Unsupervised pattern mining from symbolic temporal data

Fabian Mörchen
Siemens Corporate Research
755 College Road East
Princeton, NJ 08540
fabian.moerchen@siemens.com

ABSTRACT

We present a unifying view of temporal concepts and data models in order to categorize existing approaches for unsupervised pattern mining from symbolic temporal data. In particular we distinguish time point-based methods and interval-based methods as well as univariate and multivariate methods. The mining paradigms and the robustness of many proposed approaches are compared to aid the selection of the appropriate method for a given problem. For time points, sequential pattern mining algorithms can be used to express equality and order of time points with gaps in multivariate data. For univariate data and limited gaps suffix tree methods are more efficient. Recently, efficient algorithms have been proposed to mine the more general concept of partial order from time points. For time interval data with precise start and end points the relations of Allen can be used to formulate patterns. The recently proposed Time Series Knowledge Representation is more robust on noisy data and offers an alternative semantic that avoids ambiguity and is more expressive. For both pattern languages efficient mining algorithms have been proposed.

1. INTRODUCTION

Multivariate time series and other time related data occur in many areas and mining this data poses interesting challenges. There are many different approaches for temporal pattern mining based on various data models. They are usually designed with a particular application in mind. Many terms are used in different publications with very different meanings, e.g., *events* [69; 40; 56; 99] or *episodes* [69; 22; 49]. We present a unifying view on the temporal concepts that can be discovered by a certain method and a common language for the underlying data models. Existing approaches for unsupervised pattern mining from symbolic temporal data are categorized accordingly.

On a coarse level we distinguish between time point-based methods and interval-based methods. Another important feature is the applicability to univariate or multivariate data. We further discuss the temporal concepts a pattern language can express, the robustness to noise, and the mining paradigm used. The goal is to give researchers a guideline to select the appropriate method for their purpose. If a certain pattern language is more flexible than necessary the mining will be more costly. If the given data is noisy one needs to be aware of the influence on the resulting patterns.

We define common temporal data models in Section 2 and temporal concepts within these models in Section 3 to provide a common language for the later sections. Section 4 lists temporal operators that have been used in temporal data mining to express the different concepts. The relations of the data models with the most important temporal concepts and operators are shown in Figure 1 for time point and Figure 2 for time interval data respectively.

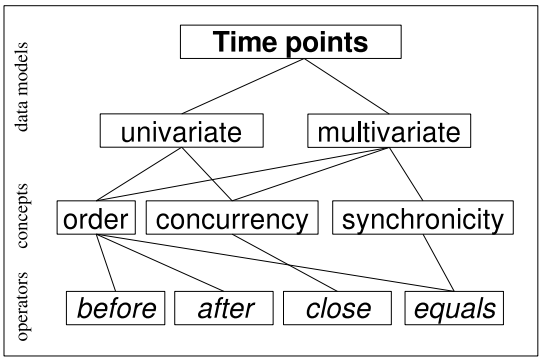

Figure 1: Temporal concepts and operators for time point data models.

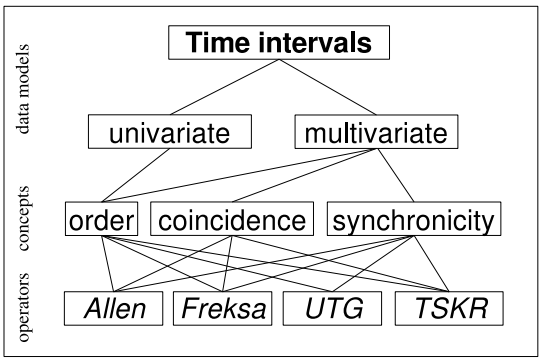

Figure 2: Temporal concepts and operators for time interval data models.

In Section 5 we review methods for unsupervised mining of frequent patterns that relate time point-based properties of a temporal process. Patterns from time interval data are compared in Section 6. The approaches are discussed in Section 7 and Section 8 lists related work. In Section 9

we conclude this survey with some implications for possible future research.

2. TEMPORAL DATA MODELS

This section introduces temporal data models and temporal concepts that are building blocks of many temporal data mining methods. See [74; 77] for formal definitions.

