About the Analysis of Time Series with Temporal Association Rule Mining

Tim Schlüter and Stefan Conrad
Institute of Computer Science
Heinrich Heine University, Düsseldorf (Germany)
{schlueter, conrad}@cs.uni-duesseldorf.de

***Abstract*—This paper addresses the issue of analyzing time series with temporal association rule mining techniques. Since originally association rule mining was developed for the analysis of transactional data, as it occurs for instance in market basket analysis, algorithms and time series have to be adapted in order to apply these techniques gainfully to the analysis of time series in general. Continuous time series of different origins can be discretized in order to mine several temporal association rules, what reveals interesting coherences in one and between pairs of time series. Depending on the domain, the knowledge about these coherences can be used for several purposes, e.g. for the prediction of future values of time series. We present a short review on different standard and temporal association rule mining approaches and on approaches that apply association rule mining to time series analysis. In addition to that, we explain in detail how some of the most interesting kinds of temporal association rules can be mined from continuous time series and present an prototype implementation. We demonstrate and evaluate our implementation on two large datasets containing river level measurement and stock data.**

I. Introduction

Time series occur in various domains in great number and heterogeneity. In the simplest and most frequent case a time series contains successive measurements of certain quantities, for instance EEG measurements in medicine (as appearing in sleep data analysis [1] and epilepsy seizure prediction [2]), measurements of sun rays in astronomy (for sunspot analysis and prediction [3]) and measurements of geological data (in order to predict river levels [4] or the El Niño phenomenon [5]). Naturally, the discovery of special coherences, the understanding of certain characteristics and the possibility of predicting future values of time series is of special interest for these domains, which has led to an intensive research in time series analysis in the last decades. The discovery of certain patterns, which are characteristic for time series and thus have a special meaning, is an important task in the area of time series analysis, and similarly, the discovery of certain pattern, namely of association rules, is an important task in the analysis of transactional data. Since transactional data with temporal information can be regarded as a special case of (multidimensional) time series, and since much research has been done in the field of temporal association rule mining, it is a natural consequence to carry these results and techniques to general time series analysis, as long as they are adequate.

This paper briefly reviews standard and temporal association rule mining techniques and their application to time series analysis. In addition to that, it explains in detail how to transfer several (temporal) association rule mining techniques - namely for discovering "standard" association rules, "basic" temporal association rules [6], sequential patterns [7], cyclic [8] and/or calendar-based association rules [9] - to the area of time series analysis. After discretizing continuous time series, we show how to apply these temporal association rule mining techniques to reveal interesting coherences (dependent on the domain) within a time series or between pairs of time series. In order to show its functionality and the influence of parameters, we evaluate this approach on two real-life datasets containing river level measurements and stock data.

The paper is organized as follows: Section II mentions the background of time series in general. The basic notions of association rule mining are given in the following section. Section IV integrates previous work in the context of association rule mining in time series and discusses some further points of interest. The next section contains several subsections, where the first explains two time series discretization techniques in detail. Subsection V-B and the following explicate the application of the temporal association rule mining approaches to time series analysis. After that, we present some details about the implementation of a system for mining temporal association rules in time series in subsection V-F, followed by an evaluation of our implementation on river level measurements and stock data in subsection V-G. Finally, in section VI we present our conclusion and identify directions for future work.

II. Time Series

In general, a time series s can be described as a sequence $(x_1, ..., x_n)$ containing n data points x_i. These data points can consist of real numbers, for instance of the river level [4] or the voltage of an EEG derivation [1] measured at certain typically equidistant points in time; or more complex, they can be highly multidimensional, for example in market basket analysis [10], where x_i corresponds to a customer's transaction containing e.g. the time of the transaction, a customer ID and bought items. The latter is an example of a very high dimensional discrete time series, whereas the time series derived from the first two examples are typically continuous and typical for the area of time series analysis.

978-1-4244-9927-4/11/$26.00 ©2011 IEEE

Association rule mining is the semi-automatic process of discovering interesting relationships in huge amounts of data (cf. section III for more details). Originally, it was designed for a very complex class of time series, namely for time series containing transactions, as they appear for instance in market basket analysis. For using techniques from this field in order to reveal interesting relationships in "normal" time series (we assume "normal" time series to contain continuous numerical values), these time series have to be transformed first. This transformation can be done by discretizing. Formally, a continuous time series s is transformed into a discrete representation $D(s) = a_1 a_2 ... a_j$, with typically $j \ll n$ and $a_i \in \Sigma$, where Σ is an arbitrary alphabet, by several methods. A positive side-effect of this transformation is the complexity and dimensionality reduction, which is very useful when dealing with large time series with very high granularities. An overview about some discretization methods will be given in section IV, and two of these methods will be explicated in subsection V-A in more detail.

III. Standard Association Rule Mining

Before introducing temporal association rule mining and integrating them in the context of time series analysis, we first have to define standard association rule mining ("standard" in contrast to "temporal" association rule mining, which pays special regard to the temporal component, cf. subsection V-B) and some details, which will be used in the following.

Let $D = \{t_1, t_2, ..., t_N\}$ be the transactional database, which contains N transactions t_i. Every t_i usually consists of a timestamp, which states when the transaction has occurred, an itemset $A \in I$, where $I = \{i_1, i_2, ..., i_d\}$ is the set of all items (e.g. corresponding to products sold in a supermarket), and further optional parameters (e.g. customer ID). (Note, that the transactions could also be presented in form of a time series $d = (t_1, t_2, ..., t_N)$ as introduce in the previous section, but we prefer to use the typical notions of association rule mining as given in this section.) The absolute support of an itemset A is the number of occurrences of A in D, which is formally defined as $sup_a(A) = |\{i \mid A \subseteq t_i \wedge 1 \leq i \leq N\}|$. The relative support, which is the number of occurrences of A divided by the number of all transactions in D, is defined as $sup_r(A) = \frac{sup_a(A)}{N}$.

