

# Association Rules Discovery in Multivariate Time Series\*

© Elena Lutsiv

University of St.-Petersburg, Faculty of Mathematics and Mechanics  
eluciv@math.spbu.ru

## Abstract

A problem of association rules discovery in a multivariate time series is considered in this paper. A method for finding interpretable association rules between frequent qualitative patterns is proposed. A pattern is defined as a sequence of mixed states. The multivariate time series is transformed into a set of labeled intervals and mined for frequently occurring patterns. Then these patterns are analyzed to find out which of them occur close to each other frequently. Some modifications and improvements of the method are proposed and discussed.

## 1 Introduction

Many classes of data that we deal with in our everyday life are temporal in their nature: medical information about patients and their diseases, goods selling data, financial data from stock markets, etc. In most cases they describe development of some variables or objects over time. All these data provide relatively small amount of knowledge by themselves, but much more information can be obtained from objects behavior analysis. Such analysis that aimed at extracting previously unknown rules from temporal data is called Temporal Data Mining or Knowledge Discovery. A comprehensive and detailed overview of temporal data mining techniques can be found, for example, in [3] and [12].

Discovered knowledge can be used both to improve understanding of underlying processes and for time series prediction given some observations in the past. In both cases it is important for all found regularities, patterns and rules to be understandable and interpretable by a domain expert. It seems that the best way for this is to create a model of time series behavior (e.g., ARMA or ARIMA models [4]), but in many cases these models are difficult to understand for a human. Furthermore, for many series, such as stock market data, it is impossible to develop a global model due to their

chaotic nature. On the other hand, when someone thinks of the system behavior, he rather keeps in mind some dependencies between typical local development pieces or scenarios. When an expert inspects system behavior over time, usually he tries to find some relatively short and simple pieces of history that occur frequently enough and are easy to interpret. The next natural step is to find out some cause-effect, coincidence or some other dependencies between frequent episodes. For example, he may find that “IF pressure falls and it is summer THEN rain will start in 24 hours with high confidence”. In this paper we propose a method for such association rules discovery from a multivariate temporal sequence.

Automation of this process requires a definition of time series similarity. Frequent episodes obtained with different measures traditionally used for estimating similarity (e.g. Euclidian distance) provide a little information about system behavior that can be interpreted by a human ([1], [5]). In this paper we use a different way to discover frequent time series episodes (or patterns). Similar approaches are used, for example, in [10] and [14]. This kind of process is believed to be much closer to a process used by a human (see [10]). The main idea is to divide up the time series into a set of labeled intervals. Each interval is a time interval during which some condition is true in the original series (for example, “a series decreases” or “a patient has flu”). This can be done in different ways: intervals and their labels can be identified empirically or as a result of some automatic process, e.g. short subsequences clustering [6]. Once we have the initial set of intervals, we build all intersections of all subsets of intervals and add them to our set. Finally we know all segments of time series where one or more simple conditions hold simultaneously.

This paper proposes the method of frequent patterns discovery from the described interval set. A pattern is defined as a sequence of state labels (simple or mixed). More formally it will be defined later. An algorithm of finding such pattern’s occurrences has been developed and is described in the paper.

The paper is organized as follows: Section 2 contains a brief overview of similar works; Section 3 defines a notion of a pattern and describes frequent patterns discovery procedure; in Section 4 a process of association rules generation is briefly outlined; Section 5 is devoted to discussions of the method and some

\* This work was partially supported by RFBR (grant 07-07-00268a).

modifications are proposed there. Section 6 contains conclusions and outlines further work directions.

## 2 Related Work

There are a number of approaches dealing with symbol sequences. In [6] a set of symbols is obtained from a time series by clustering short subsequences extracted with a sliding window. Then the sequence is mined for associations between symbols that are close enough to each other. In [13] an *a priori* style algorithm is used to discover frequent Episodes in a symbol sequence. In [8] authors apply genetic programming for pattern mining and use special hardware for efficient sequence matching.