Usually temporal data is represented based on observations at discrete time instants. Often, time is uniformly sampled. If not, there is usually some lower bound for the granularity of time. We refer to this finest level of temporal detail as *time points*[1]. Even if no explicit time values but only an ordering is given, the order can be mapped to integer values. A *time series* is a set of unique time points. A *time sequence* is a multiset of time points, i.e., it can include duplicates. A pair of time points defines a *time interval* starting at the earlier point and ending at the later point inclusively. Two intervals overlap if there is at least one time point that lies within both intervals. An *interval series* is a set of non-overlapping time intervals. In a *contiguous* interval series no gaps between two consecutive intervals are allowed. An *interval sequence* can include overlapping and even equal time intervals. The series data types can be *univariate*, i.e., consisting of a single series or *multivariate* where several series cover the same but not necessarily equally sampled time range.

A *numeric time series* is a time series with numerical values for each time point. This is the data model commonly used in statistics, see Figure 3(a) for an example. A *numeric time sequence*, i.e., a time sequence with numerical values per point, is rarely used. A *symbolic time series* is a time series with nominal values for each point (see Figure 3(b)). It can be obtained from numeric time series by discretization (e.g. [62]). A *symbolic time sequence* has nominal values with possible duplicate time points (see Figure 3(c)). The symbols *A*-*D* are observed at certain time points and two or more symbols can be observed at the same time. A typical example are status events in network monitoring [68].

A *numeric interval series* is a series of non-overlapping time intervals with numerical values for each interval, a *symbolic interval series* has nominal values, respectively. A multivariate symbolic interval sequence is shown in Figure 3(d). The symbols *A*-*C* are observed during certain time intervals. These data models are commonly obtained from univariate or multivariate numeric time series by segmentation (e.g. [57]) and feature extraction (e.g. [51]). Numeric interval data models can be used to mine quantitative association rules, e.g., for stock prices [60].

An *itemset sequence* is a time sequence where an itemset is assigned to each time point. An itemset is a subset of a set of symbols. This data model is used in sequential association rule mining [2], see Figure 3(e) for an example. Univariate symbolic time series and symbolic time sequences are special cases of itemset sequences with itemsets of size one. Multivariate symbolic time series are itemset sequences with itemsets of size equal to the dimensionality of the time series.

For each data model there can be a single long series or sequence or a database of (short) series or sequences, e.g., sets of numeric time series [29] or more typically sets of itemset sequences. A set of short sequences or series can be obtained from a longer one with a sliding window. With overlapping windows this can cause redundancy because many fragments of a frequent patterns will be observed. This can be avoided by restricting patterns such that they start at the first time point in the window [20].

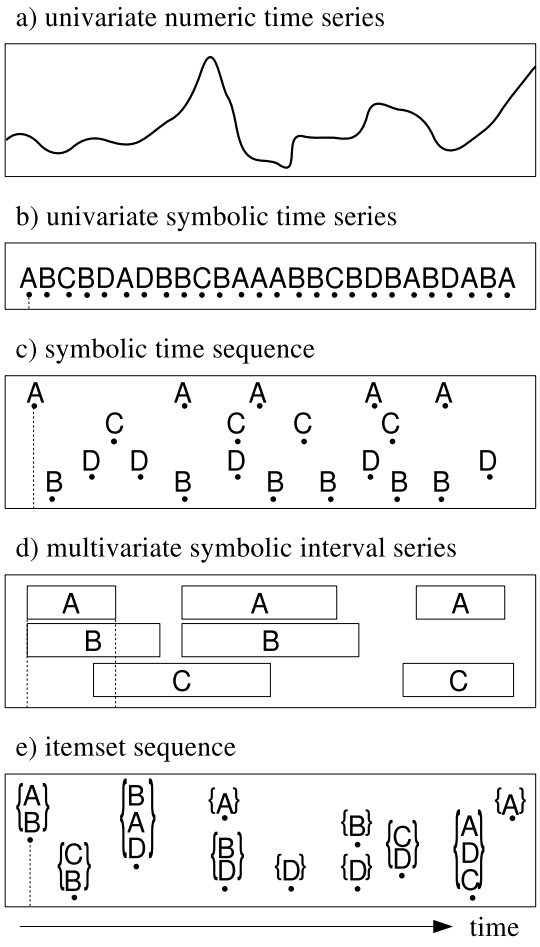

Figure 3: Some typical temporal data models.

3. TEMPORAL CONCEPTS

The concept of *duration* is the persistence or repetition of a property over several time points. Duration is what distinguishes time point from time interval data models. Time points express instantaneous information, e.g., lightning during a thunderstorm. According to the time resolution of our visual perception lightning is perceived as an instantaneous event without duration. The following thunder, however, usually lasts for a few seconds and can thus be described by an interval and has a duration.

