MARBLES: Mining Association Rules Buried in Long Event Sequences[*]

Boris Cule[**], Nikolaj Tatti[***], and Bart Goethals[**]

[**]Department of Mathematics and Computer Science, University of Antwerp,
firstname.lastname@ua.ac.be
[***]DTAI, KU Leuven,
HIIT, Department of Information and Computer Science, Aalto University
nikolaj.tatti@aalto.fi

Abstract

Sequential pattern discovery is a well-studied field in data mining. Episodes are sequential patterns that describe events that often occur in the vicinity of each other. Episodes can impose restrictions on the order of the events, which makes them a versatile technique for describing complex patterns in the sequence. Most of the research on episodes deals with special cases such as serial and parallel episodes, while discovering general episodes is surprisingly understudied. This is particularly true when it comes to discovering association rules between them.

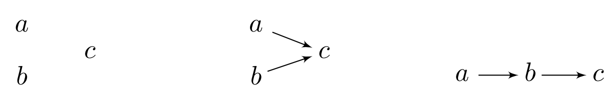
In this paper we propose an algorithm that mines association rules between two general episodes. On top of the traditional definitions of frequency and confidence, we introduce two novel confidence measures for the rules. The major challenge in mining these association rules is pattern explosion. To limit the output, we aim to eliminate all redundant rules. We define the class of closed association rules, and show that this class contains all non-redundant output. To make the algorithm efficient, we use further pruning steps along the way. First of all, we generate only free and closed frequent episodes from which we create candidate rules, we speed up the evaluation of the rules, and then prune the remaining non-closed rules from the output. Finally, we provide the user with the additional option of using a confidence boost threshold to remove the less informative rules from the output.

1 Introduction

Discovering frequent patterns in an event sequence is an important field in data mining. Episodes, first defined by Mannila et al. [16], represent a rich class of sequential patterns, enabling us to discover events occurring in the vicinity of each other while at the same time capturing complex interactions between the events.

More specifically, a frequent episode is traditionally considered to be a set of events that reoccurs in the sequence. Gaps are allowed between the events and the order in which the events are allowed to occur is specified by the episode. The frequency of an episode is usually expressed as the number of windows of specified length in which the episode occurs, but can also be defined incorporating other concepts, such as minimal windows that contain the episode. However, it is important that the frequency is monotonically decreasing so we can use the well-known level-wise approach to mine all frequent episodes.

The order restrictions of an episode are described by a directed acyclic graph (DAG): the set of events in a sequence covers the episode if and only if each event occurs only after all its parent events (with respect to the DAG) have occurred (see the formal definition in Section 3). Usually, only two extreme cases are considered. A parallel episode poses no restrictions on the order of the events, and a window covers the episode if the events occur in the window, in any order. In such a case, the DAG associated with the episode contains no edges. The other extreme case is a serial episode. Such an episode requires that the events occur in one, and only one, specific order in the sequence. Clearly, serial episodes are more restrictive than parallel episodes. If a serial episode is frequent, then its parallel version is also frequent. Examples of the three main types of episodes are given in Figure 1. To avoid drawing DAGs in trivial examples, we will use $\{a_1, \ldots, a_n\}$ to denote a parallel episode consisting of nodes $a_1, \ldots, a_n$, and $a_1 \to \cdots \to a_n$ to denote a serial episode consisting of nodes $a_1, \ldots, a_n$ and edges from node a_i to node a_j if $i < j$.



(a) parallel (b) general (c) serial

Figure 1: Episodes.

*A preliminary version appeared as "MARBLES: Mining Association Rules Buried in Long Event Sequences", in Proceedings of the SIAM International Conference on Data Mining (SDM), 2012 [10].

An association rule between two episodes expresses the fact that the occurrence of one episode implies, with a high enough probability, that another episode can be found nearby. Typically, association rules are defined such that an occurrence of a smaller episode implies the occurrence of a greater episode.

The main contribution of this paper is an algorithm that mines association rules consisting of two episodes represented by DAGs. On top of that, we introduce two novel ways to define the confidence of an association rule, based on more intuitive concepts than the traditional method using sliding windows of fixed length. To reduce the size of the output, we adopt the traditional concept of closed patterns to obtain only non-redundant association rules. We present a collection of algorithms, called MARBLES, to mine all such rules, handling all three possible approaches.

So far, very little research has gone into the search for episodes based on DAGs and even less has gone into the discovery of association rules between them. In practice, such episodes have been overshadowed by parallel and serial episodes. The main reason for this is the pattern explosion. Consider the case of itemsets, where a frequent itemset of size k has 2^k frequent subsets. With an episode of size k, described by a DAG, the number of possible subepisodes is much larger, as we also have to look at all possible subsets of its edges, the number of which grows quadratically with the number of nodes. On top of that, each association rule consists of two episodes, so the number of combinations is huge, as illustrated by the following example.

EXAMPLE 1. *Consider a sequence within which subsequence abcd occurs frequently, using a fixed window of size 4. Assume that, outside these occurrences, events a, b, c and d never occur. It is easy to see that episodes* $G = \{a, d\}$ *and* $H = a \to b \to c \to d$ *will have the same frequency. The same, of course, is also true for any episode* X*, such that* $G \subseteq X \subseteq H$*. Therefore, the confidence of all association rules* $X \Rightarrow Y$*, where* $G \subseteq X \subset Y \subseteq H$*, will be equal to 1. In this case, a single simple reoccurring pattern in the sequence results in 158 different association rules*[1]*. However, just one of those rules is actually not redundant, namely* $G \Rightarrow H$*, as all others can be derived from that one.*

However, the advantage of episodes based on DAGs is that they allow us to capture dependencies between the events while not being too restrictive. The following example illustrates that parallel and serial episodes may be insufficient as a means of discovering all interesting association rules in a dataset.

[1]we counted this number manually

EXAMPLE 2. *As an example we will use text data, namely inaugural speeches by presidents of the United States (see Section 7 for more details). Protocol requires the presidents to address the chief justice and the vice presidents in their speeches. Hence, we have discovered association rule* $\{chief, justic, vice, president\} \Rightarrow \{chief \to justic, vice \to president\}$*. This rule tells us that when these four words appear near each other, then 'chief' precedes 'justice' and 'vice' precedes 'president'. Since the actual address order varies from speech to speech, the pattern does not impose any additional restrictions. The discovered rule informs us of the frequent usage of phrases 'chief justice' and 'vice president' in each others vicinity, something we could never discover using only parallel or serial episodes.*

A popular method of reducing the size of the output in any pattern mining problem is to discover only closed patterns. A pattern is closed if there exists no superpattern with the same frequency. Some work has gone into the discovery of closed frequent episodes, but we go a step further, by introducing the concept of closed association rules. We output only the rules where the left-hand side is minimal, and the right-hand side maximal. Defining this class of association rules is not trivial, and we use a variety of computational tricks to speed up the execution of our algorithm. This allows us to output association rules consisting of two episodes represented by DAGs, and yet keep the size of the output well under control.

Once the redundant rules have been removed in this way, some of the remaining rules can still be less informative than others. We adopt the concept of confidence boost [3], a measure of how informative a rule is, to our setting, and allow the user to remove the less informative rules from the output by means of a confidence boost threshold. We experimentally confirm that the output can be singnificantly reduced in this way.

Apart from the traditional definition of the confidence of an association rule, based on the frequencies of the two episodes using a sliding window of fixed size, we introduce two additional ways to define the confidence, using either minimal windows or weighted minimal windows. We show that all three methods have their merits, and can be valuable and intuitive, depending on the nature of the input sequence and the wishes of the end user.

The rest of the paper is organised as follows: In Section 2, we discuss the most relevant related work, before presenting the main notations and concepts in Section 3. Section 4 introduces the notion of association rules using three different methods, while in Section 5 we discuss how we can limit the size of the output by eliminating redundant association rules. The algorithms that allow us to achieve this goal are presented in detail in Section 6. In Section 7 we show the results of our experiments, before presenting our conclusions in Section 8. Our implementation of the algorithm is available online[2].

2 Related Work

The first attempt at discovering frequent subsequences, or serial episodes, was made by Wang et al. [24]. The dataset consisted of a number of sequences, and a pattern was considered interesting if it was long enough and could be found in a sufficient number of sequences. A complete solution to a more general problem was later provided by Agrawal and Srikant [2] using an Apriori-style algorithm [1].

Looking for frequent general episodes in a single event sequence was first proposed by Mannila et al. [16]. The Winepi algorithm finds all episodes that occur in a sufficient number of windows of fixed length, and generates association rules $X \Rightarrow Y$, where $X \subset Y$ and both X and Y are frequent episodes. Specific algorithms were given for the case of parallel and serial episodes, but no algorithm for detecting general episodes was provided. Tatti and Cule [20] extend the definition of an episode to be able to depict simultaneous events. They provide an algorithm for generating all frequent episodes, but not association rules.

Mannila et al. also propose Minepi [16], an alternative interestingness measure for an episode, where the frequency is defined as the number of minimal windows that contain the episode. In this context, the authors also define association rules. Unfortunately, this frequency measure is not monotonically decreasing. However, the issue can be fixed by defining frequency as the maximal number of non-overlapping minimal windows [15, 19]. Zhou et al. [26] proposed mining closed serial episodes based on the Minepi method. However, the paper did not address the non-monotonicity issue of Minepi. None of these follow-up papers handled the problem of association rules.

Méger and Rigotti [17] propose a method for mining association rules of the form $X \Rightarrow Y$, such that X and Y are both serial episodes, and X is a prefix of Y. Cule et al. [9] introduce an alternative interestingness measure for episodes, combining frequency with the cohesion of an episode. They further extend this work to mine association rules [8], but the method works only for parallel episodes.

A lot of research in the field of pattern discovery has gone into eliminating redundancy. An important way to tackle this problem is by outputting only closed patterns. Within sequence mining, some research has gone into outputting only closed subsequences, where a sequence is considered closed if it is not properly contained in any other sequence which has the same frequency. Yan et al. [25], Tzvetkov et al. [22], and Wang and Han [23] proposed methods for mining such closed patterns, while Garriga [7] further reduced the output by post-processing it and representing the patterns using partial orders. Harms et al. [14], meanwhile, experiment with closed serial episodes. In another attempt to trim the output, Garofalakis et al. [11] proposed a family of algorithms called Spirit which allow the user to define regular expressions that specify the language that the discovered patterns must belong to.

Pei et al. [18], and Tatti and Cule [21] considered restricted versions of the general problem setup of finding frequent episodes. The former approach assumes a dataset of sequences where the same label can occur only once. Hence, an episode can contain only unique labels. The latter pointed out the problem of defining a proper subset relationship between general episodes and tackled it by considering only strict episodes, where two nodes having the same label had to be connected by a path. In our work, we adopt the latter approach, extend it by allowing events in the sequence to take place at the same time, and build on it further in order to mine association rules.

Further interestingness measures for episodes, either statistically motivated or aimed at removing bias towards smaller episodes, were made by Garriga [6], Gwadera et al. [12, 13], Calders et al. [5], and Tatti [19]. All these methods, however, were limited to finding interesting episodes, and stopped short of discovering association rules between them.

Tackling redundancy within association rules, and not only within the patterns they consist of, has been done within the field of frequent itemset mining, but not within episode mining. Bastide et al. [4] define rules as non-redundant if they consist of a minimal antecedent and a maximal consequent, while Balcazar [3] introduces the concept of confidence boost, a measure of how informative a rule is. We incorporate both these concepts in our approach, while dealing with the complexity of terms such as minimal and maximal in the context of episodes.