The following approaches work with interval sequences produced from the time series, for example, using feature extraction. In [11] a sequence of intervals is searched for containment relations. In [7] a time series is divided into a set of labeled intervals. Then the interval set is mined for patterns described in terms of Allen's [2] interval logic. Interesting patterns are restricted to A1 patterns, where operators can be appended only on the right hand side. Another method that uses Allen's operators for pattern definition is proposed in [10]. A set of labeled intervals is obtained from a multivariate time series and mined for patterns with a sliding window to restrict pattern length. Then patterns are filtered by their interestingness using J-measure [15].

The last mentioned method is close to the proposed approach, but its has two main disadvantages. The first is that pattern's frequency strongly depends on the width of the sliding window. If an occurrence of a pattern is a little longer than the window, it's not taken into account at all. The second disadvantage is that patterns in [10] are expressed in terms of interval logic, that makes them difficult to understand.

The approach proposed in this paper solves both of these problems. It uses "windows" of all widths, that significantly smooths the difference between frequencies of patterns with instances of close lengths. Patterns are expressed in terms of simultaneous states – in the way that is closer to the human reasoning. The author believes that a description like "pressure increases and temperature decreases for 1 hour, and then (may be after some time) pressure slightly decreases" is more natural than "an interval where pressure increases is overlapped by an interval where temperature decreases, and the latter one is met by an interval where pressure slightly decreases". Moreover, since the first statement involves only important sequence pieces, it matches more segments of the time series than the second one.

There is another work ([14]) that uses an idea of simultaneous states, but in rather simplified form: it's authors state that a labelled interval matches an "Event" (a set of simultaneously holding states) if a set of states holding on this interval exactly coincides with the Event. The method proposed in the current paper uses more flexible approach that allows to treat an interval as

matching even if some conditions except ones defined by the pattern are held on it.

## 3 Frequent Patterns Discovery

The frequent patterns discovery procedure includes three general steps:

1. Extract a basic set of simple intervals from the time series (i.e. intervals labeled with simple states);
2. Convert an initial time series into a sequence of non-overlapping intervals labeled with mixed states;
3. Mine the sequence for frequent patterns.

This section gives a definition of a pattern and describes all these stages in details.

### 3.1 Basic Interval Set

As was mentioned above, to be able to discover time series behavior qualitative patterns, we need to extract a set of labeled intervals from the series. Let's denote as  $S_0$  a set of all simple states or conditions based on which we divide our series into intervals (for example, "a patient has a high temperature" or "humidity goes up"). States in  $S_0$  are not required to be mutually exclusive, so intervals of the resulting set may overlap. This fact allows us not to restrict ourselves with only single univariate time series, but freely work with multivariate time series or even with several ones as well. Thus we obtain a set  $I_0$  from our time series – a set of intervals labeled with simple states:

$$I_0 = \{(s_1, b_1, e_1), (s_2, b_2, e_2), \dots, (s_n, b_n, e_n)\} \quad (1)$$

Here  $(s_i, b_i, e_i)$  denotes an interval labeled with a state  $s_i \in S_0$ , beginning at  $b_i$  and ending at  $e_i$ . All these intervals are required to be maximal intervals. This means that in  $I_0$  there are no adjacent intervals or intervals with non-empty intersection labeled with the same state. I.e.:

$$\forall i=1..n, j=1..n: s_i = s_j \Rightarrow e_j < b_i \vee e_i < b_j \quad (2)$$

For the next step we need to order states of  $S_0$  in some way. A choice of ordering procedure is not very important. For example, states may be ordered lexicographically according to names of their labels or somehow else. This is needed only to enable ordering of all intervals of  $I_0$  according to the rules below:

- $\forall (s_i, b_i, e_i), (s_j, b_j, e_j) \in I_0$ :
- If  $b_i < b_j$  then  $(s_i, b_i, e_i) < (s_j, b_j, e_j)$ ;
  - If  $b_i > b_j$  then  $(s_i, b_i, e_i) > (s_j, b_j, e_j)$ ;
  - If  $b_i = b_j$  then:
    - If  $s_i < s_j$  then  $(s_i, b_i, e_i) < (s_j, b_j, e_j)$ ;
    - If  $s_i > s_j$  then  $(s_i, b_i, e_i) > (s_j, b_j, e_j)$ ;
    - Note that  $s_i = s_j$  is impossible due to the maximality requirement.