The concept of *order* is the sequential occurrence of time points or time intervals. Lightning is always followed by a thunder in a storm.

The concept of *concurrency* is the closeness of two or more temporal events in time in no particular order. In a heavy thunderstorm lightning and gusts of wind occur concurrently, but there is no typical order relation between these

[1]Time points are called chronons in research on temporal databases [16]

events. Concurrency is often used for time points, especially for the data model of time sequences where the exact local order of time points is not necessarily meaningful. With intervals it correspond to the occurrence of two or more intervals within a larger sliding window and no further constraints on their relative position.

The concept of *coincidence* describes the intersection of several intervals. If rain coincides with sunshine, there will often be a rainbow visible. Both rain and sunshine could have started earlier or lasted longer, but only when they coincide the rainbow is visible.

The concept of *synchronicity* is the synchronous occurrence of two temporal events, i.e., equality of time points or time intervals. The flash of lightning and the shrinking of our pupils to adjust for the brightness are synchronous time point events (at the temporal resolution of our perception). For time intervals imagine a sunny spring afternoon with a cloud passing the sun. During this time interval it will be slightly darker and cooler. Both effects set in and end synchronously with the shadow approaching and receding.

4. TEMPORAL OPERATORS

Before we turn our attention to temporal pattern mining we review temporal operators that have been used to relate time points and time intervals expressing the identified temporal concepts.

4.1 Time Point Operators

For two points in time there are three binary operators: *before*, *equals*, and *after*. Both *before* and *after* can be accompanied by a threshold, e.g., after at most (least) k time units. This corresponds to a complete ordering of the time stamps and is used in many temporal data mining algorithms. Sometimes an operator is used to specify that two or more time stamps lie within a time interval expressing concurrency [67; 26; 112].

In [15] an operator called *temporal constraint with granularity* expresses the operators *before*, *after*, and *equals* w.r.t. a granularity of time. The authors argue, that one day is not equivalent to 24h, because the latter could cover parts of two days, an important distinction in some applications. Also, finer granularities do not necessarily have to correspond to larger ones: an hour is always part of a day, but not every hour is part of a business day. Thus, the result of the operator can be undefined. A fuzzy extension for temporal reasoning has been proposed in [28] expressing relations like *much before* or *closely after*.

4.2 Time Interval Operators

4.2.1 Allen's Interval Operators

For the purpose of temporal reasoning Allen formalized temporal logic on intervals by specifying 13 interval relations [4] and showing their completeness. Any two intervals are related by exactly one of the relations. The operators are: *before*, *meets*, *overlaps*, *starts*, *during*, *finishes*, the corresponding inverses *after*, *met by*, *overlapped by*, *started by*, *contains*, *finished by*, and *equals* (see Figure 4).

The relations are commonly used beyond temporal reasoning, e.g., for the formulation of temporal patterns (see Section 6). In [11] a fuzzy extension of Allen's interval relations is proposed by adding a preference degree to each possible relation. For other approaches to fuzzy time interval relations see [17; 79; 82]. In [100] a relaxed version of Allen's relations, called TIME, is described. The strict *meets*, *starts*, and *finishes* operators are relaxed using a threshold (originally proposed in [3]) and intervals far apart are defined to have *no relation*.

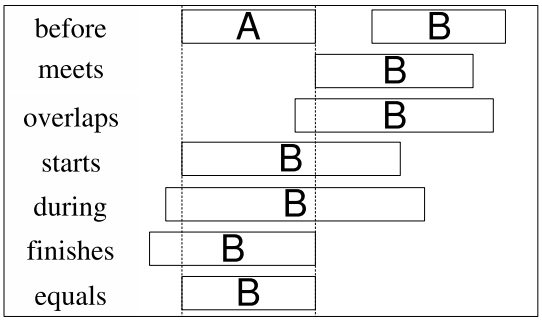

Figure 4: Examples of Allen's interval relations between the intervals A and B. The first six can be inverted.

4.2.2 Freksa's Semi-Interval Operators

Freksa generalized Allen's interval relations by using semi-intervals [32]. There are 10 operators that relate two intervals by using only one boundary of each. The operators *older* with inverse *younger* and *head to head* shown in the first two rows of Figure 5 relate two intervals by their corresponding start points. The operators *survives* with inverse *survived by* and *tail to tail* shown in the next two rows are defined using the end points of each interval. The operators *precedes* with inverse *succeeds* and *born before death* with inverse *died after birth* relate the start point of one interval with the end point of the other. These operators can be combined to form the additional operators: *contemporary* (*dies after birth* or *born before death*), *older & survived by*, *younger & survives*, *older contemporary*, *surviving contemporary*, *survived by contemporary*, and *younger contemporary*.