3 Preliminaries

In this section, we introduce the basic concepts that we will use throughout the paper. First we will describe our dataset.

Definition 1. *We define a* sequence event $e =$

[2]http://adrem.ua.ac.be/implementations

$(id(e), lab(e), ts(e))$ *as a tuple consisting of three entries, a unique id number* $id(e)$*, a label* $lab(e)$ *coming from an alphabet* Σ*, and a time stamp integer* $ts(e)$*. We will assume that if* $id(e) > id(f)$*, then* $ts(e) \geq ts(f)$*. A* sequence *is a collection of sequence events ordered by their ids.*

Note that we are allowing multiple events to have the same time stamp even when their labels are equivalent. For the sake of simplicity, we will use the notation $s_1 \cdots s_N$ to mean a sequence $((1, s_1, 1), \ldots, (N, s_N, N))$.

Definition 2. *Given a sequence* s *and two integers* i *and* j *we define a* subsequence $s[i, j] = s_i, \ldots, s_j$ *containing all events occurring between* i *and* j*. We define the* length *of subsequence* $s[i, j]$ *as*

$$len(s[i, j]) = ts(s_j) - ts(s_i) + 1.$$

Our next step is to define the patterns we are interested in.

Definition 3. *An* episode G *is represented by a directed acyclic graph with labelled nodes, that is,* $G = (V, E, lab)$*, where* $V = (v_1, \ldots, v_K)$ *is the set of nodes,* E *is the set of directed edges, and* lab *is the function* $lab : V \to \Sigma$*, mapping each node* v_i *to its label.*

When there is no danger of confusion, we will use the same letter to denote an episode and its graph.

Definition 4. *A node* n *in an episode graph is a* descendant *of a node* m *if there is a path from* m *to* n*. In that case, node* m *is an* ancestor *of node* n*.*

We are now ready to give a precise definition of an occurrence of a pattern in a sequence.

Definition 5. *Given a sequence* s *and an episode* G *we say that* s covers G*, or* G occurs *in* s*, if there is an* injective *map* f *mapping each node* v_i *to a valid index such that the node* v_i *in* G *and the corresponding sequence element* $s_{f(v_i)}$ *have the same label,* $s_{f(v_i)} = lab(v_i)$*, and that if there is an edge* (v_i, v_j) *in* G*, then we must have* $f(v_i) < f(v_j)$*. In other words, the parents of* v_j *must occur in* s *before* v_j*. If the mapping* f *is surjective, that is, all events in* s *are used, we will say that* s *is an* instance *of* G*.*

An example of a sequence covering an episode is given in Figure 2.

In order to be able to discover association rules, we must first be able to compare episodes using some sort of a subset relationship.

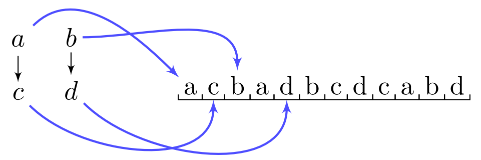


Figure 2: An example of an episode covered by a sequence.

Definition 6. *Given two episodes* G *and* H*, we say that* G *is a* subepisode *of* H*, denoted* $G \subseteq H$*, if the DAG describing episode* G *is a subgraph of the DAG describing episode* H*.*

Definition 7. *Given two episodes* G *and* H*, such that* $G \subset H$*, we can express an* association rule $G \Rightarrow H$*. We call* G *the* head *of the rule, and* H *the* tail *of the rule.*

4 Association Rules

In this section we present three possible methods to measure the frequency of an episode and the confidence of an association rule. The first one uses a sliding window of fixed size, the second is based on disjoint minimal windows, while the third one introduces the concept of weighted minimal windows.

4.1 Using Fixed Windows We start off by defining the frequency of an episode in the traditional manner, based on windows of fixed length. This definition corresponds to the definition used in Winepi [16]. The frequency is monotonically decreasing which allows us to do effective pruning while discovering frequent episodes.

Definition 8. *Given a window size* ρ *and a sequence* s*, we define the* fixed-window frequency *of an episode* G *in* s*, denoted* $fr_f(G; s)$*, to be the number of windows of size* ρ *in* s *covering the episode,*

$$fr_f(G; s) = |\{s[i, i + \rho - 1] \mid s[i, i + \rho - 1] \text{ covers } G\}|.$$

We will use $fr_f(G)$ *whenever* s *is clear from the context. An episode is* σ-frequent *(or simply* frequent*) if its frequency is higher than or equal to some given threshold* σ*.*

Note that, as we slide a window across the sequence, the first sliding window will contain just the first element of the sequence, and the last window only its last element. Thus, given a window length w and a sequence length n, there will be $w + n - 1$ windows.

In this context, association rules can be defined in the traditional manner.

Definition 9. *Given a window size ρ and episodes X and Y, such that $X \subset Y$, we define the* fixed-window confidence *of the association rule $X \Rightarrow Y$, denoted $c_f(X \Rightarrow Y)$, to be the ratio of their respective frequencies,*

$$c_f(X \Rightarrow Y) = \frac{fr_f(Y)}{fr_f(X)}.$$

Informally, we can interpret this definition as follows: $c_f(X \Rightarrow Y)$ is the percentage of windows that contain X that also contain Y. In other words, if we encounter a window that contains X, $c_f(X \Rightarrow Y)$ represents the probability that the window also contains Y.

4.2 Using Minimal Windows

Using a sliding window of fixed length has some drawbacks, particularly in the context of association rules, as can be seen in the following examples.

Example 3. *Consider a sequence within which subsequence abcd occurs frequently, using a fixed window of size 4. Assume that there are always at least three other events between each two occurrences, and that outside these occurrences events a, b, c and d never occur. The fact that $fr_f(\{a, d\}) = fr_f(a \to b \to c \to d)$ implies that $c_f(\{a, d\} \Rightarrow a \to b \to c \to d) = 1$. However, $c_f(\{b, c\} \Rightarrow a \to b \to c \to d) = \frac{1}{3}$. Intuitively, though, looking at the pattern, we note that every occurrence of $\{b, c\}$, just like every occurrence of $\{a, d\}$, implies an occurrence of $a \to b \to c \to d$. The confidences of the two rules do not reflect that.*

Example 4. *Consider sequence s_1 within which subsequence abxycd occurs frequently, and sequence s_2 within which subsequence axbcyd occurs frequently, where x and y are noise events and are not part of the patterns. Assume we are using a fixed window of size 6, and that, outside these occurrences, events a, b, c and d never occur. As in the previous example, $c_f(\{a, d\} \Rightarrow a \to b \to c \to d) = 1$, but $c_f(\{b, c\} \Rightarrow a \to b \to c \to d)$ is now equal to $\frac{1}{3}$ in s_1 and $\frac{1}{5}$ in s_2. Once again, the confidence values are not intuitive.*

The problem with basing the definition of confidence of an association rule on the fixed-window frequencies of the two episodes is that this approach focuses on the occurrences of the episodes in the windows, rather than on their occurrences in the sequence. In the above examples, while there are some windows that contain $\{b, c\}$ and not $a \to b \to c \to d$, it still holds that *every* occurrence of $\{b, c\}$ *in the sequence* can be found within an occurrence of $a \to b \to c \to d$. Our first step towards addressing this issue is to consider a different definition of the frequency of an episode in a sequence.

Definition 10. *Given a sequence s and an episode G, a window $s[a, b]$ is called a* minimal window *of G in s, if $len(s[a, b]) \leq \rho$, $s[a, b]$ covers G, and if no proper subwindow of $s[a, b]$ covers G. We define $b(s[a, b]) = ts(s_a)$ as the beginning of the window, and $e(s[a, b]) = ts(s_b)$ as its end. We denote the set of all minimal windows of G in s with $mw(G; s)$, or simply $mw(G)$, when s is known from the context. Given a set of minimal windows W, we define a function $dis(W)$ to be equal to 1 if all windows in W are pairwise disjoint, and 0 otherwise.*

An attempt was made to define the frequency of an episode as the number of its minimal windows [16]. However, this measure proved to be non-monotonic. In order to satisfy the downward-closed property, we need to consider only the non-overlapping windows [15, 19].

Definition 11. *The* disjoint-window frequency *of an episode G in a sequence s, denoted $fr_m(G)$, is defined as the maximal number of non-overlapping minimal windows within s that contain episode G. Formally,*

$$fr_m(G) = \max \{|W| \mid W \subseteq mw(G), dis(W) = 1\}.$$

A naïve way to define the confidence of an association rule $X \Rightarrow Y$ in the disjoint-window context would be to, once again, compute the ratio between the frequencies of Y and X. The following example shows that this might not be very intuitive either.

Example 5. *Consider sequence s_1 within which subsequence abcxyd occurs 100 times, and sequence s_2 within which subsequence abcbcd occurs 100 times, where x and y are noise events and are not part of the patterns. Assume that, outside these occurrences, events a, b, c and d never occur. Denote $G = \{b, c\}$ and $H = a \to b \to c \to d$. In s_1, $fr_m(G) = 100$ and $fr_m(H) = 100$. Therefore, $\frac{fr(G)}{fr(H)} = 1$. However, in s_2, $fr_m(G) = 200$ and $fr_m(H) = 100$, and $\frac{fr(G)}{fr(H)} = 0.5$. Clearly, in both sequences, each occurrence of G implies an occurrence of H, and the results for s_2 are not satisfactory.*

While defining the frequency of an episode using minimal windows solved some of the problems inherent in the fixed-window method, we clearly still need to find a better way to define the confidence of an association rule. Intuitively, we wish the confidence of rule $X \Rightarrow Y$ to express the probability of encountering a minimal window of Y having encountered a minimal window of X. More formally, we wish to know what percentage of minimal windows of X are contained within minimal windows of Y. However, in order to do this, we are forced to drop the constraint that the minimal windows in question must be disjoint. The reasoning behind this decision is shown in the following example.

Example 6. *Consider sequence $s = abcadbcbcd$. Denote $G = b \to c$ and $H = a \to b \to c \to d$. The disjoint-window frequency of H is 1, but the sequence contains two overlapping minimal windows of H. There are three minimal windows of G, and each of them is contained within a minimal window of H. However, if we were to only use non-overlapping windows of H, we would be faced with two problems. First of all, the confidence of rule $G \Rightarrow H$ would depend on our choice of disjoint minimal windows — if we chose the first minimal window of H, $s[1, 5]$, we would find two occurrences of G outside it and the confidence of the rule would be $\frac{1}{3}$, whereas if we chose the second minimal window of H, $s[4, 10]$, we would find just one occurrence of G outside it, and the confidence would be $\frac{2}{3}$. More importantly, whichever choice we made, we would not be able to get the correct result, showing that* every *occurrence of G is contained within an occurrence of H.*

Now that we have seen that we cannot define the confidence of an association rule using either the disjoint-window frequencies, or the containment of the disjoint occurrences, of the two episodes, we are ready to present a definition that corresponds exactly to our intuition.

Definition 12. *Given episodes X and Y, such that $X \subset Y$, and a minimal window $s[a, b]$ of episode X. Assume there exists a minimal window $s[c, d]$ of Y such that $c \leq a$ and $b \leq d$, then we define the* minimal-extensibility *of occurrence $s[a, b]$ of X into an occurrence of Y as*

$$ext_m(s[a, b], X, Y) = 1.$$

If there exists no such minimal window of Y, we define $ext_m(s[a, b], X, Y) = 0$.