$I_0$  – is a set of maximal simple intervals – that is ordered as described above, we call it a *basic interval set*. An example is shown on the Fig. 1. Here  $S_0 = \{A, B, C, D\}$  and  $I_0 = \{(A, 1, 4), (B, 2, 3), (C, 2, 6), (D, 3, 5)\}$ .

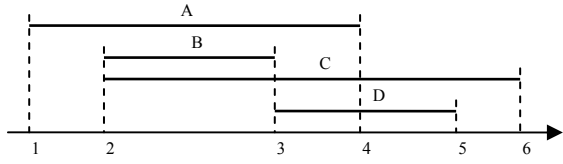


Fig. 1. An example of interval set

### 3.2 Patterns

As was mentioned above, the main goal of this method is to find association rules for patterns expressed in terms of simultaneous or mixed states. A *mixed state* is any subset of  $S_0$ . An interval  $i$  is labeled with a mixed state  $s = \{s_1, \dots, s_n\}$  if:

$$i = i_1 \cap i_2 \cap \dots \cap i_n: i_k \in I_0 \text{ and } i_k \text{ is labeled with } s_k, k = 1..n \quad (3)$$

In other words, if two or more simple intervals overlap, then their intersection is labeled with a mixed state, composed of all simple state labels of these intervals. Since  $I_0$  consists of maximal intervals, all simple states in a mixed state are different.

For further description of patterns and matching procedure we need to define a sequence  $I$  that will be searched for pattern occurrences. Let  $D = \{d_1, d_2, \dots, d_m\}$  be an ordered sequence of all different beginning and ending points of  $I_0$  intervals.  $I$  consists of all intervals  $(s, d_k, d_{k+1})$ , where  $s$  is a mixed state that includes all simple states holding during this time interval. For example, Fig. 2 (a) illustrates  $I$  for intervals shown on Fig. 1:

$$D = \{1, 2, 3, 4, 5, 6\};$$

$$I = (\{A\}, 1, 2), (\{A, B, C\}, 2, 3), (\{A, C, D\}, 3, 4), (\{C, D\}, 4, 5), (\{C\}, 5, 6)$$

Note that  $I$  consists of maximal intervals due to maximality of all intervals of  $I_0$ .

A *pattern* is defined as a sequence of mixed states. For example,  $\langle \{A\}, \{A, B\}, \{C\} \rangle$  denotes a pattern that consists of mixed states  $\{A\}$ ,  $\{A, B\}$  and  $\{C\}$ , where A, B and C are simple states.

Consider:

– a pattern  $P = \langle ps_1, \dots, ps_n \rangle$

– a subsequence  $Q$  of  $I$ :  $Q = ((s_1, b_1, e_1), (s_2, b_2, e_2), \dots, (s_m, b_m, e_m))$ , where  $(s_k, b_k, e_k) \in I$  and  $e_k \leq b_{k+1}$ .

An algorithm that matches  $P$  and  $Q$  is outlined below. After successful matching the sequence  $Q$  is broken into  $n$  groups of consecutive intervals so that  $k$ -th group is associated with  $ps_k$ .  $Q$  matches  $P$  only if  $u > n, v > m$  and all intervals of  $Q$  are associated with states of  $P$  when this process finishes.

$u := 1; v := 1;$

while ( $u \leq n$  and  $v \leq m$  and  $ps_u \subseteq s_v$ )

  while ( $(v = 1$  or  $e_{v-1} = b_v$   
  or  $(s_{v-1}, b_{v-1}, e_{v-1})$  is  
  associated with  $ps_{u-1}$ )

    and  $v \leq m$  and  $ps_u \subseteq s_v$

    and ( $u = n$  or  $ps_{u+1} \not\subseteq s_v$

    or  $ps_{u+1} \subseteq ps_u$ )  
    associate  $(s_v, b_v, e_v)$  with  $ps_u$ ;  
     $v := v + 1$ ;  
  end while  
   $u := u + 1$ ;  
end while.