The generalization to semi-intervals was motivated by the observation that *"in no case, more than two relations between beginnings and endings of events must be known for uniquely identifying the relation between the corresponding events"*. For temporal reasoning the semi-interval representation has the advantage that coarse or incomplete knowledge is represented by simpler formulas instead of long disjunctions. A concept of similarity among the relations is described. Freksa's semi-interval relations were used to mine temporal association rules in [91].

4.2.3 Roddick's Midpoint Interval Operators

Roddick combined Allen's interval relations with the five point-interval relations of [106] considering the relative positions of the interval midpoints [92]. A total of 49 relations is obtained, e.g., nine different versions of *overlaps*. Two examples based on Allen's relation A overlaps B are shown in Figure 6(a). In the first case the midpoints of each interval are within the other interval. In the second case the midpoint of A is before B and the midpoint of B is after A. The two different versions of Allen's relation can thus be interpreted as large or small overlap. The motivation for the new operators is handling data with coarse time stamps and data from streams with arbitrary local order. The authors

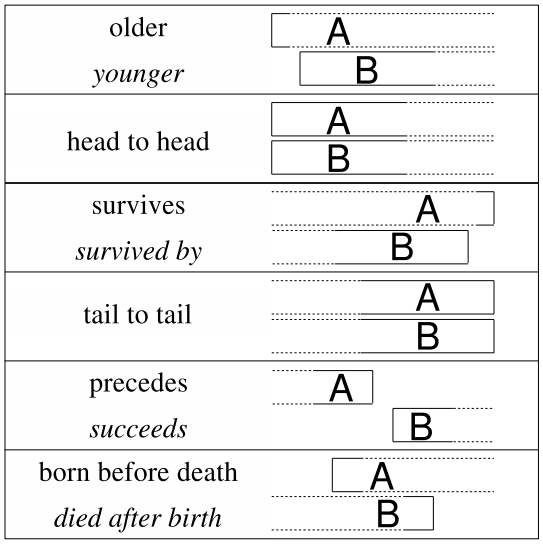

Figure 5: Examples of Freksa's semi-interval relations between the intervals A and B. Inverse operators are shown in italics.

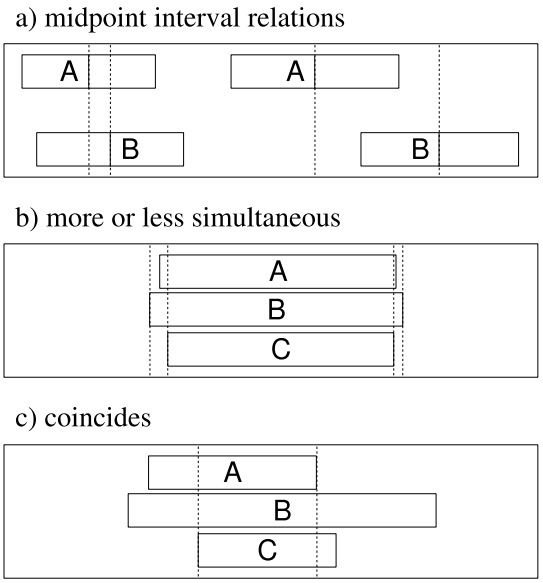

Figure 6: Alternative interval operators.

also describe the relation between the models of Allen and Freksa and the respective extensions to midpoints and/or intervals of equal durations.

4.2.4 Other Interval Operators

The *containment* operator used in [107] is equivalent to the disjunction of Allen's *equals*, *starts*, *during*, and *ends*. The Unification-based Temporal Grammar (UTG) proposed by Ultsch contains an approximate version of Allen's *equals* operator called *more or less simultaneous* [104; 39; 105] (see Figure 6(b)). The start and end points of the intervals are not required to be exactly equal, they only need to be within a small time interval. A further generalization, called *coincides*, was proposed in [75] by dropping the constraints on the boundary points, only requiring some overlap between the intervals (see Figure 6(c)). For two intervals this is equivalent to the disjunction of Allen's *overlaps*, *starts*, *during*, *finishes*, the four corresponding inverses, and *equals*.

4.3 Temporal Logic Operators

Temporal logic operators apply to data given as temporal facts. They evaluate the truth of facts at time points or during time intervals.

