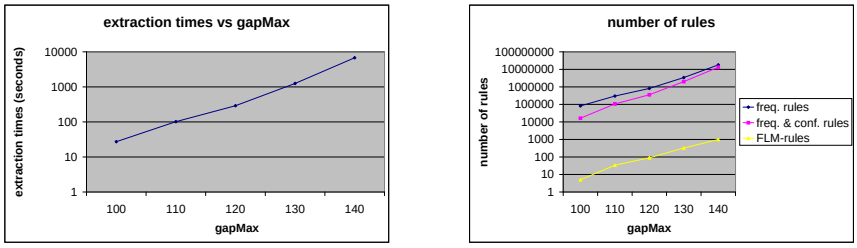

Fig. 3. Experiments on a seismic dataset

showed that extracting episode rules under the maximum gap constraint and finding the window sizes leading to a local maximum of confidence, can be efficiently performed in practice. Furthermore, no local maximum has been found on random datasets, while meaningful dependencies corresponding to local maximum of confidence have been found in a real seismic dataset.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proc. of the Int. Conf. SIGMOD'93*, pages 207–216, Washington D.C., USA, May 1993.
2. G. Casas-Garriga. Discovering unbounded episodes in sequential data. In *Proc. of the Int. Conf. PKDD'03*, pages 83–94, Croatia, September 2003. LNCS 2838.
3. K. Hatonen, M. Klemettinen, H. Mannila, P. Ronkainen, and H. Toivonen. Tasa: Telecomunications alarm sequence analyzer or: How to enjoy faults in your network. In *Int. Symp. NOMS'96*, pages 520–529, Kyoto, Japan, April 1996.
4. M. Leleu, C. Rigotti, J.-F. Boulicaut, and G. Euvrard. Constrained-based mining of sequential patterns over datasets with consecutive repetitions. In *Proc. of the Int. Conf. PKDD'03*, pages 303–314, Croatia, September 2003. LNCS 2838.
5. H. Mannila and H. Toivonen. Discovery of generalized episodes using minimal occurrences. In *Proc. of the 2nd Int. Conf. KDD'96*, pages 146–151, Portland, Oregon, August 1996.
6. H. Mannila, H. Toivonen, and A. Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1(3):259–298, November 1997.
7. H. Mannila, H. Toivonen, and I. Verkamo. Discovering frequent episodes in sequences. In *Proc. of the 1st Int. Conf. KDD'95*, pages 210–215, Canada, August 1995.
8. N. Méger, C. Leschi, N. Lucas, and C. Rigotti. Mining episode rules in STULONG dataset. Technical report, LIRIS Lab, Lyon, France, June 2004.
9. N. Méger and C. Rigotti. Constraint-based mining of episode rules and optimal window sizes. Technical report, LIRIS Lab, Lyon, France, June 2004.
10. R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *Proc. of the 5th Int. Conf. EDBT'96*, pages 3–17, Avignon, France, September 1996.
11. M. Zaki. Sequence mining in categorical domains: incorporating constraints. In *Proc. of the 9th Int. Conf. on CIKM'00*, pages 422–429, USA, November 2000.