There are some facts that should be noted:

1. If an interval  $(s_k, b_k, e_k)$  belongs to a group associated with  $ps_u$ , then  $ps_u \subseteq s_k$ .
2. If  $e_k < b_{k+1}$ , then  $(s_k, b_k, e_k)$  and  $(s_{k+1}, b_{k+1}, e_{k+1})$  cannot be associated with the same element of  $P$ . I. e. non-adjacent intervals cannot be associated with the same element of  $P$ .
3. If  $ps_{u+1} \subseteq ps_u$ ,  $(s_{k-1}, b_{k-1}, e_{k-1})$  is associated with  $ps_u$  and  $ps_{u+1} \subseteq s_k$ , then  $(s_k, b_k, e_k)$  will be associated with  $ps_{u+1}$  only if  $ps_u \not\subseteq s_k$ . This is due to semantics of such patterns. A pattern  $\langle \{A, B\}, \{A\} \rangle$  means that at first conditions  $A$  and  $B$  must hold for some time simultaneously and then there must be some time when  $A$  is true and  $B$  is false. If an existence of one of these intervals is not important, then the pattern is to be reduced to  $\langle \{A\} \rangle$  or  $\langle \{A, B\} \rangle$ .

Fig. 2 (b, c) shows how a sequence  $(\{A\}, \{A, B, C\}, \{A, C, D\}, \{C, D\}, \{D\})$  matches patterns  $\langle \{A\}, \{A, B\}, \{C, D\}, \{D\} \rangle$  and  $\langle \{A\}, \{B\}, \{C\}, \{D\} \rangle$ .

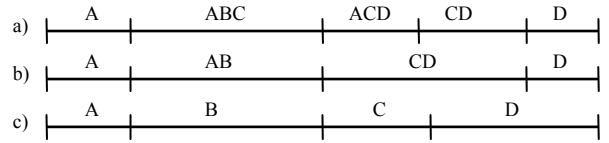


Fig. 2. Pattern matching

To find all subsequences of  $I$  that match  $P$  – occurrences of  $P$  – the following procedure is used:

1. Let  $u = 1$ .
2. Find and add to the resulting subsequence the earliest interval  $(s_{k(u)}, b_{k(u)}, e_{k(u)})$  so that  $ps_u \subseteq s_{k(u)}$ .
3. If  $ps_{u+1} \subseteq ps_u$ , then find the maximum  $d(u)$  so that  $(s_{k(u)}, b_{k(u)}, e_{k(u)}), (s_{k(u)+1}, b_{k(u)+1}, e_{k(u)+1}), \dots, (s_{k(u)+d(u)}, b_{k(u)+d(u)}, e_{k(u)+d(u)})$  are adjacent intervals and  $ps_u \subseteq s_{k(u)}, ps_u \subseteq s_{k(u)+1}, \dots, ps_u \subseteq s_{k(u)+d(u)}$ . Add these intervals to the resulting subsequence.
4. If  $ps_{u+1} \not\subseteq ps_u$ , then find the maximum  $d(u)$  so that  $(s_{k(u)}, b_{k(u)}, e_{k(u)}), (s_{k(u)+1}, b_{k(u)+1}, e_{k(u)+1}), \dots, (s_{k(u)+d(u)}, b_{k(u)+d(u)}, e_{k(u)+d(u)})$  are adjacent intervals and  $ps_u \subseteq s_{k(u)}, ps_{u(u)} \subseteq s_{k(u)+1}, \dots, ps_u \subseteq s_{k(u)+d(u)}$  and  $ps_{u+1} \not\subseteq s_{k(u)}, ps_{u+1} \not\subseteq s_{k(u)+1}, \dots, ps_{u+1} \not\subseteq s_{k(u)+d(u)}$ . Add these intervals to the resulting subsequence.
5. Repeat steps 2 – 4 with the next  $u$  and all intervals beginning later than  $e_{k(u)+d(u)}$  until  $u > n$  or the end of  $I$  is reached. If all elements of  $P$  are processed, then the resulting sequence matches  $P$ .
6. Repeat steps 1 – 5 with all intervals beginning later than  $e_{k(1)+d(1)}$  until the end of  $I$  is reached.