In [10] first order Linear Temporal Logic (LTL) is used for planning tasks. LTL is an extension of First Order Logic (FOL) used to specify temporal facts about predicates in a sequence of worlds. The main temporal operators are *until* and *next*, additionally the derived operators *always* and *eventually* are used. An extension with the core operators *since*, *until*, *next*, and *previous*, called First Order Temporal Logic (FOTL), is described in [83]. The operators *always*, *sometimes*, *before*, *after*, and *while* can be derived. In [24; 25] a first order language with the temporal base operators *sometimes*, *always*, *next*, and *until* and some derived operators is used to form temporal classification rules. In [93] three interval predicates are used: *always*, *sometime*, and *true-percentage*. The predicates explicitly include start and end points of an interval and are evaluated on facts instantiated at time points. The truth of a static predicate, e.g., a range condition for a numeric variable, is tested for each time point in an interval, counting how often it is true. For *always* this needs to be true for all time points, *sometimes* corresponds to least one success, and *true-percentage* can be parametrized as a compromise of the other two. The Event logic defined in [99] is a combination of Allen's relations with logical predicates that hold on intervals. The AMA (*and meets and*) logic used in [31] is a subset of the complete Event logic restricted to conjunctions of several meeting conjunctions of positive interval literals.

5. TIME POINT PATTERNS

The most commonly searched temporal concept in univariate symbolic time series is order, i.e., a sub-sequence of symbols occurring sequentially but not necessarily consecutively. In Figure 7(a) the occurrences of the sequence $B \rightarrow C \rightarrow B$ are shown. Similar problems occur in string matching and computational biology (see [41]). The discovery of typical patterns in a single symbolic time series can be done by constructing a suffix tree or a variant thereof. The method described in [109] supports patterns with wild cards and group characters. The wild cards can be used to model (short) gaps in a pattern. A pattern trie with statistics stored in the tree nodes is built to efficiently find the most frequent patterns in a long symbolic time series [109; 110]. The higher the minimum frequency is set, the faster the algorithm runs. The maximum length of a pattern needs to be specified upon tree construction when all sub-sequences of a certain length are extracted from the time series with a sliding window. The efficient calculation of several interestingness measures for such sub-sequences is described in [5]. Over- and underrepresented sequences w.r.t. a binomial or Markov background model can be found efficiently. The interestingness can be visualized with suffix trees using different font sizes and colors for the nodes [6]. In [55] several traversal strategies of suffix tries are compared for efficiency.

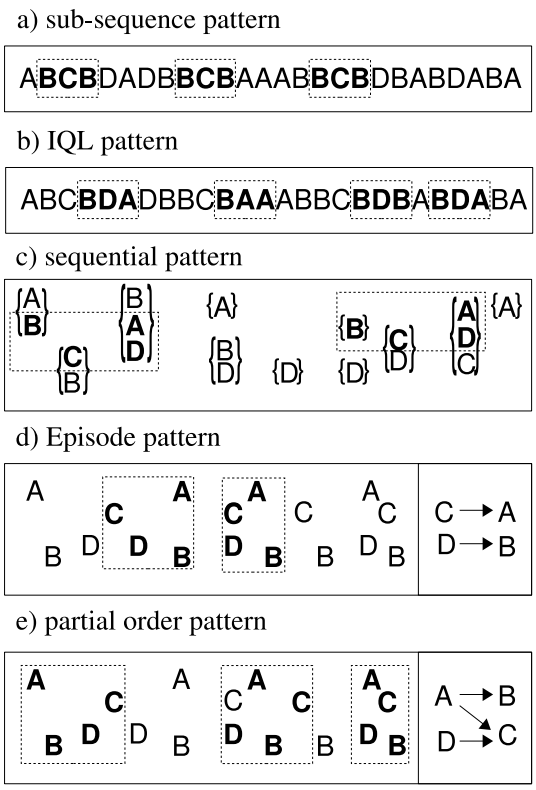

Figure 7: Examples for time point patterns.