An *association rule* is an implication of the form $A \Rightarrow B$, where A and B are two disjoint itemsets. The meaning of $A \Rightarrow B$ is, that if itemset A occurs in transaction t_i, itemset B will most likely be in that transaction too. More concrete and applied to market basket analysis, $A \Rightarrow B$ states that customers, who buy items from itemset A, tend to buy the items from itemset B too. An example for such an association rule might be $\{\text{computer}\} \Rightarrow \{\text{internet security package}\}$. The knowledge derived from the discovery of association rules can be used in many ways, for instance in product placement. (The given definition of $A \Rightarrow B$ is the standard definition of an association rule, without special regard to the temporal component. Several kinds of temporal association rules, which are defined through slight modifications of the standard definition, are introduced in the following sections.) Typically, the strength of an association rule is measured in the two terms support and confidence. Support determines how often a rule is applicable to a given database. Again, we distinguish between absolute support (sup_a) and relative support (sup_a) of an association rule $A \Rightarrow B$, which are defined as follows:

$$sup_a(A \Rightarrow B) = sup_a(A \cup B),$$

$$sup_r(A \Rightarrow B) = \frac{sup_a(A \Rightarrow B)}{N}.$$

Confidence determines how frequently items in B appear in transactions that contain A, i.e. how reliable an association rule is; it is defined by

$$conf(A \Rightarrow B) = \frac{sup_a(A \Rightarrow B)}{sup_a(A)}.$$

An association rule with a very low support can simply occur coincidentally and thus be uninteresting for the outcome of the data analysis. To avoid this, a threshold `minsup` for the minimal support of a rule is given by the user. Itemsets or association rules that have a support higher than `minsup` are called *frequent*. Analogously, a threshold `minconf` can be defined for the minimal confidence. An itemset or an association rules, that satisfies both the `minsup` and `minconf` constraint is called *strong*.

IV. Related Work and Discussion

In the last two decades there has been an intense research interest in the discovery of association rules. One of the first papers about mining association rules in large databases is [11], which introduces the basic notions of association rule mining (cf. section III). In one of their following papers, the same authors introduce the apriori algorithm [12], which exploits the downward closure property of frequent itemsets ("every subset of a frequent itemset is also frequent") for mining association rule more efficiently (cf. subsection V-F). The apriori algorithm is the quasi-standard algorithms for mining standard association, which is used as basis for many following papers from different authors. A completely different approach without apriori-typical candidate generation is [13], which proposes the FP growth algorithm that is quite comparable in efficiency to apriori [14]. Another tree-based approach for efficiently mining association rules besides FP growth is presented in [15].

The approaches mentioned before aim at discovering standard association rules without special regard to the temporal component. One of the first approaches for discovering temporal association rules is [7], which introduces the notion of sequential patterns. Sequential patterns are patterns derived from events that happen sequentially (c.f. subsection V-B for more detail), they have been generalized in [16] and their discovery has been made more efficient in several works (e.g. in [16] and [17]). A similar concept are episodes, which are collections of events that occur relatively close to each other ([18], [19]). Other kinds of temporal association rules involve

specific temporal constraints, e.g. like the ones captured by the simple rule format $A \Rightarrow^T B$ proposed in [6], which denotes "if A occurs, then B occurs within time T" (to which we also refer as "basic" temporal association rule), or the "sequential association rules with constraints and time lags" proposed in [20]. Other constraints are demanded for discovering more natural sounding kinds of temporal association rules, e.g. for cyclic association rules (rules from events that happen cyclically) [8], periodic association rules [21] or for rules, which hold in specific time intervals that can be described by calendar-based patterns [9]. Subsection V-B and the following explicate the simple rule format, sequential pattern, cyclic and calendar-based association rules and their application to time series analysis in more detail.

Most of these association rule mining approaches base on timestamped data, where only the point in time when the transaction has occurred is known. Since for some kind of data the start and end time point of a transaction is available, there are also approaches for discovering association rules on interval-based data, e.g. ([22], [23], [24]), which consider the relationships between intervals in terms of Allen's interval logic [25]. In [23], this kind of temporal association rule mining is named state sequences mining.

Since so much work has been invested in the discovery of (temporal) association rules, it is a logical consequence to transfer these approaches to time series analysis in general (i.e. to continuous time series with numerical values), in order to find interesting relationships in time series with these methods efficiently too. Originally, association rule mining was designed for transactional data, thus the time series have to be transformed in an appropriate format to which the basic ideas of association rule mining can be applied. This transformation can be done by discretizing the continuous time series, for instance with the clustering-based approach presented in [6] or SAX [26] (cf. subsection V-A for details about these two approaches).

As first step in the discretization process the time series have to be segmented applicatively, either in segments of the same or of different lengths (cf. for instance ([27], [28]) for an overview). After that a symbol (or a "state" as denote in [23]) is assigned to each segment, resulting in the discretized representation $D(s) = a_1 a_2 ... a_j$ as introduce in section II. The discretized representation can either be seen as sequence of points a_i, where each point contains start and end time (or optionally, one of these two time points or the mean) and the symbol/state assigned to this segment, or simply as string containing the assigned symbols in the accordant order (if the segments are of the same length). The two discretization methods presented in [6] and [26] result in segment of the same length, i.e. every symbol derived from a segment has the same duration. In contrast to the inductive approaches, where shapes of time series are derived directly from the time series (e.g. like in the clustering-based approach [6], where the symbols represent shapes obtained by clustering all subsequences, which are derived by a sliding window), deductive approaches fix the shapes of interest in advance, for instance as proposed in [29] in terms like "constant", "linearly increasing" or "convexly increasing" (what can be derived by checking all possible combinations of zero/positive/negative first and second derivative, given that the signal from which the time series was derived is almost everywhere twice differentiable).