### 3.3 Frequent Patterns Discovery

In this section the procedure of frequent patterns discovery is described.

Let a time series under consideration be of length  $L$ . Consider a time “window” of width  $w \leq L$  that overlaps  $[0, L]$ . There are  $(L + w)$  different windows of width  $w$ . Consider a set of all different windows of length  $\leq L$ . This set is of size  $3L^2/2$ .

Consider a pattern  $P$  of cardinality  $n$  and a set of all intervals  $\{[pb_1, pe_1], [pb_2, pe_2], \dots, [pb_r, pe_r]\}$  so that for each  $1 \leq i \leq r$ :

- $pb_i < pb_{i+1}$  (if  $i < r$ );
- there is a subsequence  $((s_1, b_1, e_1), (s_2, b_2, e_2), \dots, (s_m, b_m, e_m))$  of  $I$  matching  $P$  so that:
  - if  $n \geq 2$ , then:
    - $e_x = pb_i$  and  $b_y = pe_i$  (where  $(s_x, b_x, e_x)$  is the last interval associated with  $ps_i$ , and  $(s_y, b_y, e_y)$  is the first interval associated with  $ps_n$  during matching);
    - there are no shorter (i.e. with less  $|b_y - e_x|$ ) subsequences of  $I$  matching  $P$  within  $[pb_i, pe_i]$ .
  - if  $n = 1$ , then  $pb_i = b_1$  and  $pe_i = e_1$ .

$n \geq 2$ :

$$|supp(P)| = \sum_{k=1}^r \left( (pb_k - pb_{k-1})(L - pl_k) - \frac{(pb_k - pb_{k-1})^2}{2} \right) \quad (4)$$

$n = 1$ :

$$|supp(P)| = \sum_{k=1}^r \left( (pe_k - pe_{k-1})(L + pl_k) - \frac{(pe_k - pe_{k-1})^2}{2} - \frac{pl_k^2}{2} \right)$$

$supp(P)$  – a support of  $P$  – is a set of all windows of length  $\leq L$  that contain at least one of  $[pb_i, pe_i]$  (if  $n \geq 2$ ) or overlap with at least one of  $[pb_i, pe_i]$  (if  $n = 1$ ),  $1 \leq i \leq r$ .

This definition is similar to a support definition given in [10], but considers “sliding windows” of any length between  $0$  and  $L$ , that allows not to limit a length of a pattern occurrence. In common words it can be expressed like this: if we take any window of length  $\leq L$ , and consider it as a time series, and try to find occurrences of  $P$  in it, and succeed, then this window is an element of  $supp(P)$ .

$|supp(P)|$  is a cardinality of  $supp(P)$ . A pattern is called *frequent* if  $|supp(P)| \geq supp_{min}$ . Assuming  $pb_0 = -L + pe_1$  and  $pe_0 = -L + pb_1$ , (4) is a formula for  $|supp(P)|$  evaluation.

An important property of these definitions is that any subpattern of a frequent pattern is also frequent. This enables the following procedure of frequent pattern discovery: find all frequent patterns of length 1, then construct patterns of length  $k+1$  by appending one mixed state to frequent patterns of length  $k$  and get a set  $F_{k+1}$  – a set of all frequent patterns of length  $k+1$ . The process finishes when no new frequent patterns can be found. In more details the procedure is described below:

*Step 1*: Scan  $I$  and find all patterns  $\langle ps \rangle$  of length 1 so that there exists an interval  $(s, b, e) \in I$  with  $ps \subseteq s$ . Add all frequent patterns to  $F_1$ .