The Interagon Query Language (IQL, [54]) is a flexible pattern language for univariate symbolic time series similar to regular expressions [34] including, e.g., negation and disjunctions. It covers the temporal concepts of order, concurrency, and duration. Duration is modelled by allowing repetition of symbols. In Figure 7(b) we show the occurrences of the pattern $B \rightarrow \neg C \rightarrow (A|B)$ (B is followed by something other than C and then by A or B). IQL patterns are mined in [94] using Genetic Programming [58] where each individual is the syntax tree of a candidate pattern. The candidates are evaluated using special hardware to speed up the search.
The search of frequent itemset sub-sequences is commonly called *sequential pattern mining* [2]. One typical application is customers purchasing sets of items at different times, where the seller is interested in products typically bought some time after an initial purchase to give recommendations. The pattern $\{B\} \rightarrow \{C\} \rightarrow \{A, D\}$ if shown in Figure 7(c). Usually a set of itemset sequences is searched but using a sliding window such a set can be obtained from a single long itemset sequence. All approaches for mining itemset sequences can also be applied to symbolic time series and sequences by considering each symbol as an itemset of size one. Sequential patterns can express patterns with gaps of arbitrary length unless explicitly restricted, e.g., by the sliding window length.
The first method to mine itemset sequences was the AprioriAll [2] algorithm based on the Apriori principle of mining small frequent patterns first and combining them to form larger patterns [1]. A faster method is SPADE (Sequential Pattern Discovery using Equivalence classes) [124]. Generalizations of sequential patterns using concept hierarchies and temporal constraints were proposed in [101]. In [90] sequential patterns are combined with multidimensional records associated with the sequential data. They also describe a fast algorithm called GSP (Generalized Sequential Pattern). Recently, an efficient bitmap based algorithm has been proposed [8].
An algorithmically different approach for mining sequential patterns is called *pattern-growth* [44] using, e.g., the PrefixSpan [87] algorithm. The pattern-growth approach uses a depth-first search that avoids the generation and testing of candidate patterns. Another advantage of this method is the possibility to push user defined constraints [34] deep into the mining algorithm, pruning the search space as early as possible [86; 88]. The SPAM (Sequential Pattern Mining, [9]) approach is also a depth-first search using a memory resident bitmap representation to achieve a high speedup compared to SPADE in general and to PrefixSpan for large data sets. In [34] regular expressions constraints are used to guide the search.
The number of reported sequential pattern patterns can be reduced by restricting the search to closed sequential patterns, i.e., patterns with no super-patterns of the same frequency. The CloSpan [115] algorithm uses the pattern growth paradigm to mine closed sequences. To free the user from choosing the minimum frequency [103] propose to mine the top k closed patterns raising the threshold dynamically [103]. In [111] the Bi-Directional Extension checking (BIDE) algorithm is demonstrated to outperform CloSpan. The Apriori and PrefixSpan algorithms were extended to multivariate sequential data in [123]. In [122; 35] sequential patterns with temporal annotations quantifying the duration between successive symbols are mined. Approximate sequential patterns are defined in [59]. For more publications and more details on the mining algorithm see [126].
Sequential patterns are capable of expressing order and synchronicity. The Episode patterns of [69; 68] are more general, because they can also express concurrency. Serial Episodes express an order of time points. The length of the pattern is restricted by a maximum temporal distance between the first and the last symbol. In parallel Episodes the symbols of a pattern can occur in any order within the window. In partially ordered Episodes both concepts are mixed, e.g., an order of parallel Episodes or concurrent serial Episodes. The parallel combination of two serial Episodes is shown in Figure 7(d) with a graph based representation of the order relations on the right. An Apriori algorithm is used for mining Episodes. The type of patterns and the width of a sliding window need to be specified by the user. Typical applications for Episodes include analysis of telecommunication data or web server logs.
Different methods have bee used for counting the occurrences of Episodes. The WINEPI [69] method uses the relative frequency, i.e., the number of windows with the pattern divided by the total number of windows. The MINEPI [67] method uses the concept of minimal occurrences, i.e., a window around the pattern that does not contain sub windows with the same pattern. In [12; 18] these approaches are criticized for the need of a fixed maximum window length. Instead they constrain the maximum distance between successive symbols in a pattern. This way an Episode pattern can be arbitrarily long. The frequency counting is performed with respect to the pattern length. [70] show that

the method of [18] is incomplete and propose the WinMiner algorithm [70].

An extension of Episodes expressed as temporal logic programs with operators like *until* or *since* is described in [83]. In [66] Episodes are mined with an generative approach. [46] mines closed sets of Episodes. Efficient detection of occurrences of known Episodes in data can be done using directed acyclic sub-sequence graphs [102]. In [42] the significance of Episodes for Bernoulli and Markov background models is calculated. A formal connection between Episodes and discrete Hidden Markov Models is shown in [61] and utilized in discovering frequent Episodes.

In [71] Episodes are combined with a subset of Allen's relations to express more temporal relations among Episodes than concurrency. Within each frequent Episode sub patterns are searched such that between the time intervals defined by the start and end point of each sub pattern the during, overlaps, or meets relation holds.