Definition 13. *Given episodes X and Y, such that $X \subset Y$, we define the* minimal-window confidence *of the association rule $X \Rightarrow Y$, denoted $c_m(X \Rightarrow Y)$, to be the proportion of minimal windows of X that are contained within a minimal window of Y,*

$$c_m(X \Rightarrow Y) = \frac{1}{|mw(X)|} \sum_{w \in mw(X)} ext_m(w, X, Y).$$

Algorithm MinWinConfidence, given in Algorithm 1, computes the minimal-window confidence of a given association rule. The algorithm takes as input two ordered lists of minimal windows, V for the head episode and W for the tail episode. The algorithm then enumerates windows in V and tries to find a covering window from W. Since W and V are ordered, we do not need to search the covering window from beginning but instead from the last inspected window. This brings the run-time to $O(|V| + |W|)$. Note that the number of minimal windows is bounded by the number of elements in a sequence, and is in practice significantly lower.

Algorithm 1: MinWinConfidence. Computes minimal-window confidence $c_m(X \Rightarrow Y)$.

```
input : list of minimal windows
        $V = \{v_1, \ldots, v_N\}$ of episode $X$, list of
        minimal windows $W = \{w_1, \ldots, w_M\}$ of
        episode $Y$
output: $c_m(X \Rightarrow Y)$
1 $u \leftarrow 0$; $i \leftarrow 1$;
2 foreach $v \in V$ do
3     while $i \leq M$ and $e(w_i) \leq e(v)$ do
4         $i \leftarrow i + 1$;
5     if $b(w_i) \leq b(v)$ then $u \leftarrow u + 1$;
6 return $u/N$;
```

4.3 Weighted Minimal Windows The problem with using minimal windows is that they do not take the cohesion of a pattern into account. A minimal window of size 2 will make the same contribution towards the frequency of an episode as a minimal window of size 10. Similarly, using only minimal windows, the confidence of the association rule $X \Rightarrow Y$ would depend solely on the inclusion of minimal windows of X inside minimal windows of Y, regardless of how much the window would need to be expanded in order to find the whole of Y. The following example illustrates these problems further.

Example 7. *Consider sequences $s_1 = axbcyd$ and $s_2 = abxycd$. Episode $b \to c$ occurs once in each sequence, so its disjoint-window frequency would be equal to 1 in each sequence. However, the implied pattern* an occurrence of a b is followed by an occurence of a c *is clearly stronger in s_1, where b is immediately followed by a c. In both sequences, the minimal window of $b \to c$ is contained within a minimal window of $a \to b \to c \to d$, and the confidence of the rule $b \to c \Rightarrow a \to b \to c \to d$ using minimal windows would therefore be equal to 1 for both sequences. However, the meaning of the rule,* an occurrence of episode $b \to c$ is likely to be contained within a nearby occurrence of *$a \to b \to c \to d$, can be seen more clearly in s_2. In s_1, we must look much further before we encounter the whole of the larger episode.*

To address these problems, we must first modify the definition of the frequency of an episode in a sequence.

DEFINITION 14. *The* total weight *of a set of windows W in a sequence s, denoted $tw(W)$, is defined as*

$$tw(W) = \sum_{w \in W} \frac{1}{len(w)}.$$

The weighted-window frequency *of an episode G in a sequence s, denoted $fr_w(G)$, is defined as*

$$fr_w(G) = \max \{tw(W) \mid W \subseteq mw(G), dis(W) = 1\}.$$

Informally, the shorter the window, the greater its contribution towards the overall frequency of the episode. Note that, once again, we need to consider only non-overlapping windows in order to satisfy the downward-closed property. However, this time, as the following example illustrates, we need to be much more careful as to which windows we choose, hence the need to maximise the sum of the windows' inverse lengths, rather than simply their number.

EXAMPLE 8. *Consider sequence $s = abxywzcbaxywzc$ and parallel episode $X = \{a, b, c\}$. Sequence s contains three minimal windows of this episode, namely $s[1, 7]$, $s[7, 9]$ and $s[8, 14]$. Note that the second minimal window overlaps with both the first and the third, so the disjoint-window frequency of X would be equal to 2. The two windows that would count towards this frequency would be $s[1, 7]$ and $s[8, 14]$. However, if we want to maximise the sum of the inverse lengths of the windows, we cannot simply take the maximal number of non-overlapping windows and then compute this sum. In our example, we have two possible sets of non-overlapping windows, $s[1, 7]$ and $s[8, 14]$, and $s[7, 9]$ alone. In the first case, we have two windows of size 7, and the resulting sum would be $\frac{2}{7}$. In the second case, we have one window of size 3, and the sum would, therefore, be equal to $\frac{1}{3}$. We see that by choosing a smaller number of non-overlapping windows, we actually get a higher weighted-window frequency for the episode. We conclude that $fr_w(X) = \frac{1}{3}$.*

In order to compute $fr_w(X)$ we use a dynamic program given in Algorithm 2. The algorithm takes as input a list of minimal windows for an episode G and computes $fr_w(G)$.

Given an ordered list of minimal windows $V = \{v_1, \ldots, v_N\}$, we need to select a subset of V, containing disjoint windows that maximise the total weight. In order to do that, let c_i be the maximal weight of a subset of disjoint windows of $\{v_i, \ldots, v_N\}$. Then it is easy to see that

$$c_i = \max(1/len(v_i) + c_d, c_{i+1}),$$

Algorithm 2: WEIGHTFREQUENCE. Computes weighted frequency $fr_w(X)$.

```
   input : list of minimal windows
           $V = \{v_1, \ldots, v_N\}$ of episode $X$
   output: $fr_m(X)$
1  $j \leftarrow 1$;
2  foreach $v_i \in V$ do
3      while $j \leq N$ and $e(v_i) \geq b(v_j)$ do
4          $j \leftarrow j + 1$;
5      $d_i \leftarrow j$;
6  $c_{N+1} \leftarrow 0$;
7  for $i = N \ldots 1$ do
8      $c_i \leftarrow \max(1/len(v_i) + c_{d_i}, c_{i+1})$;
9  return $c_1$;
```

where d is the index of the next minimal window v_d that is disjoint with v_i. The left side in the *max* corresponds to using v_i in the subset and the right side corresponds to omitting v_i from the so far best disjoint collection.

The algorithm first finds d_i, the index of the next disjoint window for each v_i and then constructs the weights c_i. Both steps require $O(N)$ time and memory.

As was already shown in Example 7, we also need to take the various lengths of the minimal windows into account when computing the confidence of association rules. For reasons similar to those discussed in Section 4.2, we once again cannot take only the disjoint minimal windows into account. Intuitively, we want the confidence of rule $X \Rightarrow Y$ to correspond to the likelihood of encountering the whole of Y once we have encountered X. The nearer the whole of Y is to X on average, the higher the confidence should be.

DEFINITION 15. *Given episodes X and Y, such that $X \subset Y$, and a minimal window $s[a, b]$ of episode X. Assume there exists a minimal window $s[c, d]$ of Y such that $c \leq a$ and $b \leq d$, and that this is the smallest window that contains both Y and $s[a, b]$, then we define the* weighted-extensibility *of occurrence $s[a, b]$ of X into an occurrence of Y as*

$$ext_w(s[a, b], X, Y) = \frac{len(s[a, b])}{len(s[c, d])}.$$

If there exists no such minimal window of Y, we define $ext(s[a, b], X, Y) = 0$.

DEFINITION 16. *Given episodes X and Y, such that $X \subset Y$, we define the* weighted-window confidence *of the association rule $X \Rightarrow Y$, denoted $c_w(X \Rightarrow Y)$, to be the average weighted-extensibility of an occurrence of X*

into an occurrence of Y,

$$c_w(X \Rightarrow Y) = \frac{1}{|mw(X)|} \sum_{w \in mw(X)} ext_w(w, X, Y).$$

The following example illustrates that, when extending an occurrence of X in order to find an occurrence of Y, it is important that we find the smallest such occurrence.

EXAMPLE 9. *Consider sequence* $s = axybca$ *and episodes* $X = b \to c$ *and* $Y = \{a, b \to c\}$. *Sequence* s *contains one minimal window of* X, *namely* $s[4,5]$. *This occurrence of* X *can be extended in search of an occurrence of* Y. *There are two candidate minimal windows of* Y *that can be considered,* $s[1,5]$ *and* $s[4,6]$. *If we choose the former, the extensibility of* $s[4,5]$ *into* Y *would equal* $\frac{2}{5}$, *and if we use the latter, this value would rise up to* $\frac{2}{3}$. *Since we are interested in how far we need to look in order to extend an occurrence of* X *into an occurrence of* Y, *we clearly need to look for the smallest minimal window of* Y *that satisfies the conditions.*

We now show what effect the new definitions would have on the patterns discussed in Example 7.

EXAMPLE 10. *Consider sequences* $s_1 = axbcyd$ *and* $s_2 = abxycd$, *and episodes* $X = b \to c$ *and* $Y = a \to b \to c \to d$. *First of all, we see that* $fr_w(X; s_1) = \frac{1}{2}$, *while* $fr_w(X; s_2) = \frac{1}{4}$. *However, we also see that* $c_w(X \Rightarrow Y)$ *equals* $\frac{1}{3}$ *in* s_1 *and* $\frac{2}{3}$ *in* s_2. *This shows that both problems mentioned in Example 7 have been successfully addressed.*

Table 1 shows an overview of the results of applying the three methods presented in Sections 4.1, 4.2 and 4.3 on the patterns and sequences discussed in Examples 7 and 10.

We conclude this section by presenting an algorithm, given in Algorithm 3, that computes the weighted-window confidence of a given association rule.

The WEIGHTCONFIDENCE algorithm is similar to the MINWINCONFIDENCE algorithm discussed earlier. It also takes two ordered lists of minimal windows as input, V for the head episode and W for the tail episode. The only difference is that this time we do not need to simply find any covering window, but the smallest such window. This costs us an extra for-loop which brings the run-time to $O(|V| + |W| + |V|C)$, where C is the average number of windows in W covering a window $v \in V$. In practice, this number is small, making the algorithm efficient.

Pattern	s_1	s_2
$fr_f(X)$	5	3
$fr_f(Y)$	1	1
$c_f(X \Rightarrow Y)$	0.2	0.33
$fr_m(X)$	1	1
$fr_m(Y)$	1	1
$c_m(X \Rightarrow Y)$	1	1
$fr_w(X)$	0.5	0.25
$fr_w(Y)$	0.17	0.17
$c_w(X \Rightarrow Y)$	0.33	0.67

Table 1: An overview of all presented methods applied to the sequences given in Example 7. The window size used for the fixed-window method was 6.