*Step  $k+1$* : For each  $P = \langle ps_1, \dots, ps_k \rangle \in F_k$  find all patterns  $Q = \langle qs_1, \dots, qs_k \rangle \in F_k$  so that  $qs_1 = ps_2, \dots, qs_{k-1} = ps_k$ , and generate patterns  $\langle ps_1, \dots, ps_k, qs_k \rangle$  from  $P$  and each described  $Q$ . Test all generated patterns and add all frequent ones to  $F_{k+1}$ .

### 4 Association Rules Discovery

Once all frequent patterns are discovered, we can try to find association rules. Let’s denote a rule as  $A \mapsto B$ . But first of all we need to define which patterns will be considered as associated. This must be some condition (*cond*) describing how antecedent pattern ( $A$ ) and succedent pattern ( $B$ ) are disposed. For example, this can be “ $B$  starts during  $A$  and ends within 2 hours after  $A$  ends” or “ $B$  starts within 5 minutes after  $A$  ends”. This condition is to be chosen by a domain expert so that discovered rules would be easy to interpret.

Consider all occurrences of  $A$  and  $B$  that satisfy condition *cond*. Let  $supp_{A \mapsto B}(A)$  and  $supp_{A \mapsto B}(B)$  be a supports of  $A$  and  $B$  respectively, that are built basing only on such occurrences. Support of  $A \mapsto B$  is defined in (5).

$$supp(A \mapsto B) = supp_{A \mapsto B}(A) \cap supp_{A \mapsto B}(B) \quad (5)$$

The simplest way to generate association rules is to evaluate a confidence:

$$conf(A \mapsto B) = \frac{|supp(A \mapsto B)|}{|supp(A)|}$$

for each pair of frequent patterns and select the rule only if  $conf(A \mapsto B) \geq conf_{min}$ . But this will result in a large set of relatively useless rules. For example, if occurrences of  $B$  cover almost all time series, then all rules with  $B$  as a succedent will be selected, but most of them will be of small interest.

$$J(B; A) = p(A) \left( p(B|A) \log \left( \frac{p(B|A)}{p(B)} \right) + (1 - p(B|A)) \log \left( \frac{1 - p(B|A)}{1 - p(B)} \right) \right) \quad (6)$$

[9] gives a survey of the most successful interestingness measures used in knowledge discovery. One of the most popular measures for association rules evaluation is a J-measure [15] (see (6)). It is a special case of Shannon’s mutual information and takes into account both rule frequency and rule “surprisingness”. If J-measure of a rule is less than some threshold value, then it is not interesting enough and should be excluded from consideration.

In our context  $p(B)$  is a probability of pattern  $B$  occurring in randomly selected window of length  $\leq L$ , and  $p(A \mapsto B)$  is a probability of the fact that  $B$  occurs in a randomly selected window of length  $\leq L$  that contains an occurrence of  $A$  and *cond* is true for these occurrences. The probabilities are evaluated as:

$$p(B) = \frac{|supp(B)|}{3L^2/2}; \quad p(B|A) = \frac{|supp(A \mapsto B)|}{|supp(A)|} \quad (7)$$

## 5 Discussion and Evaluation

There are a number of issues that should be discussed about the proposed method. One of them is a number of mixed states. In general case, having  $N$  simple states, we obtain  $2^N$  possible mixed states. But in practice some of them cannot take place simultaneously (or can hold only simultaneously) due to some reason. For example, consider 3 time series: pressure, temperature and humidity data, and 6 simple states for each of them: “increase”, “slow increase”, “fast increase”, “decrease”, “slow decrease”, “fast decrease”. There are  $2^{3*6}$  possible mixed states, but it is obvious that slowly increasing and fast increasing segments of the same time series cannot overlap, or decreasing and fast decreasing segments of the series always overlap, and so on. Thus the number of possible mixed states is reduced to  $7^3$  (or even  $5^3$  if any decreasing/increasing segment is either fast or slowly decreasing/increasing). Moreover, possibly not all of them have support exceeding  $supp_{min}$ , so they will be removed from consideration after the first step of frequent patterns discovery procedure.