In [89] it is shown that Episodes cannot express all possible partial orders. An example from [89] is shown in Figure 7(e) on the right hand side. In [19] such partial orders were mined from of itemset sequences as follows: First, an algorithm for mining closed sequential patterns was applied [115; 111; 103]. The authors note that pairs of a closed sequential pattern and the corresponding list of sequences in which they appear, unlike for closed itemsets [85], do not form a Galois lattice [33]: There can be sequential patterns occurring in the same sequences (or windows) that are not contained in one another. In the next step pairs consisting of a set of closed sequential patterns and a set of transactions in which all these patterns occur are formed. The algorithm ensures maximality of these pairs in the sense that no additional sequence occurs in all the transactions and there is no additional transaction in which all the sequences occur. These maximal pairs form a Galois lattice and each of them is converted into a partial order resulting in a lattice of closed partial orders. In [74] it is noted that forming such pairs is an instance of the frequent itemset problem and the CHARM [125] algorithm is used to mine them. [89] proposes a more efficient pattern-growth algorithm to directly mine closed partial order patterns.

The approaches for unsupervised pattern discovery in symbolic time series and sequences are summarized in Table 1 roughly ordered by increasing expressivity of the pattern language. For several contributing authors the earlier publication is listed. All methods for symbolic time sequences can also be applied to univariate symbolic time series. An extension to multivariate symbolic time series is usually possible as well.

The first two methods are targeted at univariate data. Sequences are designed to find order relations, while IQL is very flexible and can express all temporal concepts applicable to univariate time series. Sequential patterns can be mined in itemset time sequences, symbolic time sequences, and multivariate symbolic time series. By allowing several symbols in the pattern for a single time point they can express synchronicity. Episodes additionally express the concept of concurrency, because in parallel Episodes no order constraint is placed on the involved symbols. Episodes are typically mined from symbolic time sequences, but an extension to itemsets expressing synchronicity is possible. In [19] such an extension is mentioned for partial orders.

6. TIME INTERVAL PATTERNS

The interval data model implicitly covers the concept of temporal duration, but the length of the interval is not always used in selecting or expressing the patterns.

The search for containments of intervals in a multivariate symbolic interval series or sequences is described in [107]. A containment pattern expresses the temporal concept of coincidence. A containment lattice is constructed from the intervals and implication rules are generated. The duration of the intervals is not used. Two examples of *A contains B contains C* are shown in Figure 8(a).

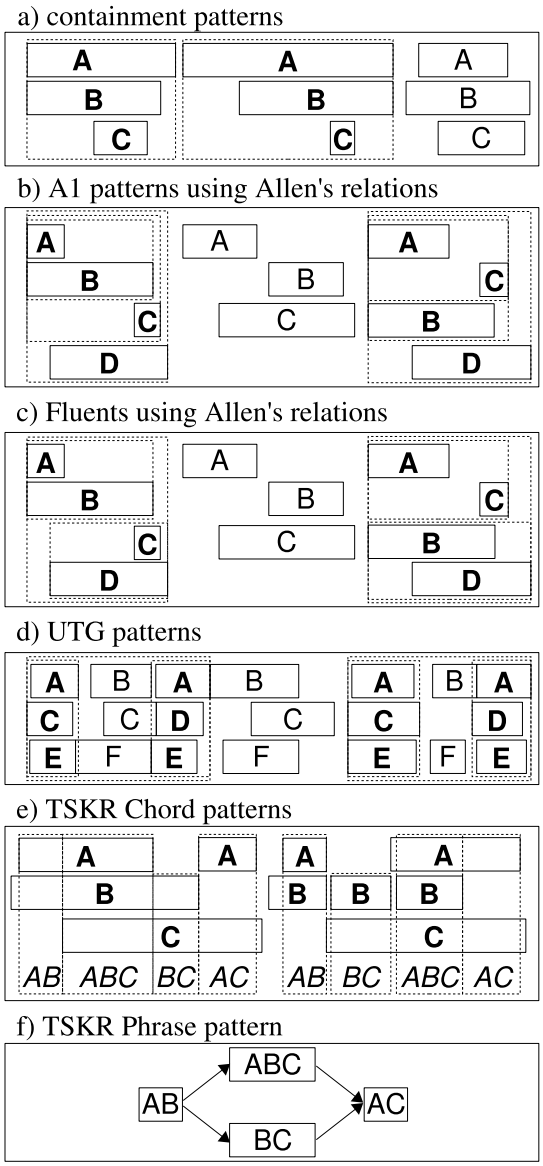

Figure 8: Examples for time interval patterns.