Algorithm 3: WEIGHTCONFIDENCE. Computes weighted-window confidence $c_w(X \Rightarrow Y)$.

```
input : list of minimal windows
        $V = \{v_1, \ldots, v_N\}$ of episode $X$, list of
        minimal windows $W = \{w_1, \ldots, w_M\}$ of
        episode $Y$
output: $c_w(X \Rightarrow Y)$
1  $c \leftarrow 0$; $i \leftarrow 1$;
2  foreach $v \in V$ do
3      while $i \leq M$ and $e(w_i) \leq e(v)$ do
4          $i \leftarrow i + 1$;
5      $j \leftarrow i$;
6      $l \leftarrow \infty$;
7      while $j \leq M$ and $b(w_j) \leq b(v)$ do
8          if $len(w) \leq l$ then
9              $l \leftarrow len(w)$;
10             $i \leftarrow j$;
11         $j \leftarrow j + 1$;
12     $c \leftarrow c + len(w)/l$;
13 return $c/N$;
```

5 Eliminating Redundancy

In this section, we will denote the frequency of an episode G with $fr(G)$, and the confidence of a rule $X \Rightarrow Y$ with $c(X \Rightarrow Y)$, regardless of which method we are using, as what follows applies to all three methods.

5.1 Closed Association Rules

It is a known fact in episode mining that two different DAGs may actually represent the same episode. To tackle this redundancy, we must impose some restrictions on the types of DAGs we will discover.

DEFINITION 17. *An episode* G *is called* transitively closed *if for two nodes* n *and* m, *such that there exists a path from* n *to* m, *there also exists an edge from* n *to* m.

Definition 18. *An episode G is called* strict *if for any two nodes v and w in G sharing the same label, there exists a path either from v to w or from w to v.*

As has been shown by Tatti and Cule [21], the issue of the above mentioned redundancy is resolved within the class of transitively closed strict episodes. Therefore, we will mine only association rules consisting of strict, transitively closed episodes. In the remaining text, we consider episodes to be strict and transitively closed, unless stated otherwise.

A traditional step towards further reducing the output is to generate only closed episodes.

Definition 19. *An episode G is* closed *if there exists no episode H, such that $G \subset H$ and $fr(G) = fr(H)$.*

However, we want to eliminate redundancy within the association rules we output, and not only among the frequent episodes. In fact, what we want to achieve is to output only the rules that give us most information. More precisely, we want the left-hand side to be minimal (the most general episode) and the right-hand side maximal (the most specific episode), among all those for which the rule holds. To achieve this, we first need to define what exactly we mean by minimal in this context.

Definition 20. *An episode G is* free *if there exists no episode H, such that $H \subset G$ and $fr(G) = fr(H)$.*

If we wish to fully eliminate redundancy in the output, we must consider only those rules that consist of a free episode on the left-hand side, and a closed episode on the right-hand side.

Definition 21. *Given two association rules $R_1 = X_1 \Rightarrow Y_1$ and $R_2 = X_2 \Rightarrow Y_2$, we say that R_1 is a* subset *of R_2 if $X_2 \subseteq X_1$ and $Y_1 \subseteq Y_2$. In this case, we denote $R_1 \subseteq R_2$. If $X_2 \subset X_1$ or $Y_1 \subset Y_2$, we denote $R_1 \subset R_2$.*

Unlike the frequency of an episode, the confidence of an association rule is not necessarily a monotonotic measure, given this subset relationship between rules. The following example illustrates that, given rules R_1 and R_2, such that $R_1 \subset R_2$, the confidence of R_1 could be smaller than, greater than, or equal to the confidence of R_2.

Example 11. *Consider sequence $s = aacbabda$, using the minimal window method. Consider episodes $G_1 = a$, $G_2 = a \rightarrow b$, $G_3 = a \rightarrow c \rightarrow b$ and $G_4 = a \rightarrow c \rightarrow b \rightarrow d$, and rules $R_1 = G_1 \Rightarrow G_4$, $R_2 = G_2 \Rightarrow G_4$ and $R_3 = G_2 \Rightarrow G_3$. Clearly, it holds that $G_1 \subset G_2 \subset G_3 \subset G_4$, and therefore $R_3 \subset R_2 \subset R_1$. To compute the minimal-window confidence of the three rules, we first need to identify all minimal occurrences of the four episodes in s. There are four minimal occurrences of G_1, $s[1,1]$, $s[2,2]$, $s[5,5]$ and $s[8,8]$, two minimal occurrences of G_2, $s[2,4]$ and $s[5,6]$, one minimal occurrence of G_3, $s[2,4]$, and one minimal occurrence of G_4, $s[2,7]$. For R_1, we see that two of the four minimal occurrences of G_1 can be found within a minimal occurrence of G_4, and therefore $c_m(R_1) = 0.5$. For R_2, we find that both minimal occurrences of G_2 can be found within a minimal occurrence of G_4, so $c_m(R_2) = 1$. Finally, for R_3, we see that just one minimal occurrence of G_2 can be found within a minimal occurrence of G_3, and $c_m(R_3) = 0.5$. To sum up, $R_3 \subset R_2 \subset R_1$, but $c_m(R_1) = c_m(R_3) < c_m(R_2)$.*

Example 11 illustrates that we cannot apply the usual definition of closure to association rules. It is possible for two rules, R_1 and R_2, such that $R_1 \subset R_2$, to have the same confidence purely *by coincidence*. However, in such a case, we cannot derive the confidence of R_1 using the confidence of R_2, and we cannot leave R_1 out of the output. We must therefore be a little bit more careful when defining non-redundant rules.

Definition 22. *Given episodes X and Y, such that $X \subset Y$, the association rule $R = X \Rightarrow Y$ is* not closed *if there exists a rule R_1, such that $R \subset R_1$ and $c(R) = c(R_1)$, and there exists no rule R_2, such that $R \subset R_2 \subset R_1$ and $c(R) \neq c(R_2)$. An association rule that does not satisfy these conditions is* closed.

5.2 Confidence Boost

In reality, mining only closed rules might not be enough to sufficiently reduce the output, as some redundant rules will still be discovered, as illustrated by the following example.

Example 12. *Assume we are given an input sequence s, in which subsequence abcde occurs 999 times, and subsequence abxde just once. Using the fixed window method with the window size of 5, the frequency of all subepisodes of $a \rightarrow b \rightarrow c \rightarrow d \rightarrow e$ that include a and e, but do not include c will be 1 000, while the frequency of all subepisodes that include a, c and e will be 999. As a result, the confidence of rule $\{a, e\} \Rightarrow a \rightarrow b \rightarrow c \rightarrow d \rightarrow e$ will be equal to 0.999, while the confidence of rule $\{a, e\} \Rightarrow a \rightarrow b \rightarrow d \rightarrow e$ would be equal to 1. Both rules would be closed, and both would appear in the output. However, it could be argued that the second rule is not very informative, given the first rule, as it, in fact, covers just one extra occurrence in the dataset.*

To solve this type of redundancy, we turn to the concept of *confidence boost*, as proposed by Balcazar [3].

Definition 23. *Given episodes X and Y, such that $X \subset Y$, the* confidence boost *of association rule $X \Rightarrow Y$*

is defined as

$$cb(X \Rightarrow Y) = \frac{c(X \Rightarrow Y)}{max_{X' \subseteq X, Y \subseteq Y'} c(X' \Rightarrow Y')},$$

under the conditions that $X \Rightarrow Y \neq X' \Rightarrow Y'$ and that Y' is frequent. If the set of rules in the denominator is empty, $cb(X \Rightarrow Y) = \infty$.

Given a certain confidence boost threshold β, the naïve approach would be to remove all rules from the output that have a confidence boost lower than β. However, such an approach involves a number of risks, as illustrated by the following example.

Example 13. *Assume, once again, that we are given an input sequence s, in which subsequence abcde occurs 999 times, and subsequence abxde just once. As seen in Example 12, $c(\{a, e\} \Rightarrow a \rightarrow b \rightarrow c \rightarrow d \rightarrow e) = 0.999$ and $c(\{a, e\} \Rightarrow a \rightarrow b \rightarrow d \rightarrow e) = 1$. However, $cb(\{a, e\} \Rightarrow a \rightarrow b \rightarrow d \rightarrow e) = 1/0.999 \approx 1.001$. Given a confidence boost threshold of, say, 1.1, rule $\{a, e\} \Rightarrow a \rightarrow b \rightarrow d \rightarrow e$ would be removed from the output, as it is not very informative. However, if we chose a confidence threshold of 1, we would find neither of the two rules in the output, and important insight into the properties of the dataset would be lost.*

The informativeness of a rule clearly depends on the other rules in the output, and removing one uninformative rule from the output can make another previously uninformative rule become informative. Therefore, rather than looking at the absolute confidence boost of an individual rule, we need to evaluate the confidence boost of a rule relative to the other rules in the output.

Definition 24. *Given episodes X and Y, such that $X \subset Y$, the confidence boost of association rule $X \Rightarrow Y$* with respect to a set of rules $\mathcal{R}$ *is defined as*

$$cb_{\mathcal{R}}(X \Rightarrow Y) = \frac{c(X \Rightarrow Y)}{max_{X' \subseteq X, Y \subseteq Y'} c(X' \Rightarrow Y')},$$

under the conditions that $X \Rightarrow Y \neq X' \Rightarrow Y'$ and $X' \Rightarrow Y' \in \mathcal{R}$. If the set of rules in the denominator is empty, $cb_{\mathcal{R}}(X \Rightarrow Y) = \infty$.

The key, therefore, is to reduce the output to the bare minimum, but no more than that.

Definition 25. *Given a sequence s, a user-chosen window size ρ, a frequency threshold σ, a confidence threshold ϕ, and a confidence boost threshold β, a set of closed association rules $\mathcal{R}$ is* self-sufficient *if the following conditions hold:*

1. *for each $R = X \Rightarrow Y \in \mathcal{R}$, $fr(Y) \geq \sigma$,*
2. *for each $R \in \mathcal{R}$, $c(R) \geq \phi$,*
3. *for each $R \in \mathcal{R}$, $cb_{\mathcal{R}}(R) \geq \beta$,*
4. *for each $R' = X' \Rightarrow Y' \notin \mathcal{R}$, either $fr(Y') < \sigma$, $c(R') < \phi$, or $cb_{\mathcal{R}}(R') < \beta$*

Definition 25 states that an acceptable output set of association rules must be minimal. Either adding or removing a single rule would break at least one of the conditions for the set to be self-sufficient. The following proposition shows that a self-sufficient set of association rules is also unique.

Proposition 5.1. *Given a sequence s, a user-chosen window size ρ, a frequency threshold σ, a confidence threshold ϕ, and a confidence boost threshold β, there exists one and only one self-sufficient set of closed association rules.*

Proof. We will prove a more general statement: Given a set of rules $\mathcal{B}$, there is a unique set of rules $\mathcal{R} \subseteq \mathcal{B}$ such that $cb_{\mathcal{R}}(R) \geq \beta$ for any $R \in \mathcal{R}$ and $cb_{\mathcal{R}}(R) < \beta$ for any $R \in \mathcal{B} \setminus \mathcal{R}$. We call $\mathcal{R}$ to be a self-sufficient set with respect to $\mathcal{B}$.

If we prove this, then the proposition immediately follows if we set $\mathcal{B}$ to be the set of all closed rules that satisfy the thresholds.

We will prove this using induction over the size of $\mathcal{B}$. Obviously, the result holds for $|\mathcal{B}| = 1$.