One of the first approaches that directly uses association rule mining for time series analysis is [6]. The authors present the clustering-based discretization method and the simple temporal rule format mentioned above. A very similar approach for the discovery of "sequential association rules with time lags" was proposed in [20]. An approach, that uses SAX [26] as discretization method instead of the clustering-based method from [6], is [30], which apart from that adapts [6] for mining association rules in the simple rule format. A different approach is [31], where the discretization of the time series is done by scale-space filtering [32], after which state sequences mining is applied in order to discover relationships between intervals.

V. Temporal Association Rule Mining Applied to Time Series Analysis

This section explicates the application of some temporal association rule mining approaches (basic temporal association rules, sequential patterns, cyclic and/or calendar-based association rules) to the area of time series analysis. Since continuous time series have to be transformed in an appropriate format before applying association rule mining algorithms, we first describe two discretization techniques, which we used in our implementation for analyzing time series with the former mentioned temporal association rule approaches.

A. Time Series Discretization

In the past years, a vast number of time series discretization and quantization methods have been proposed in literature; for an overview confer for instance [33], [34] and [35]. In the following, we present two methods in detail, which discretize continuous time series with equidistant points in time, namely SAX [26] and the clustering-based method proposed in [6].

SAX [26] is a symbolic aggregate approximation, which is used in several time series problems as time series representation [36]. As preprocessing step, SAX uses piecewise aggregate approximation (PAA) in order to reduce the dimension and the complexity of a time series s, i.e. the time series is divided into j equally sized frames, where the mean value of the data points falling in a frame is used as value for representing it. After being normalized, the time series has a highly Gaussian distribution [35], and the remaining data points are discretized into equiprobable symbols of a chosen alphabet Σ. Figure 1 illustrates the procedure on a cut-out of an example time series.

A discretization technique, where the alphabet Σ is derived directly from the data, is the clustering-based method proposed in [6]. It uses a sliding window mechanism, where a window

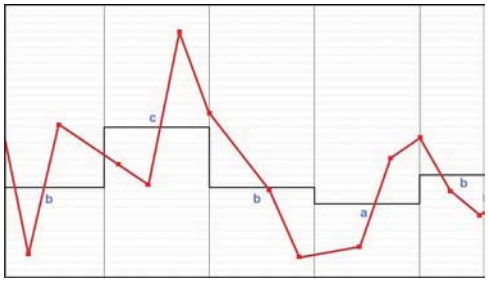

Fig. 1. Simplified illustration of SAX on a cut-out of an example time series: the time series (red with rectangles), its PAA representation (black), and the symbols assigned to the time series in each frame. With an $\Sigma = \{\mathrm{a}, \mathrm{b}, \mathrm{c}\}$, the discretization of the cut-out is "bcbab".

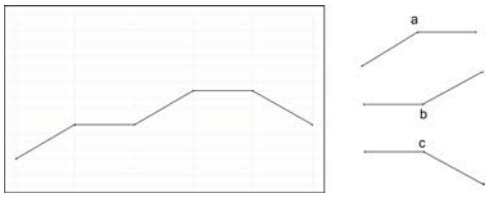

Fig. 2. Illustration of the clustering-based discretization for a simple time series $s = (1, 2, 2, 3, 3, 2)$, window size $l = 2$ and step size 1 (adapted from [6]). The primitive shapes derived through k-means clustering (with $k = 3$) are shown on the right of the plot; with $\Sigma = \{\mathrm{a}, \mathrm{b}, \mathrm{c}\}$ the discretized time series is $D(s) = \mathrm{abac}$.

of fixed size l is running with a fixed step size (in the following, assumed to be 1 for simplicity in presentation) over the data points of time series s, so that each window forms a subsequence $s_i = (x_i, ..., x_{i+l-1})$. The set of all subsequences $W(s) = \{s_i \mid 1 \leq i \leq (n - l + 1)\}$ is clustered, e.g. like in our approach by k-means (cf. for instance [37]). After clustering, each subsequence s_i belongs to a certain cluster, thus a certain symbol representing the cluster can be assigned to every subsequence s_i. (By using k-means as clustering algorithm, we obtain a clustering with exactly k clusters and thus an alphabet Σ of size k, again where each symbol represents a cluster.) The discretized time series $D(s)$ consists of the symbol representation for each subsequence in the respective order. Thus a data-derived representation of time series s is obtained, where, according to window and step size, each symbol represents a primitive shape. Figure 2 illustrates the whole procedure on a simple example time series. (Note, that the more the subsequences overlap, the more correlated they are. Thus it makes sense to allow a step size of > 1 for the sliding window movement in order to prevent too much correlation.)

B. Association Rule Mining in Time Series

When applying (temporal) association rule mining to a non-transactional database of discretized time series, the definitions have to be adapted, as will be shown in the following. The notion of absolute support of a symbol a from the discretization alphabet Σ, can be defined as the frequency of a in the discretized time series $D(s)$. For simplicity, we regard $D(s) = a_1 a_2 ... a_j$ as string in the following in order to define the notions more formally. Then, absolute and relative support of a symbol $a \in \Sigma$ are given by

$$sup_a(a) = |\{i \mid a_i = a \wedge 1 \leq i \leq j\}| \text{ and}$$

$$sup_r(a) = \frac{sup_a(a)}{j},$$

where j is the length of the discretization string $D(s)$. Analogously, absolute and relative support of a string $A \in \Sigma^+$ can be defined as

$$sup_a(A) = \left| \left\{ \; i \; \mid \; [D(s)]_i^{i+|A|-1} = A \; \right\} \right| \text{ and}$$

$$sup_r(A) = \frac{sup_a(A)}{j - |A| + 1},$$

with $1 \leq i \leq (j - |A| + 1)$, where $|A|$ denotes the length of string A and $[D(s)]_b^e$ the substring of $D(s)$, beginning at position b and ending at e (e.g. $[\mathrm{String}]_1^3 = \mathrm{Str}$).