Another problem is very short mixed-state intervals appearing due to inaccurate initial time series division (especially if it was performed by a human). On Fig. 3 (a) the bounds of intervals labeled with simple states A and B were defined roughly, and this resulted in a small interval labeled with a mixed state {A, B}. A simple solution is to ignore intervals shorter than some threshold  $\varepsilon$  when searching for pattern occurrences. For example, if  $l < \varepsilon$ , then a sequence from the Fig. 3 (a) does not match a pattern  $\langle \{A\}, \{A, B\}, \{B\} \rangle$ . The same idea can be applied to situation when there is a small gap between intervals labeled with the same state (see Fig. 3 (b)). Such gaps may appear, for example, due to smoothing faults. In this case it may be ignored during patterns discovery and both intervals may be considered as one continuous interval.

But what if the user wants to have patterns that exactly define which intervals must be adjacent and which may have a gap between them in a matching sequence? For example, he wants to extend pattern  $\langle \{A\}, \{B\}, \{C\} \rangle$  with some information like “in a matching sequence {A}-interval must be met by {B}-interval, but there may be a gap between {B}-interval and {C}-interval”. Such patterns can be enabled by introducing of a special symbol “gap” ( $\square$ ). A pattern described above can be expressed like  $\langle \{A\}, \{B\}, \square, \{C\} \rangle$ . “Gap” is not a mixed state, so pattern cardinality does not change, but a space of all candidate patterns increases.

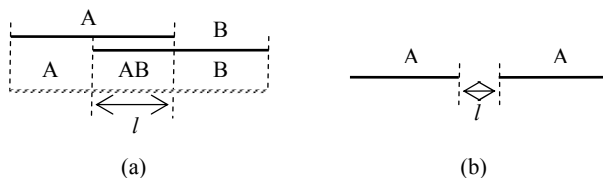


Fig. 3. Too short intervals

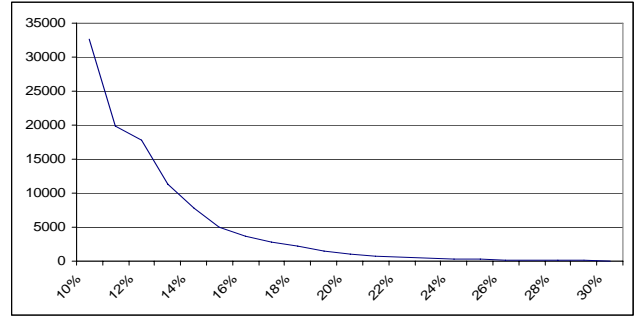


Fig. 4. Dependency of frequent patterns number from  $supp_{min}$

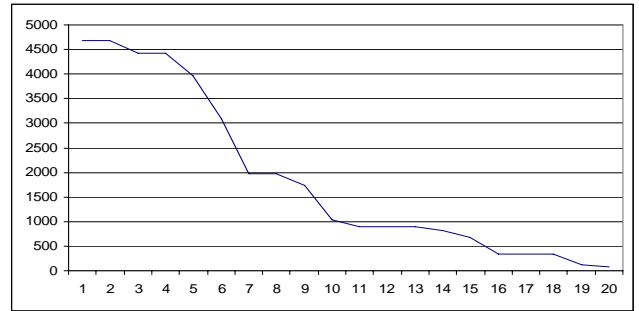


Fig. 5. Dependency of frequent patterns number from  $\varepsilon$

Fig. 4 and Fig. 5 present dependencies of discovered frequent patterns number from  $supp_{min}$  and  $\varepsilon$  respectively. Data were obtained by applying of a pattern discovery algorithm to a simulated interval set. To simulate it the following parameters were used:

- number of variables: 3;
- number of labels: 9;
- random interval length from 1 to 100;
- total series length: 30000.

Fig. 4 shows that the number of patterns is inversely proportional to  $supp_{min}$ , and it seems that this number decreases close to linearly if  $\varepsilon$  grows (Fig. 5).