Assume that the result holds for any base set of size smaller than k and assume that $|\mathcal{B}| = k$. Since the subset relation of rules is a partial order, there must be a rule R in $\mathcal{B}$ for which there is no rule $R' \in \mathcal{B}$ such that $R \subset R'$. This immediately implies that $cb_{\mathcal{U}}(R) = \infty$ for any subset $\mathcal{U} \subseteq \mathcal{B}$. This means that R must be included in a self-sufficient set. Let

$$\mathcal{V} = \{V \in \mathcal{B} \mid V \subset R,\ c(V)/c(R) < \beta\}.$$

Since R must be in a self-sufficient set, none of the rules in $\mathcal{V}$ can be in a self-sufficient set. Let $\mathcal{W} = \mathcal{R} \setminus (\mathcal{V} \cup \{R\})$. By the induction assumption, let $\mathcal{U}$ be the unique self-sufficient set with respect to $\mathcal{W}$ and let $\mathcal{S} = \mathcal{U} \cup \{R\}$. Set $\mathcal{S}$ is a self-sufficient set with respect to $\mathcal{B}$. To see the uniqueness, let $\mathcal{S}'$ be a self-sufficient set w.r.t. $\mathcal{B}$. We already saw that $R \in \mathcal{S}'$ and that $\mathcal{V} \cap \mathcal{S}' = \emptyset$. Let $\mathcal{U}' = \mathcal{S}' \setminus \{R\}$. Then $\mathcal{U}'$ is a self-sufficient set with respect to $\mathcal{W}$ and by the induction assumption $\mathcal{U} = \mathcal{U}'$ which implies that $\mathcal{S} = \mathcal{S}'$. □

Formally, given a sequence s, a user-chosen window size ρ, a frequency threshold σ, a confidence threshold ϕ, and a confidence boost threshold β we wish to find the self-sufficient subset $\mathcal{R}$ of closed association rules of the form $X \Rightarrow Y$, where $X \subset Y$, $fr(Y) \geq \sigma$, $c(X \Rightarrow Y) \geq \phi$, and $cb_{\mathcal{R}}(R) \geq \beta$. However, finding the self-sufficient set of association rules is not necessarily trivial, as illustrated by the following example.

EXAMPLE 14. *Assume that our output $\mathcal{R}$ consists of four confident association rules, R_1, R_2, R_3 and R_4, such that $R_1 \subset R_2 \subset R_3 \subset R_4$. If $c(R_1) = 0.99$, $c(R_2) = 0.9$, $c(R_3) = 0.8$ and $c(R_4) = 0.72$, then $cb_{\mathcal{R}}(R_1) = 0.99/0.9 = 1.1$, $cb_{\mathcal{R}}(R_2) = 0.9/0.8 = 1.125$, $cb_{\mathcal{R}}(R_3) = 0.8/0.72 \approx 1.11$ and $cb_{\mathcal{R}}(R_4) = \infty$, as no superrule of R_4 has been found. Using a confidence boost threshold of 1.2, only R_4 would be informative enough. However, if we removed R_3 from the output to obtain a set of rules $\mathcal{R}_1 = \{R_1, R_2, R_4\}$, the relative confidence boost of R_2 would now be above the threshold — $cb_{\mathcal{R}_1}(R_2) = 0.9/0.72 = 1.25$. By removing R_3 from the output, R_2 has become informative enough and should be kept. By then removing R_1 from $\mathcal{R}_1$, we would obtain a self-sufficient set of rules, $\{R_2, R_4\}$. Alternatively, we may have chosen to first remove R_2 from $\mathcal{R}$, and thus obtain $\mathcal{R}_2 = \{R_1, R_3, R_4\}$. Now, R_1 would be informative enough, as $cb_{\mathcal{R}_2}(R_1) = 0.99/0.8 \approx 1.24$. By then removing R_3 from $\mathcal{R}_2$, we would obtain set $\mathcal{R}_3 = \{R_1, R_4\}$. However, note that this set is not self-sufficient, as condition 4. from definition 25 is not satisfied. There exists a rule $R_2 \notin \mathcal{R}_3$, such that $cb_{\mathcal{R}_3}(R_2) = 0.9/0.72 = 1.25 \geq \beta$.*

The above example shows that the order in which the rules are removed from the output is very important, a fact that must be taken into account when developing an algorithm for discovering the self-sufficient set of rules. Our algorithm, described in detail in Section 6, first generates all closed association rules, and then in a systematic and deterministic manner removes rules until the set becomes self-sufficient.

5.3 Finding the self-sufficient subset

Our final step is to find the self-sufficient subset of rules from a given set of closed association rules. The proof of Proposition 5.1 gives us a simple approach for this: find a maximal rule R and add it to the output, remove R and the all the rules $R' \subset R$ with $c(R')/c(R) < \beta$, and repeat until there are no rules left. The pseudo-code for the algorithm is given in Algorithm 4.

Finding a maximal rule requires $O(|\mathcal{B}|)$ time, hence the algorithm requires $O(|\mathcal{B}|)$ time. In practice, this can be optimized by first constructing a lattice, that is, a directed graph where each node is a single rule and there is an edge from R_1 to R_2 if and only if $R_1 \subsetneq R_2$. This lattice can be constructed by querying all superrules for each rule, as explained in Section 6.2. Once this lattice, say $\mathcal{L}$, is constructed we can use it to compute the self-sufficient set in $O(|V(\mathcal{L})| + |E(\mathcal{L})|)$ time, as shown in Algorithm 5.

Algorithm 4: BOOST. Computes self-sufficient subset.

```
  input : set of rules $\mathcal{B}$, boost threshold $\beta$
  output: self-sufficient subset $\mathcal{R}$ of $\mathcal{B}$
1 $\mathcal{R} \leftarrow \emptyset$;
2 while $\mathcal{B} \neq \emptyset$ do
3     $R \leftarrow$ a maximal rule in $\mathcal{B}$;
4     $\mathcal{R} \leftarrow \mathcal{R} \cup \{R\}$;
5     $\mathcal{V} \leftarrow \{V \in \mathcal{B} \mid V \subsetneq R,\ c(V)/c(R) < \beta\}$;
6     $\mathcal{B} \leftarrow \mathcal{B} \setminus (\mathcal{V} \cup \{R\})$;
7 return $\mathcal{R}$;
```

Algorithm 5: BOOST. Alternative way of computing self-sufficient subset.

```
  input : set of rules $\mathcal{B}$, boost threshold $\beta$
  output: self-sufficient subset $\mathcal{R}$ of $\mathcal{B}$
1 $\mathcal{L} \leftarrow$ lattice constructed from $\mathcal{B}$.
2 foreach $R \in \mathcal{B}$ in reverse topological order do
3     if there is an unmarked rule $V \supsetneq R$ s.t. $c(R)/c(V) < \beta$ then
4         mark $R$;
5 return unmarked rules from $\mathcal{B}$;
```

6 MARBLES

In this section, we present our algorithms. We start off by describing the first step, mining frequent episodes, before moving on to the algorithms we developed for mining closed association rules.

6.1 Mining episodes

In order to generate association rules, we first need to generate frequent episodes. As our goal is to mine only closed association rules we do not need to consider all frequent episodes. We will resort to i-closed episodes [21].

DEFINITION 26. *An i-closure is a function $icl(G; s) = H$ mapping an episode G to an episode H such that $G \subseteq H$. H is constructed from G in two phases. Firstly, if an event with a label l occurs in every minimal window of G and there is no node in G labelled with l, then we add a new node into G with label l. This is repeated until no new additions are possible. In the second phase, we add new edges. If in every instance of G a node v occurs before a node w, then we add an edge from v to w.*

DEFINITION 27. *Given a sequence s, we say that an episode G is i-closed if $G = icl(G; s)$. From the* *fundamental properties of the closure, it follows that G is the maximal episode for which the closure is equal to G. Similarly, we say that G is i-free (or an i-generator) if there is no H such that $H \subset G$ and $icl(G; s) = icl(H; s)$. In other words, G is a minimal episode that can produce $icl(G; s)$.*

Note that there can be several minimal episodes that have the same i-closure but only one maximal episode.

The reason we are using i-closed episodes is that we can use them to generate closed association rules.

PROPOSITION 6.1. *Let $X \Rightarrow Y$ be a closed association rule. Then X is i-free and Y is i-closed.*

Proof. All three confidence measures depend only on the minimal windows of X and Y. As demonstrated in the proof of Theorem 6 in the paper where i-closure was originally proposed [21], if two episodes, say G and H, have the same i-closure, i.e., $icl(G; s) = icl(H; s)$, then they have exactly the same minimal windows. This implies that $c(X \Rightarrow Y) = c(X \Rightarrow icl(Y; s))$ proving that Y must be i-closed. Similarly if we have $X' \subseteq X$ such that $icl(X'; s) = icl(X; s)$, then $c(X \Rightarrow Y) = c(X' \Rightarrow Y)$ which immediately implies that $X = X'$. Consequenly, X must be i-free. □

To mine episodes we employ the MINEEPISODES algorithm given in the original paper [21]. The algorithm follows a standard BFS-approach for closed patterns by discovering first all frequent i-free patterns and computing the closure of each i-free episode. Hence, the output of this algorithm are all frequent i-free episodes and their closures. Typically, free patterns are suppressed from the final output but in this case we need them to generate the left-hand side of the association rules.

MINEEPISODES can either use fixed windows or minimal windows as a frequency constraint. The algorithm computes the frequency by first discovering all minimal windows and then computing the frequency using the discovered windows. We use WEIGHTFREQUENCE to compute whenever we use $fr_w(G)$ as a frequency constraint.

6.2 Mining association rules Now that we have mined episodes, the next step is to build association rules. In order to do that, we introduce MARBLES given in Algorithm 6. The algorithm takes as input episodes mined in the first step and builds rules from these episodes. We assume that episodes are provided in specific groups. The input consists of a list of pairs, where the first element is an i-closed episode, say X, and the second element is a list, say $\mathcal{G}$, of all i-free episodes that have X as their closure. The reason for this grouping is that the confidence is equal for all rules of form $G \Rightarrow Y$, where $G \in \mathcal{G}$. In fact, $c(G \Rightarrow Y) = c(X \Rightarrow Y)$.

Of the three definitions of confidence given in Section 4, the fixed-window confidence is the only one that is monotonic. Confidence based on (weighted) minimal windows is not monotonic, hence we have to resort to an exhaustive enumeration.

Algorithm 6: MARBLES. Mines closed association rules.

```
input : confidence threshold $\phi$, list $\mathcal{X}$ of pairs
        $(X, \mathcal{G})$, where $X$ is an $i$-closed episode
        and $\mathcal{G}$ are the $i$-free episodes having $X$
        as their $i$-closure
output: list of closed association rules
1  $\mathcal{R} \leftarrow \emptyset$;
2  foreach $(X_1, \mathcal{G}_1), (X_2, \mathcal{G}_2) \in \mathcal{X}$ s.t. $X_1 \subseteq X_2$ do
3      foreach $G \in \mathcal{G}_1$ do
4          add $R = G \Rightarrow X_2$ to $\mathcal{R}$;
5          if $c(R) < \phi$ then mark $R$;
6  foreach $R_1 \in \mathcal{R}$ do
7      $\mathcal{S} \leftarrow \emptyset$;
8      foreach $R_2 \in \mathcal{R}, R_1 \subsetneq R_2$ do
9          remove any rule $S$ from $\mathcal{S}$ s.t. $R_2 \subseteq S$;
10         if there is no $S \in \mathcal{S}$ s.t. $S \subseteq R_2$ then
11             add $R_2$ to $\mathcal{S}$;
12     if there is $S \in \mathcal{S}$ s.t. $c(R_1) = c(S)$ then
13         mark $R_1$;
14 return unmarked rules from $\mathcal{R}$;
```

The first loop in the algorithm discovers all confident association rules using i-free and i-closed episodes. This set contains all closed association rules, and possibly some redundant rules. To remove the redundant rules, for each rule R_1, we first construct a set of its minimal superrules $\mathcal{S}$. If any of the rules in $\mathcal{S}$ has the same confidence as R_1, we mark R_1 as non-closed, and consequently purge it from the output.