In the analysis of a time series s the meaning of an association rule $A \Rightarrow B$ is, that if string A occurs somewhere in $D(s)$, it will most likely be followed by B. Thus the absolute and the relative support of an association rule $A \Rightarrow B$ is defined as

$$sup_a(A \Rightarrow B) = sup_a(AB) \text{ and}$$

$$sup_r(A \Rightarrow B) = \frac{sup_a(A \Rightarrow B)}{j - |AB| + 1},$$

where the denominator of the latter equals the number of possible occurrences of $A \Rightarrow B$ (AB denotes the string concatenation of A and B). With this support definition, the confidence of an association rule in a time series is defined exactly like in a transactional database, see above.

Depending on the size of the time series database, a huge number of association rules can be discovered. If the parameter `minsup` and `minconf` are set properly, this number will be reduced to a smaller number of more important association rules, through which a domain expert should browse in order to find interesting and unexpected association rules. Since the number of the remaining strong association rules might still be very large, the expert can be guided by the use of interestingness measures for association rules (cf. for instance [38]). We decided to use the J measure, which was proposed in [39], because it provides a useful and sound method for ranking rules in a manner which trades-off rule frequency and confidence [6]. The J measure for an association rule $A \Rightarrow B$ can be defined as

$$J(A; B) = P(A) \left[P(B \mid A) \log \frac{P(B \mid A)}{P(B)} + (1 - P(B \mid A)) \log \frac{1 - P(B \mid A)}{1 - P(B)} \right],$$

where $P(A)$ and $P(B)$ denote the probability of string A (or respectively string B) occurring at a random position in $D(s)$ (i.e. the relative support of A or B) and $P(B \mid A)$ the probability of B occurring at a random position in $D(s)$ with A preceding (i.e. the confidence of $A \Rightarrow B$).

There are many different kinds of temporal association rules (i.e. association rules with special regard to the temporal component) one could think of, for instance from events that happen sequentially [7], cyclically [8], periodically [21] or in special times which can be described by calendar-based patterns [9]. All these rules require their own definition of support, which will be given in the following.

C. Basic Temporal Association Rules

The simplest temporal association rule format, to which we refer as "simple" or "basic" temporal association rule, is "if A occurs, then B occurs within time T" [6], written as $A \Rightarrow^T B$ for two strings A and B and a duration T. (T can either be seen as duration in an arbitrary time unit, or, as in the following, as symbol distance in the string representation of the discretized time series.) The absolute support of such an association rule can be stated as

$$sup_a(A \Rightarrow^T B) = F(A, B, T),$$

where $F(A, B, T)$ is defined as

$$\left|\left\{ i \mid [D(s)]_i^{i+|A|-1} = A \wedge B \in [D(s)]_{i+w+1}^{i+w+T-1} \right\}\right|,$$

with $1 \leq i \leq (j - |AB| + 1)$. $F(A, B, T)$ is the number of occurrences of A, that are followed by B in the distance of T symbols. (Amongst others, the parameter w is a tribute to the clustering based discretization method used in our approach, confer subsection V-A. Since two subsequent symbols a_i and a_{i+1} are derived from two overlapping sliding windows, a_i and a_{i+1} are strongly correlated. Thus the optional distance parameter w determines the minimal gap, that should lie between the occurrences of A and B in the discretization string.) And finally, the relative support of $A \Rightarrow^T B$ can be defined as

$$sup_r(A \Rightarrow^T B) = \frac{sup_a(A \Rightarrow^T B)}{j - |AB| + 1 - w}.$$

Confidence is defined as usual, i.e.

$$conf(A \Rightarrow^T B) = \frac{sup_a(A \Rightarrow^T B)}{sup_a(A)},$$

and the J measure of $A \Rightarrow^T B$ is given by $J(A; B_T)$ [6] (detailed definition see above), where $P(B_T)$ denotes the probability of at least one B occurring in a randomly chosen window of size T in $D(s)$ (i.e. the number of windows of size T, that contain B, divided by the number of all possible windows, which is $j - T + 1$) and $P(B_T \mid A)$ the probability of at least on B occurring in a randomly chosen window of size T, which starts with A (i.e. $F(A, B, T)$ divided by the absolute support of A).

The format of the temporal association rule $A \Rightarrow^T B$ is quite simple, but highly expandable, as contemplated in [6]: A and B may come from one time series discretization $D(s)$, which allows *intra time serial* association rule mining, or A may come from time series s_1 and B from another time series s_2, which leads to pairwise *inter time serial* association rule mining between $D(s_1)$ and $D(s_2)$. The adaptation to the algorithms for calculating support and confidence values are straight forward. Note, that the rule format $A \Rightarrow^T B$ can be seen as generalization of the standard association rule format $A \Rightarrow B$, as introduced at the beginning of this subsection (if parameter T is set to null).