## 6 Conclusion and Future Work

This paper proposes a method and an algorithm for frequent patterns and rules discovery in labeled interval sequence. The interval sequence can be obtained, for example, from multivariate time series by dividing each variable’s behavior into labeled intervals. Since pattern is a sequence of mixed states, a domain expert can easily interpret all found frequent patterns and rules. Inaccuracy of series division and data smoothing defects can be compensated by filtering intervals with minimum interval length. The method was tested over simulated interval set and dependency of a frequent patterns number from minimum support and minimum interval length were investigated.

The problem of the proposed method is a large number of generated frequent patterns and rules. A J-measure can help to reduce it, but there still remains a problem of ranking patterns by their interpretability and

“importance”. Also an effective algorithm for frequent patterns discovery needs to be developed. Although an initial interval set is converted to a simple sequence of mixed state intervals and a pattern is either a mixed states sequence, classical methods of finding subsequence in a sequence are not suitable for pattern matching, since this process is rather complicated.

[15] Smyth, P., Goodman, R. M. Rule Induction Using Information Theory. In: Knowledge Discovery in Databases, chapter 9. MIT Press (1991) 159 – 176

## References

- [1] Agrawal, R., Lin, K.-L., Sawhney, H.S., Shim, K.: Fast Similarity Search in The Presence of Noise, Scaling and Translation in Time-Series Databases. In: Proc. of the 21st Int. Conf. on Very Large Databases, Zurich, Switzerland (1995)
- [2] Allen, J.F.: Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11) (1983) 832 – 843
- [3] Antunes, C., Oliveira, A.: Temporal Data Mining: an Overview. In: KDD Workshop on Temporal Data Mining, San Francisco (2001) 1–13
- [4] Box, G.E.P., Jenkins, G.M.: *Time Series Analysis: Forecasting and Control*. San-Francisco, Holden-Day (1976)
- [5] Brendt, D.J., Clifford, J.: Finding Patterns in Time Series: A Dynamic Programming Approach. In: *Advances in Knowledge Discovery And Data Mining*, chapter 9. MIT Press (1996) 229 – 248
- [6] Das, G., Lin, K.-L., Mannila, H., Renganathan, G., Smyth, P.: Rule Discovery from Time Series. In: Proc. of the 4th Int. Conf. on Knowledge Discovery and Data Mining, AAAI Press (1998) 16 – 22
- [7] Fu, A.W.C., Kam, P.S.: Discovering Temporal Patterns for Interval-Based Events. In: Kambayashi, Y., Mohania, M.K., Tjoa, A.M., eds.: *Second International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2000)*. Volume 1874., London, UK, Springer (2000) 317 – 326
- [8] Hetland, M.L., Saetrom, P.: Temporal Rule Discovery Using Genetic Programming and Specialized Hardware. In: Proc. of 4th Int. Conf. on Recent Advances in Soft Computing (2002) 182 – 188
- [9] Hilderman, R.J., Hamilton, H.J.: Knowledge discovery and interestingness measures: A survey. Technical Report CS 99-04, Department of Computer Science, University of Regina (1999)
- [10] Höppner, F.: Discovery of Temporal Patterns – Learning Rules about the Qualitative Behavior of Time Series. In: Proc. of the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases, Lecture Notes in Artificial Intelligence 2168, Springer (2001)
- [11] Hua, K.A., Maulik, B., Tran, D., Villafane, R.: Mining Interval Time Series. In: *Data Warehousing and Knowledge Discovery*. (1999) 318 – 330
- [12] Lin, W., Orgun, M.A., Williams, G.J.: An Overview of Temporal Data Mining. In: *Proceedings of the 1st Australian Data Mining Workshop*, Canberra, Australia (2002)
- [13] Mannila, H., Toivonen, H., Verkamo, A.I.: Discovery of Frequent Episodes in Event Sequences. *Data Mining and Knowledge Discovery* 1 (1997) 259 – 289
- [14] Moerchen, F., Ultsch, A.: Mining Hierarchical Temporal Patterns in Multivariate Time Series. In: Proc. of the 27th German Conference on Artificial Intelligence (KI). Ulm, Germany (2004)