In order to implement this algorithm efficiently, we need an efficient technique for enumerating all superepisodes of a given episode (used in the first loop) and for enumerating all superrules of a given association rule (used in the second loop). Assume that we have a list of episodes $\mathcal{E}$. We begin by partioning $\mathcal{E}$ into groups

$$\mathcal{E}_L = \{G \in \mathcal{E} \mid G \text{ has } L \text{ as labels}\}.$$

Let $\mathcal{I} = \{\mathcal{E}_L \mid \mathcal{E}_L \neq \emptyset\}$ be the collection of these groups.

We then create subsets of these groups

$$\mathcal{I}_{l,o} = \{\mathcal{E}_L \in \mathcal{I} \mid l \text{ occurs in } L \text{ at least } o \text{ times}\}.$$

Fix $\mathcal{E}_L \in \mathcal{I}$. We can sort the nodes of each episode in $\mathcal{E}_L$ according to their labels. Since we are working only with strict episodes, if the same label occur twice or more, the parent node will go first. Let $v_1, \ldots, v_{|L|}$ be the nodes of episodes in $\mathcal{E}_L$. Let $e = (v_i, v_j)$ be an edge. We define a set of episodes

$$\mathcal{I}_e = \{G \in \mathcal{E}_L \mid G \text{ contains } e\}.$$

Note that $\mathcal{I}_e$ depends on $\mathcal{E}_L$ but we have supressed this from notation.

When searching for superepisodes of a given episode, say G, we first find episode groups $\mathcal{E}_L$ whose labels are a superset of or equal to the labels of G. This is done by taking an intersection of each set $\mathcal{I}_{l,o}$, where l occurs exactly o times in G.

After this is done, we then find a mapping between the labels of G and L (if there are several, we test them all). Once a mapping is found we map the nodes of G to L and find all episodes that contain the edges of G by taking the intersection of sets $\mathcal{I}_e$, where e is an edge of G mapped to L.

The computational complexity of intersecting two sets of size N and M is $O(N + M)$. Hence, computational complexity of performing a query is $O(|\mathcal{E}|K(1 + |E(G)|))$, where K is the total number of mappings of G to label sets L we need to consider. In theory, K can be exponentially large, but in practice it will stay small. In addition, $|\mathcal{I}_{l,o}|$ and $|\mathcal{I}_e|$ will typically be significantly smaller. Hence, in practice we will need significantly less time to perform a query.

To query association rules, we build a similar indexing structure, only now we group rules by their head episode. We index these groups using the head episode and within each group we index each rule with its tail.

7 Experiments

In this section, we present the results of our experiments. We first experimented on a synthetic dataset containing a planted pattern to compare the results of our three methods and to check that they do not produce spurious output. We then applied our methods to a number of various real-life datasets, to see how they perform in various settings applied to different types of datasets.

7.1 Synthetic Dataset To test if our methods recognise a pattern planted in an input sequence, and to check how well the methods reduce the output resulting from such a pattern, we generated a synthetic dataset. The dataset contains 100 000 occurrences of episode G depicted in Figure 3, interleaved with noise. Each occurrence of an a is followed by occurrences of b and c, with a 50% chance of a b coming before c, and a 50% chance of a c coming before b. After b and c have both occurred, there follows a d. There was a 1% chance that a noise event occurs between any two of the four events making up the pattern. Two occurrences of G are separated by occurrences of 10 to 15 noise events (the exact number was randomly chosen for each gap). All noise events were randomly chosen from a group of 100 different noise events. The resulting sequence was 1 651 937 items long, of which 1 251 937 were noise events.

$$\begin{tikzcd} & b \arrow[dr] & \\ a \arrow[ur] \arrow[dr] & & d \\ & c \arrow[ur] & \end{tikzcd}$$

Figure 3: An episode planted into the synthetic dataset.

The fixed-window and the weighted-window methods gave similar results. With a window of size 6, a confidence threshold of 1, and a low enough frequency threshold, the output consisted of 13 closed association rules. Among them were four rules imposing a total order on a set of two events, namely $\{a, b\} \Rightarrow a \to b$, $\{a, c\} \Rightarrow a \to c$, $\{b, d\} \Rightarrow b \to d$ and $\{c, d\} \Rightarrow c \to d$. Six rules imposed a partial order on sets of three events, while two rules imposed a total order among all four events given a partial order, namely $\{a, d, b \to c\} \Rightarrow a \to b \to c \to d$ and $\{a, d, c \to b\} \Rightarrow a \to c \to b \to d$. Finally, one rule stood out, as the left-hand side was smaller than the right-hand side, identifiying which episode implied the complete occurrence of G itself. The rule was $\{a, d\} \Rightarrow G$.

Reducing the window size to 5 or 4 resulted in some occurrences of G not fitting into the window, so the frequencies of the discovered episodes dropped, but the confidence of the rules remained unchanged, as did the output, as long as the frequency threshold was low enough. Due to the nature of the measures, we had to considerably lower the confidence threshold to find further rules. With the fixed-window method, the next discovered rule was $\{b, d\} \Rightarrow \{b, c\} \to d$ with a confidence of around 89% . Three similar rules scored fractionally lower, while rules $\{b\} \Rightarrow \{c\}$ and $\{c\} \Rightarrow \{b\}$ had a confidence of around 83%. Lowering the confidence threshold to 0.6 resulted in 49 closed association rules, none of which could be removed with a 1.1 confidence boost threshold, though 16 could with a 1.15 boost threshold. The same behaviour could be observed with the weighted-window method, though the confidence of the rules was different.

The results using the minimal-window method with a window size of 6 were considerably different. Instead of rule $\{a, d\} \Rightarrow G$, the minimal-window method with a confidence threshold equal to 1 found four rules, namely $\{a\} \Rightarrow G$, $\{b\} \Rightarrow G$, $\{c\} \Rightarrow G$ and $\{d\} \Rightarrow G$. These four rules not only fully described the implications inherent in the planted pattern, but also constituted the complete output, as all other rules with a high enough frequency could be derived from these four. Remarkably, even if we lowered the confidence threshold to 0, these four were the only closed rules we discovered. Lowering the frequency threshold resulted in finding rules in which a total order was imposed on b and c being included in the output, but really interesting results started to come through if we lowered the size of the window. When we lowered the window size to 5, some occurrences of G were no longer discovered, and the four rules above no longer had a confidence equal to 1. With a confidence threshold equal to 1, we now discovered rule $\{a, d\} \Rightarrow G$, and six other rules extending a singleton into a partially ordered set of three events. However, lowering the confidence threshold to 0.99, we again found the four rules mentioned above, increasing the size of the output to 11 (the seven rules with confidence equal to 1 were now closed, and therefore included in the output). However, applying a confidence boost threshold of just 1.01, we could again reduce the output to the original four rules. Again, reducing the confidence threshold all the way to 0 could not provide us with any more informative rules, as these four told us all there was to know about the dataset. Exactly the same output was generated using a window size of 4, a confidence threshold of 0.97 or lower, and a boost threshold of 1.03, as the confidence of the four rules equalled just above 97% with these parametes.

These experiments confirmed that the minimal-window method is more suited for identifying which small patterns can be extended into greater ones, while the fixed-window and the weighted-window methods give more value to rules that impose a stricter order on a pattern of a given size. The minimal-window method provides us with no information as to how far from the small pattern we need to look in order to find the greater pattern (as long as it can be found within the given window size), while the other two methods assign a higher value to rules where this distance is relatively smaller. Clearly, all three methods have their merits, and which one should be used largely depends on the dataset and the wishes of the user.

Finally, it is worth noting that the noise we introduced into the dataset is only an issue in a dense dataset, where the position of an item in the sequence acts as its time stamp. In sparse data, with real-valued time stamps, such noise will have no effect on either the frequency of an episode or the confidence of an association rule. Assume that events a and b occur at time stamps t_1 and t_2, respectively. As long as the distance between t_1 and t_2 is smaller than the window size, those two events will be found within the window, regardless of how many (noise) events occur between them.

7.2 Real-life Datasets We also tested our algorithm on five real-world datasets, three of which were text datasets — *address*, consisting of the inaugural addresses by the presidents of the United States[3], merged to form a single long sequence, *moby*, the novel Moby Dick by Herman Melville[4], and *abstract*, consisting of the first 739 NSF award abstracts from 1990[5], also merged into one long sequence. We processed the sequences using the Porter Stemmer[6] and removed the stop words. Our fourth dataset contained a sequence of alarms triggered in a factory, stretching over 18 months. An entry in the dataset consists of a time stamp and an event type. Finally, the fifth dataset contained information about trains delayed at departure from a Belgian railway station. The dataset consists of actual departure times of delayed trains, coupled with train numbers, stretching over a period of one month. The characteristics of the five datasets are summarised in Table 2. Note that the three text datasets are dense, as there are no time stamps involved. Implicitly, of course, the events (words) are assumed to have "taken place" on consecutive time stamps. The remaining two datasets, *alarms* and trains, are sparse. In both datasets, times are measured in seconds, and most time stamps are not associated with any event. Furthermore, these two datasets also contain events that occur at the same time, which is never the case in the text datasets.

Sequence	Size	$\|\Sigma\|$	type
address	62 066	5 295	dense
moby	105 719	10 277	dense
abstract	67 828	6 718	dense
alarms	514 502	9 595	sparse
trains	10 115	1 280	sparse

Table 2: Characteristics of the five datasets. The first column contains the size of the sequence, the second column the number of unique symbols in the sequence, and the third the type of data.

[3]taken from `http://www.bartleby.com/124/pres68`

[4]taken from `http://www.gutenberg.org/etext/15`

[5]taken from `http://kdd.ics.uci.edu/databases/nsfabs/nsfawards.html`

[6]`http://tartarus.org/~martin/PorterStemmer/`

In all our experiments, we kept the window size and the frequency threshold fixed, while varying the confidence and the confidence boost threshold. The main goal of our experiments was to demonstrate how we tackle the problem of pattern explosion. For each set of experiments, we therefore pushed the frequency threshold as low as possible, until the algorithm for generating all confident association rules began to take too long or ran out of memory. This demonstrated the need to generate i-closed and i-free episodes as an intermediary step, as our algorithm for mining only the closed rules ran much faster, and could handle much lower frequency thresholds. For the text datasets, we used the window size of 15, while for *alarms* and *trains* we used 10 and 30 minutes, respectively, after consulting domain experts in each field. For both of the real-life datasets, an average window contained approximately 10 events, but this number was considerably higher in peak times, and regularly dropped to 0 during the night.

Figure 4 shows how the total number of confident association rules compared with the number of closed rules we discovered in all five datasets, using the fixed-window, minimal-window, and weighted-window method, respectively. The results show that the reduction was significant at all thresholds. Among the text datasets, the best reduction was obtained in the *abstract* dataset, which is not surprising, keeping in mind that this is the most structured dataset of the three. This was particularly visible in the minimal window method, where the reduction was smallest for the *address* and *moby* dataset, and highest for the other three datasets. Due to the low frequency thresholds, we found a large number of rules with confidence equal to 1. Most of these rules were of the form $\{x, y\} \Rightarrow x \rightarrow y$ using all three methods, while the minimal-window method discovered a number of such rules of the form $x \Rightarrow x \rightarrow y$ or $y \Rightarrow x \rightarrow y$.