D. Sequential Patterns

Sequential patterns, which have been proposed in [7], are temporal association rules derived from events that happen sequentially. An obvious example from market basket analysis for such a rule might be derived from the following observation made in a video rental business: People who rent the DVD "The Fellowship of the Ring" and then "The Two Towers", most likely rent the DVD with the third part of the "Lord of the Rings" trilogy, namely "The Return of the King", thereafter too. Sequential patterns, as proposed in [7], just state that if someone rents the first and then the second movie, he will most likely rent the third movie afterwards too, but they do not state when this will happen. Since this information is interesting too, we decided to use the extension of the simple rule format $A \Rightarrow^T B$ for sequential pattern mining: "if A_1 and A_2 and ... and A_k occur within V units of time, then B occurs within time T", written as $A_1, A_2, ..., A_k \Rightarrow^{V,T} B$ [6]. Again, A_i and B may come from one or different time series discretization, allowing intra and inter time serial sequential pattern mining.

The definitions of support and confidence for all possible intra and inter time serial sequential pattern rules between one and several time series follow the basic idea behind $A \Rightarrow^T B$, thus as an example here we only give the definition for sequential pattern rule mining for $k = 2$, with A_1 and A_2 coming from time series s_1 and B coming from time series s_2: The absolute support of such a temporal association rule $A_1, A_2 \Rightarrow^{V,T} B$ is defined as

$$sup_a(A_1, A_2 \Rightarrow^{V,T} B) = F(A_1, A_2, B, V, T),$$

which states the number of occurrences of A_1 and A_2 in time series s_1 within time V, which are followed by B in s_2 in a maximal temporal distance of T (from the beginning of A_1). Formally, $F(A_1, A_2, B, V, T)$ is given by

$$\left|\left\{ i \mid [D(s_1)]_i^{i+|A_1|-1} = A_1 \quad \wedge \quad A_2 \in [D(s_1)]_{i+|A_1|}^{i+V-1} \quad \wedge \quad B \in [D(s_2)]_{x+|A_2|}^{i+T-1} \right\}\right|,$$

where x denotes the position of the beginning of A_2 in s_1. (Note, that the given definition forbids overlapping of A_1, A_2 and B. Furthermore, we removed the parameter w, which was introduced in the definition of $F(A, B, T)$ above, in the definition of $F(A_1, A_2, B, V, T)$ due to lucidity.) The relative support and the confidence of $A_1, A_2 \Rightarrow^{V,T} B$ are given by

$$sup_r(A_1, A_2 \Rightarrow^{V,T} B) = \frac{sup_a(A_1, A_2 \Rightarrow^{V,T} B)}{|D(s_2)| - |A_1 A_2 B|} \text{ and}$$

$$conf(A_1, A_2 \Rightarrow^{V,T} B) = \frac{sup_a(A_1, A_2 \Rightarrow^{V,T} B)}{F(A_1, A_2, V)},$$

and finally, the J measure for the rule $A_1, A_2 \Rightarrow^{V,T} B$ can be stated as $J(A_1 \Rightarrow^V A_2; B_T)$, where $P(B_T)$ denotes the probability of at least one B occurring in a randomly chosen window of size T in $D(s_2)$, $P(A_1 \Rightarrow^V A_2)$ the probability of association rule $A_1 \Rightarrow^V A_2$ occurring at a random position in $D(s_1)$ (i.e. the relative support of $A_1 \Rightarrow^V A_2$) and $P(B_T \mid A_1 \Rightarrow^V A_2)$ denoting the probability of at least on B occurring in a randomly chosen window of size T in $D(s_2)$, which starts with A_1 in $D(s_1)$, followed by A_2 within V time units (i.e. $F(A_1, A_2, B, V, T)$ divided by the absolute support of $A_1 \Rightarrow^V A_2$).

E. Cyclic and Calendar-Based Association Rules

Another interesting temporal association rule format was proposed in [8]. A rule is called a *cyclic association rule*, if it represents regular cyclic variations over time. An example for this kind of association rules, again from market basket analysis, is the rule {beer} ⇒ {potato crisps}. The support of this rule will probably be relatively low during the whole day and thus not discovered, but in certain regular time intervals, e.g., every day from 7-9PM (and in the corresponding database segment or part of the time series, respectively), it will surely have a support high enough for being discovered. For a more formal definition and the transfer to time series analysis, let us assume that time is given in a fixed unit τ (for instance in hours), and that the i-th time interval, $i \geq 0$, is denoted by τ^i, which corresponds to the time interval $[i \cdot \tau, (i+1) \cdot \tau]$. Let $D(s)^i$ denote the concatenation of the symbols of $D(s)$, that corresponds to τ^i, and refer to it as time segment i. The absolute cyclic support $cycSup_a^i(A \Rightarrow B)$ of an association rule $A \Rightarrow B$ in the time segment i is given by

$$cycSup_a^i(A \Rightarrow B) = \left| \left\{ \; i \;\; | \;\; \left[D(s)^i\right]_i^{i+|AB|-1} = AB \; \right\} \right|,$$

i.e. the number of occurrences of $A \Rightarrow B$ in the time segment i, and the relative cyclic support is defined as

$$cycSup_r^i(A \Rightarrow B) = \frac{cycSup_a^i(A \Rightarrow B)}{|D(s)^i| - |AB| + 1}.$$

The confidence measure for $D(s)^i$ is defined analogously to the confidence measures of the previous temporal association rules. A cycle c is a tuple (l, o), which consists of a length l and an offset o, $0 \leq o \leq l$, both given in time units. An association rule is said to have a cycle $c = (l, o)$, if it holds in every l-th time unit starting with time unit τ_o. An association rule that has a cycle is called *cyclic*. The beer and potato crisps rule for instance would have the two cycles $c_1 = (24, 19)$ and $c_1 = (24, 20)$, that denote "every 24 hours in the 19th hour" and "every 24 hours in the 20th hour", which is is exactly every day's time from 7 till 9PM.