Figure 5 shows how the output could be further reduced by using a confidence boost threshold. We give results for all five datasets, using the fixed-window, minimal-window, and weighted-window method, respectively. Again, it can clearly be seen that the output can be significantly reduced in more structured datasets, while the results were not impressive for the more diverse *address* and *moby* datasets.

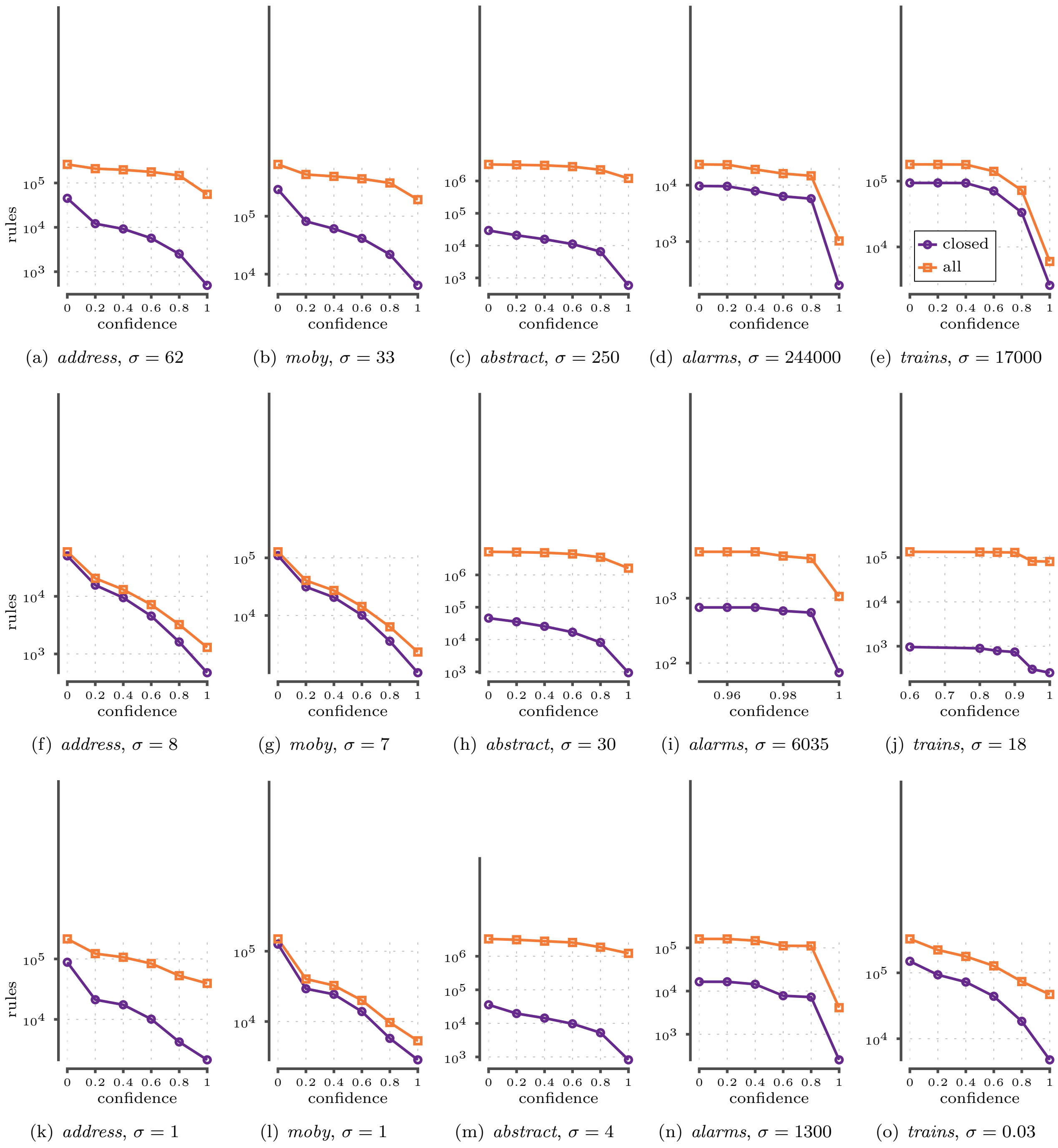
Finally, we give an overview of some of the more interesting association rules discovered in the three text datasets in Figures 6, 7 and 8. Note that our experiments confirmed that the three methods can rank rules differently. For example, Figures 6 (a) and (c) give the top rules of size 2 in the *address* dataset for $\phi = 1$ using the fixed window and the weighted window method, respectively, while Figures 7 (a) and (b) give the top rules of size 2 in the *moby* dataset for $\phi = 1$ using the fixed window and the minimal window method, respectively. Figure 8 (a), on the other hand, shows a rule of size 2 that was ranked top for $\phi = 1$ for all three methods in the *abstract* dataset. Figure 6 (g) shows a rule with $c_f \approx 0.88$ discovered in the *address* dataset. However, due to the rule shown in Figure 6 (h) having $c_f \approx 0.87$, the confidence boost of this rule is actually smaller than 1.01. Using a confidence boost threshold of 1.1, rule shown in Figure 6 (g) would be removed from the output. A similar example from the *moby* dataset is shown in Figures 7 (c) and (d). Finally, a very large rule is shown in Figure 8 (e). Here, the occurrence of a general episode consisting of four words implies, with a confidence equal to 1, the occurrence of a serial episode consisting of six words. Unfortunately, non-disclosure agreements prevent us from presenting similar examples from the *alarms* and *trains* datasets.

8 Conclusions

In this paper, we present a complete association rule miner for strict episodes, a large subclass of general episodes, represented by directed acyclic graphs (DAGs). We approach the problem from three angles, defining the frequency using windows of fixed size, minimal windows, and weighted minimal windows. While the first method is computationally the least demanding, we show that the other two can be more intuitive and meaningful. We define and mine association rules within all three settings.

Furthermore, we tackle the problem of pattern explosion. The pattern explosion is exponential already when it comes to itemsets, it is much more of a problem within the field of episodes, and it culminates when we attempt to generate association rules between general episodes. We define closed association rules, thus eliminating all redundant rules from the output. Our algorithms take advantage of various properties of closed rules, such as the fact that the left-hand side must consist of a free episode, and the right-hand side of a closed episode, which allows us to discover closed association rules efficiently. Our experiments demonstrate that the reduction in the size of the output is considerable. Further reduction can be achieved by setting a confidence boost threshold, thus eliminating the least informative rules.

In future work, we intend to investigate if it is possible to extend this work to the complete class of general episodes, dropping the constraint that they must be strict. An interesting property of the confidence of an association rule is that it is not always a monotonic measure. Further research could also be dedicated to



(a) *address*, $\sigma = 62$ (b) *moby*, $\sigma = 33$ (c) *abstract*, $\sigma = 250$ (d) *alarms*, $\sigma = 244000$ (e) *trains*, $\sigma = 17000$

(f) *address*, $\sigma = 8$ (g) *moby*, $\sigma = 7$ (h) *abstract*, $\sigma = 30$ (i) *alarms*, $\sigma = 6035$ (j) *trains*, $\sigma = 18$

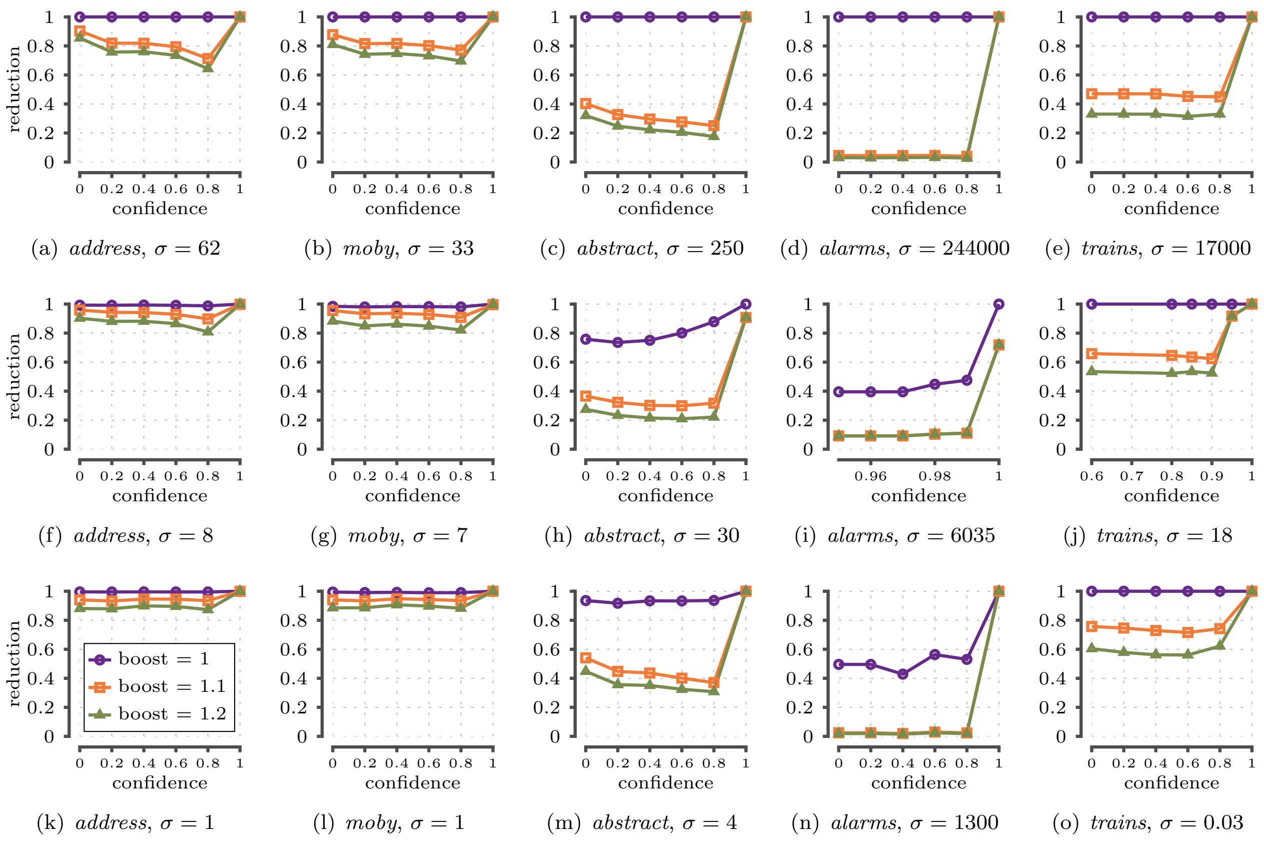
(k) *address*, $\sigma = 1$ (l) *moby*, $\sigma = 1$ (m) *abstract*, $\sigma = 4$ (n) *alarms*, $\sigma = 1300$ (o) *trains*, $\sigma = 0.03$

Figure 4: The comparison of the number of closed association rules and the total number of rules discovered in the five datasets, using the three different methods. (a)-(e) fixed windows, (f)-(j) minimal windows, (k)-(o) weighted windows.

examining if it would still be possible to somehow prune some of the rules, based on some other criteria.