An association rule format, which is quite similar to cyclic association rules, are calendar-based association rules [9]. A *calendar-based association rule* is an association rule, that is valid in certain time intervals, which are specified by a former defined calendar schema. An example for such a schema might be (year, month, day); an example for a special time interval in this calendar schema might be the triple $(*, 11, *)$, where $*$ denotes every arbitrary integer in the domain of the accordant attribute, in this case day and year. $(*, 11, *)$ represents every day in November of every year, which - at least in Germany - is the time with the most traffic jams. As well as in the foregoing approaches, support is calculated by dividing the number of occurrences of a certain association rule by the the maximal possible number of occurrences of a rule of same length. Thus, we omit the formal definitions of calendar-based support, refer to [9] and give one more example for clarification instead: In the calendar-based schema (month, day, hour), the beer and potato crisps rule from above would hold in the time represented by $(*, *, 19) \cup (*, *, 20)$, because this denotes the 19th and 20th hour of every day in every month, what refers to the time from 7 to 9PM.

Obviously, the intersection of correlations, which can be expressed by calendar-based association rules, and of correlations, which can be expressed by cyclic association rules, is very large, thus it is a matter flavor, how these rules are presented.

F. Implementation

The basic idea of our implementation for mining temporal association rules (as mentioned in the previous subsections) in time series is straight forward: first, the time series have to be obtained and processed from a source. In our case, the data is stored in a MySQL database, from which it easily can be retrieved in adequate granularities. After an optional step of storing every time series derived from the database on disk or loading it from there, the time series are discretized by the two methods presented in subsection V-A, which results in three discretized time series representation for each time series: the first discretization is obtained by SAX, the second by the clustering-based approach, where the basic shapes are derived from each time series on their own, and the third is derived by the clustering-based approach, where the basic shapes are derived from all time series together (which makes more sense for the interpretation of inter time serial association rules). Afterwards the discovery of the different kinds of temporal association rules (as presented in subsection V-B) is started for each type independently.

The apriori algorithm, which was presented in [11], is the quasi-standard algorithm for efficiently mining association rules in transactional databases. It bases on a lexicographical ordering of the items in each transaction, but since this ordering would destroy fundamental temporal relations, it cannot be applied directly for the discovery of temporal association rules in a time series database. Thus, for efficiently finding association rules between strings in discretized time series, we can only rely on two basic concepts of this algorithm, which are the monotonicity property and the candidate generation. The monotonicity property applied to time series states, that a string can only be frequent, if every substring is frequent too. Thus we do not have to search for association rules containing every possible string in the discretizations, but only for strings, that consist of smaller frequent strings. As first step

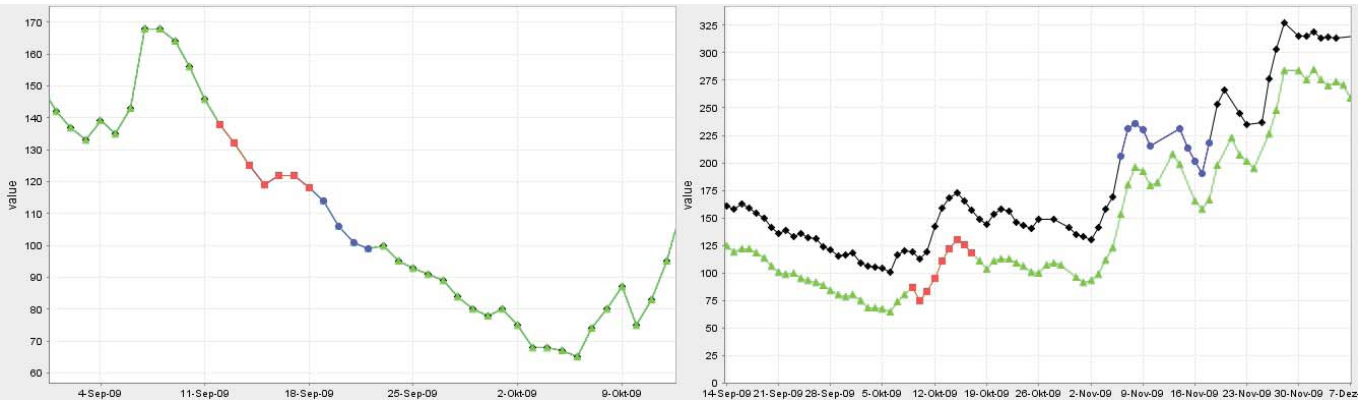

Fig. 3. Example of an intra time serial association rule (left) and an inter time serial association rules (right) of the form $A \Rightarrow^T B$, where A is display red (with rectangles) and B blue (with circles).

in discovering temporal association rules, we are determining every symbol, which is frequent (according to the parameter `minsup`). The concatenations of the pairs of these symbols form the candidate set C_2, from which all frequent strings of length 2 are derived, by checking if support is higher than `minsup`. All frequent strings of length $k-1$ form the set L_{k-1}, from which again the candidate set C_k is generated by adding every string of length k, which can be generated from merging two strings from L_{k-1}, that conform in $k-2$ succeeding characters. L_k again is generated from C_k by checking the `minsup` constraint.

After determining the finite set $\bigcup_k L_k$ of all frequent strings, the temporal association rules can simply be generated by counting their frequencies, as explicated in subsection V-B.

We implemented our system in Java using the Eclipse IDE. Several data mining algorithms in Java are freely available, thus for discretizing we went the easy way and used the SAX implementation from Pavel Senin [40] and the k-means algorithm from Weka [41] for obtaining the clustering-based representations of the time series.

G. Evaluation

This section presents the evaluation of our system for time series analysis on two large datasets containing river level measurements and stock data. The river level measurements consists of 72,000,000 tuples from 394 river stations, which are obtained by the Federal Institute for Nature, Environment and Consumerism of North Rhine-Westphalia (Germany)[1], and the stock data consists of 47,000 tuples of daily closing prices from 32 companies traded at the Nasdaq stock market from 2006-2009, obtained from the Yahoo finance server[2]. All experiments have been carried out on an Intel Core 2 Duo 3.0 GHz desktop PC running Windows Vista with 3 GB of main memory.

We conducted several experiments with varying parameters. In the first setting we tested the runtime for discovering intra time serial association rules versus number and size of time series. As expected, the runtime increased with increasing number and size of time series. We conducted the same series of experiments with four different pairs of `minsup` and `minconf` values, written as $(\texttt{minsup}, \texttt{minconf})$, namely with $(2\%, 50\%)$, $(10\%, 50\%)$, $(10\%, 70\%)$ and $(15\%, 70\%)$, with the result, that the pair $(2\%, 50\%)$ yielded the largest number of discovered association rule, followed by $(10\%, 50\%)$, $(10\%, 70\%)$ and $(15\%, 70\%)$ (thus the number of discovered association rules is anti proportional to `minsup`). Setting `minsup` and `minconf` to $(2\%, 50\%)$ and only changing T, it shows that the number of discovered association rules increases with T. Increasing w (the parameter that determines the distance between the first and the second part of the rule) yields a decrease in the number of discovered association rules. The runtime of the discovery of inter time serial association rules is quadratic in the number of time series, because the time series are compared pairwise, which was affirmed by our experiments.

Figure 3 gives an example of an intra and an inter time serial occurrence of an association rule of the form $A \Rightarrow^T B$.

VI. Conclusion and Future Work

In this paper we presented a brief review on the analysis of time series with temporal association rule mining. We gathered some previous approaches and explained in detail how to transfer four intuitive temporal association rule mining approaches (simple temporal association rules as proposed in [6], sequential patterns [7], cyclic [8] and calendar-based association rules [9]) to the analysis of continuous time series. In addition to that, we presented a system that applies these temporal association rule mining techniques to time series analysis (after discretizing the time series with SAX and two versions of the clustering-based approach from [6]), in order to reveal domain-dependent interesting coherences in one time series or between pairs of time series. We evaluated our system on two large datasets containing river levels measurements and stock data, which showed the functionality and the influence of the different parameters.

[1] http://www.lanuv.nrw.de/
[2] http://finance.yahoo.com/

Apart from integrating further approaches for temporal association rules (e.g. interval-based approaches like [31] and [24] after discretizing the time series to an appropriate format as explained in section IV), there are some more points left for future work: First of all, further time series discretization methods have to be tested in the transformation process, for instance deductive ones, since clustering of time series was claimed to be not very meaningful [42] (apart from that we could not find major differences in the results of our clustering- and SAX-based discretizations), and especially dynamic ones, since both the methods we use are rather static in the sense that one symbol has always the same duration. An appropriate dynamic discretization might not only be interesting from the view point of data compression, but also to make basic shapes (and thus the rules) more expressive. Furthermore, an interestingness measure for association rules, which combines support, confidence and a measure, that penalizes the overlapping of occurrences of an association rule, could be very useful in addition to existing interestingness measures like the J measure, since rules with many overlapping parts are less expressive (which is not captured by the existing interestingness measures).

REFERENCES

[1] T. Schlüter and S. Conrad, "An approach for automatic sleep stage scoring and apnea-hypopnea detection," in *Proc. of the 10th IEEE Int. Conf. on Data Mining (ICDM)*, 2010, pp. 1007–1012.

[2] F. Mormann, R. G. Andrzejak, C. E. Elger, and K. Lehnertz, "Seizure prediction: the long and winding road." *Brain*, 2006.

[3] T. Xu, J. Wu, Z. Wu, and Q. Li, "Long-term sunspot number prediction based on EMD analysis and AR model," *Chin. J. Astron. Astrophys.*, vol. 8, pp. 337–342, 2008.

[4] T. Schlüter and S. Conrad, "TEMPUS: A Prototype System for Time Series Analysis and Prediction," in *IADIS European Conf. on Data Mining 2010*. IADIS Press, 2010, pp. 11–18.

[5] G. Cimino, G. D. Duce, L. K. Kadonaga, G. Rotundo, A. Sisani, G. Stabile, B. Tirozzi, and M. Whiticar, "Time series analysis of geological data," *Chemical Geology*, vol. 161, pp. 253 – 270, 1999.

[6] G. Das, K. Lin, H. Mannila, G. Renganathan, and P. Smyth, "Rule discovery from time series," in *Knowledge Discovery and Data Mining*, 1998, pp. 16–22.

[7] R. Agrawal and R. Srikant, "Mining sequential patterns," in *ICDE*. IEEE Computer Society Press, 1995.

[8] B. Ozden, S. Ramaswamy, and A. Silberschatz, "Cyclic association rules," in *ICDE*, 1998, pp. 412–421.

[9] Y. Li, P. Ning, X. S. Wang, and S. Jajodia, "Discovering calendar-based temporal association rules," in *TIME*, 2001, pp. 111–118.

[10] T. Brijs, G. Swinnen, K. Vanhoof, and G. Wets, "Using association rules for product assortment decisions: A case study," in *Knowledge Discovery and Data Mining*, 1999, pp. 254–260.

[11] R. Agrawal, T. Imielinski, and A. N. Swami, "Mining association rules between sets of items in large databases," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, Washington, D.C., 1993, pp. 207–216.