References

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB 1994)*, pages 487–499, 1994.



(a) *address*, $\sigma = 62$ (b) *moby*, $\sigma = 33$ (c) *abstract*, $\sigma = 250$ (d) *alarms*, $\sigma = 244000$ (e) *trains*, $\sigma = 17000$

(f) *address*, $\sigma = 8$ (g) *moby*, $\sigma = 7$ (h) *abstract*, $\sigma = 30$ (i) *alarms*, $\sigma = 6035$ (j) *trains*, $\sigma = 18$

(k) *address*, $\sigma = 1$ (l) *moby*, $\sigma = 1$ (m) *abstract*, $\sigma = 4$ (n) *alarms*, $\sigma = 1300$ (o) *trains*, $\sigma = 0.03$

Figure 5: The comparison of the number of closed association rules discovered in the five datasets with varying confidence boost thresholds, using the three different methods. For easier comparison, the number of rules is expressed as a proportion of the total number of closed rules discovered at each confidence threshold. (a)-(e) fixed windows, (f)-(j) minimal windows, (k)-(o) weighted windows.

[2] R. Agrawal and R. Srikant. Mining sequential patterns. *11th International Conference on Data Engineering (ICDE 1995)*, 0:3–14, 1995.

[3] J. L. Balcázar. Formal and computational properties of the confidence boost of association rules. *CoRR*, abs/1103.4778, 2011.

[4] Y. Bastide, N. Pasquier, R. Taouil, G. Stumme, and L. Lakhal. Mining minimal non-redundant association rules using frequent closed itemsets. In *Computational Logic*, pages 972–986, 2000.

[5] T. Calders, N. Dexters, and B. Goethals. Mining frequent itemsets in a stream. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007)*, pages 83–92, 2007.

[6] G. Casas-Garriga. Discovering unbounded episodes in sequential data. In *Knowledge Discovery in Databases: PKDD 2003, 7th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 83–94, 2003.

[7] G. Casas-Garriga. Summarizing sequential data with closed partial orders. In *Proceedings of the SIAM International Conference on Data Mining (SDM 2005)*, pages 380–391, 2005.

[8] B. Cule and B. Goethals. Mining association rules in long sequences. In *Proceedings of the 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2010)*, pages 300–309, 2010.

[9] B. Cule, B. Goethals, and C. Robardet. A new constraint for mining sets in sequences. In *Proceedings of the SIAM International Conference on Data Mining (SDM 2009)*, pages 317–328, 2009.

[10] B. Cule, N. Tatti, and B. Goethals. Marbles: Mining association rules buried in long event sequences. In *Proceedings of the SIAM International Conference on Data Mining (SDM 2012)*, pages 248–259, 2012.

[11] M. Garofalakis, R. Rastogi, and K. Shim. Mining sequential patterns with regular expression constraints. *IEEE Transactions on Knowledge and Data Engineering*, 14(3):530–552, 2002.

[12] R. Gwadera, M. J. Atallah, and W. Szpankowski. Markov models for identification of significant episodes. In *Proceedings of the SIAM International Conference*

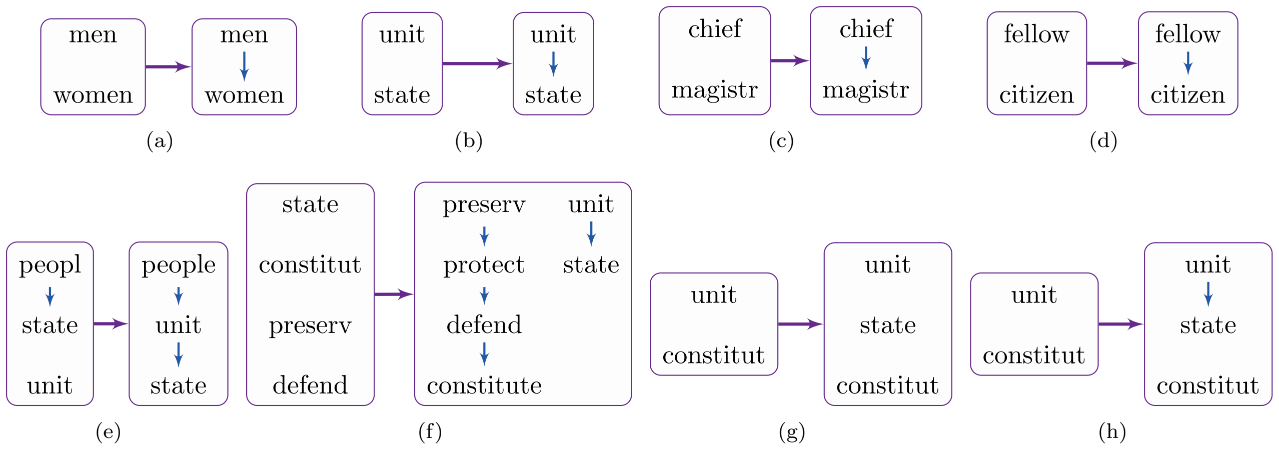


Figure 6: Examples of association rules discovered in the *address* dataset. (a) The most frequent rule with $c_f = 1$. (b) The most frequent rule with $c_f > 0.8$ and with $c_w > 0.8$. (c) The most frequent rule of size 2 with $c_w = 1$. (d) The most frequent rule with $c_m > 0.8$. (e) The most frequent rule with $c_w = 1$. (f) The most frequent rule of size 6 with $c_w = 1$. (g) A rule that can be removed using the confidence boost threshold, $c_f \approx 0.88$. (h) A more informative rule than the one in (g), $c_f \approx 0.87$.

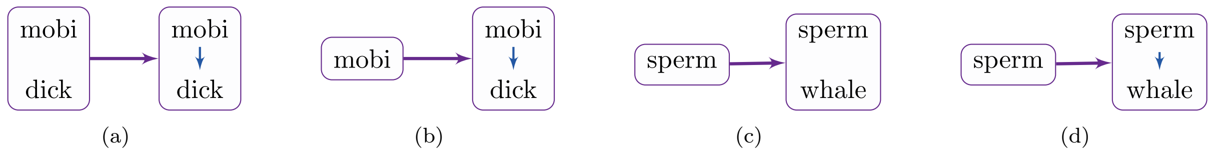


Figure 7: Examples of association rules discovered in the *moby* dataset. (a) The most frequent rule with $c_f = 1$. (b) The most frequent rule with $c_m = 1$. (c) A rule that can be removed using the confidence boost threshold, $c_m \approx 0.87$. (d) A more informative rule than the one in (c), $c_m \approx 0.84$.

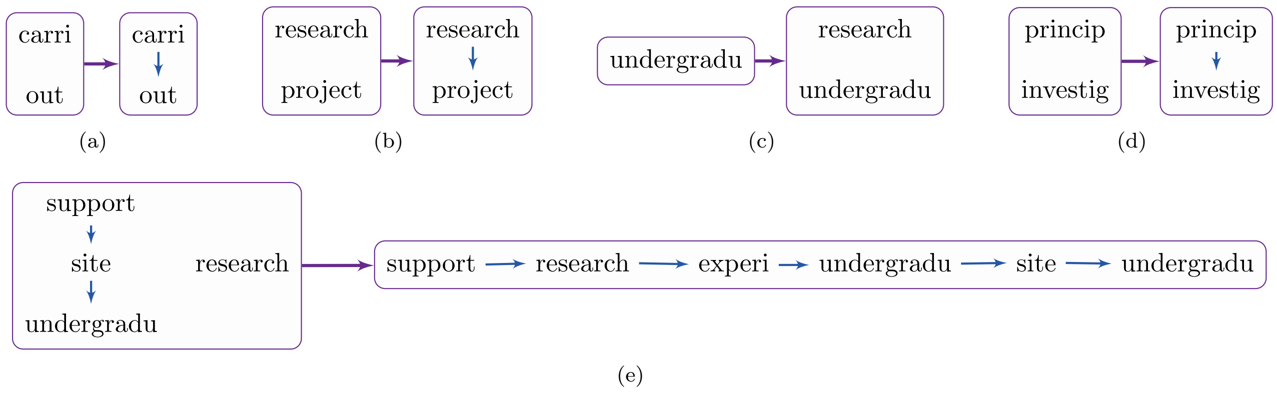


Figure 8: Examples of association rules discovered in the *abstract* dataset. (a) The most frequent rule of size 2 with $c_f = 1$, $c_m = 1$ and $c_w = 1$. (b) The most frequent rule with $c_f > 0.8$. (c) The most frequent rule with $c_m > 0.8$. (d) The most frequent rule with $c_w > 0.8$. (e) The most frequent rule of size 6 with $c_m = 1$.

on Data Mining (SDM 2005), pages 404–414, 2005.

[13] R. Gwadera, M. J. Atallah, and W. Szpankowski. Reliable detection of episodes in event sequences. *Knowledge and Information Systems*, 7(4):415–437, 2005.

[14] S. K. Harms, J. S. Deogun, J. Saquer, and T. Tadesse. Discovering representative episodal association rules from event sequences using frequent closed episode sets and event constraints. In *Proceedings of the IEEE International Conference on Data Mining (ICDM 2001)*, pages 603–606, 2001.

[15] S. Laxman, P. S. Sastry, and K. P. Unnikrishnan. A fast algorithm for finding frequent episodes in event streams. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge discovery and data mining (KDD 2007)*, pages 410–419, 2007.

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

[17] N. Méger and C. Rigotti. Constraint-based mining of episode rules and optimal window sizes. In *Knowledge Discovery in Databases: PKDD 2004, 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 313–324, 2004.

[18] J. Pei, H. Wang, J. Liu, K. Wang, J. Wang, and P. S. Yu. Discovering frequent closed partial orders from strings. *IEEE Transactions on Knowledge and Data Engineering*, 18(11):1467–1481, 2006.

[19] N. Tatti. Significance of episodes based on minimal windows. In *Proceedings of the 9th IEEE International Conference on Data Mining (ICDM 2009)*, pages 513–522, 2009.

[20] N. Tatti and B. Cule. Mining closed episodes with simultaneous events. In *Proceedings of the 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD 2011)*, pages 1172–1180, 2011.

[21] N. Tatti and B. Cule. Mining closed strict episodes. *Data Mining and Knowledge Discovery*, 2011. `http://dx.doi.org/10.1007/s10618-011-0232-z`.

[22] P. Tzvetkov, X. Yan, and J. Han. Tsp: Mining top-k closed sequential patterns. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 2003)*, pages 347–354, 2003.

[23] J. Wang and J. Han. Bide: Efficient mining of frequent closed sequences. *20th International Conference on Data Engineering (ICDE 2004)*, 0:79, 2004.

[24] J. T.-L. Wang, G.-W. Chirn, T. G. Marr, B. Shapiro, D. Shasha, and K. Zhang. Combinatorial pattern discovery for scientific data: some preliminary results. *ACM SIGMOD Record*, 23(2):115–125, 1994.

[25] X. Yan, J. Han, and R. Afshar. Clospan: Mining closed sequential patterns in large datasets. In *Proceedings of the SIAM International Conference on Data Mining (SDM 2003)*, pages 166–177, 2003.

[26] W. Zhou, H. Liu, and H. Cheng. Mining closed episodes from event sequences efficiently. In *Proceedings of the 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining(1)*, pages 310–318, 2010.