[12] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th Int. Conf. Very Large Data Bases*. Morgan Kaufmann, 1994, pp. 487–499.

[13] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *SIGMOD '00: Proc. of the ACM SIGMOD Int. Conf. on Management of data*. New York, NY, USA: ACM, 2000, pp. 1–12.

[14] J. Hipp, U. Güntzer, and G. Nakhaeizadeh, "Algorithms for association rule mining a general survey and comparison," *SIGKDD Explor. Newsl.*, vol. 2, pp. 58–64, 2000.

[15] G. Goulbourne, F. Coenen, and P. H. Leng, "Algorithms for computing association rules using a partial-support tree," *Knowledge Based Systems*, vol. 13, no. 2-3, pp. 141–149, 2000.

[16] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," in *Proc. of Int. Conf. on Extending Database Technology (EDBT)*, vol. 1057. Springer, 1996, pp. 3–17.

[17] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu, "Prefixspan: Mining sequential patterns by prefix-projected growth," in *ICDE*. IEEE Computer Society, 2001, pp. 215–224.

[18] H. Mannila and H. Toivonen, "Discovering generalized episodes using minimal occurrences," in *KDD*, 1996, pp. 146–151.

[19] H. Mannila, H. Toivonen, and A. Inkeri Verkamo, "Discovery of frequent episodes in event sequences," *Data Min. Knowl. Discov.*, vol. 1, 1997.

[20] S. K. Harms, J. S. Deogun, and T. Tadesse, "Discovering sequential association rules with constraints and time lags in multiple sequences," in *Proc. of the 13th Int. Symp. on Foundations of Intelligent Systems*, ser. ISMIS '02. London, UK, UK: Springer-Verlag, 2002, pp. 432–441.

[21] J. Han, G. Dong, and Y. Yin, "Efficient mining of partial periodic patterns in time series database," in *Proc. ICDE*, 1999, pp. 106–115.

[22] P.-S. Kam and A. Fu, "Discovering temporal patterns for interval-based events," in *Data Warehousing and Knowledge Discovery*, ser. Lecture Notes in Computer Science, Y. Kambayashi, M. Mohania, and A. Tjoa, Eds. Springer Berlin / Heidelberg, 2000, vol. 1874, pp. 317–326.

[23] F. Höppner, "Learning temporal rules from state sequence," in *IJCAI Workshop on Learning from Temporal and Spatial Data*, 2001.

[24] E. Winarko and J. F. Roddick, "Armada - an algorithm for discovering richer relative temporal association rules from interval-based data," *Data Knowl. Eng.*, vol. 63, no. 1, pp. 76–90, 2007.

[25] J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, pp. 832–843, 1983.

[26] J. Lin, E. J. Keogh, S. Lonardi, and B. Y. chi Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *DMKD*, M. J. Zaki and C. C. Aggarwal, Eds. ACM, 2003, pp. 2–11.

[27] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "Segmenting time series: A survey and novel approach," in *Data mining in Time Series Databases. Published by World Scientific*, 1993, pp. 1–22.

[28] J. Molina, J. Garcia, A. Garcia, R. Melo, and L. Correia, "Segmentation and classification of time-series: Real case studies," in *Intelligent Data Engineering and Automated Learning - IDEAL 2009*, ser. Lecture Notes in Computer Science, E. Corchado and H. Yin, Eds. Springer Berlin / Heidelberg, 2009, vol. 5788, pp. 743–750.

[29] F. Höppner, "Time series abstraction methods - a survey," in *Informatik bewegt: Informatik 2002 - 32. Jahrestagung der Gesellschaft für Informatik e.v. (GI)*. GI, 2002, pp. 777–786.

[30] K. Warasup and C. Nukoolkit, "Discovery association rules in time series data," 2008.

[31] F. Höppner, "Learning dependencies in multivariate time series," in *ECAI Workshop on Knowledge Discovery from Temporal- and Spatio-Temporal Data*, 2002.

[32] T. Lindeberg, *Scale-Space Theory in Computer Vision*. Kluwer Academic Publishers, 1994.

[33] C. S. Daw, C. E. A. Finney, and E. R. Tracy, "A review of symbolic analysis of experimental data," *Review of Scientific Instruments*, vol. 74, no. 2, pp. 915–930, 2003.

[34] S. Kotsiantis and D. Kanellopoulos, "Discretization techniques: A recent survey," *GESTS Int. Transactions on Computer Science and Engineering*, vol. 32, no. 1, pp. 47–58, 2006.

[35] J. Lin, E. Keogh, L. Wei, and S. Lonardi, "Experiencing sax: a novel symbolic representation of time series," *Data Mining and Knowledge Discovery*, vol. 15, pp. 107–144, 2007, 10.1007/s10618-007-0064-z.

[36] Eamonn Keogh, "SAX homepage," last visited 02/2011. [Online]. Available: http://www.cs.ucr.edu/~eamonn/SAX.htm

[37] J. Han, *Data Mining: Concepts and Techniques*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005.

[38] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, 1st ed. Addison Wesley, 2005.

[39] P. Smyth and R. M. Goodman, "An information theoretic approach to rule induction from databases," *IEEE Trans. Knowl. Data Eng.*, vol. 4, no. 4, pp. 301–316, 1992.

[40] Pavel Senin, "SAX - jmotif - homepage," last visited 02/2012. [Online]. Available: http://code.google.com/p/jmotif/wiki/SAX

[41] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, 2009.

[42] E. Keogh and J. Lin, "Clustering of time-series subsequences is meaningless: implications for previous and future research," *Knowl. Inf. Syst.*, vol. 8, pp. 154–177, 2005.