In [56] interval patterns are composed with Allen's relations. The patterns can express the temporal concepts of coincidence, synchronicity, and order. The duration of an interval is not explicitly used. The patterns are mined from a set of interval sequences with an Apriori algorithm. The support of a pattern is determined by counting the sequences in

Author(s)/Year	Method/Keyword	Data model			Temporal concepts			
		univariate symbolic time series	multivariate symbolic time series	symbolic time sequence	duration	order	concurrency	synchronicity
Vilo (1998) [109]	Sequences	×				×		
Saetrom & Hetland (2003) [94]	IQL	×			×	×	×	
Agrawal & Srikant (1995) [2]	Sequential patterns	×	×	×		×		×
Mannila et al. (1995) [69]	Episodes	×	×	×		×	×	
Casas-Garriga (2005) [19]	Partial order	×	×	×		×	×	×

Table 1: Categorization of unsupervised frequent pattern mining methods based on time point data models

which it appears. This can easily be generalized to counting the occurrences within sliding windows of a single long interval sequence. The search space is restricted to right concatenations of intervals to existing patterns, so-called A1 patterns, and by a threshold for the maximum length of a pattern. In Figure 8(b) two patterns are shown that could be described as *(((A starts B) overlaps C) overlaps D)* as indicated by the nested dashed boxes in the first example. In the second example the boxes correspond to a different possible representation of the same pattern, namely *(((A before C) started by B) overlaps D)*. Note that we exchanged the vertical position of B and C to better nest the boxes, but did not change the positioning of the intervals on the time axis. Both representations would be reported by the algorithm with identical support because they describe the same qualitative pattern.

In [22] a subset of Allen's relations is used to mine association patterns, called Fluents, in multivariate symbolic interval series [22; 23]. Patterns are searched with an Apriori algorithm over sliding windows. A set of interval sequences could be used alternatively. The meets and before relations are merged. Duration of intervals is not modelled. The patterns are restricted to composites of single intervals and/or already found patterns. The significance of such a pair is determined with contingency tables counting co-occurrences. In Figure 8(c) we used the same intervals as in Figure 8(b) and indicated the following two possible representations with dashed boxes: *(A starts B) overlaps (C during D)* and *(A before C) starts (B overlaps D)*. Note that the A1 formulations are also possible with Fluents but not vice versa. Which pattern is reported depends on the pairwise significance scores. Again, several representations could be reported for the same qualitative pattern.

In contrast, the pattern format proposed in [48; 52] offers a unique representation by including the pairwise relations of all intervals according to Allen. The interval constellation from the previous examples in Figure 8(b)-(c) would be described as *A starts B, A overlaps D, A before C, B overlaps D, B overlaps C, C during D*. A single symbolic interval series is mined using a sliding window and an Apriori algorithm. The support is not determined by counting occurrences as in [56; 22] but rather by recording the lifetime of a pattern within the sliding window. The pattern format can of course also be used with a set of interval sequences and occurrence counting. The usage of quantitative attributes, e.g., the length of the intervals or gaps are proposed in [53]. This way duration is expressible in addition to the concepts of coincidence, synchronicity, and order. Further extensions for handling feature ambiguity [50] and rule set condensation [49] are proposed.

Later, a similar solution was independently described in [84]. The pattern format is equivalent to [52] but uses only a subset of Allen's relations. A tree-based enumeration algorithm [13] is used for efficient mining. In [113] the patterns of [52] are mined with a modified sequential pattern algorithm [63].

The TCon method for temporal knowledge conversion presented in [37; 40; 38] does not use Allen's interval relations. They are criticized for the strict conditions relating interval boundaries. The patterns are instead expressed with the Unification-based Temporal Grammar (UTG) [104; 39; 105], a hierarchical pattern language developed for the description of patterns in multivariate time series. So-called Temporal Complex Patterns are constructed via several abstraction levels. The patterns on each level are associated with a linguistic representation in form of temporal grammatical rules to enable the interpretation by experts. First a multivariate symbolic interval series is obtained from numeric time series. Event patterns describe several *more or less simultaneous* intervals and express the concept of synchronicity. Two example of the Events involving the intervals A, C, E and A, D, E, respectively, are shown in Figure 8(d) with the dashed boxes. The duration of Events is annotated in the UTG rule and significance is measured by conditional probability estimates. Events are required to contain one interval from each dimension of the multivariate interval series. Sequence patterns express an ordering of several Events. The larger dashed boxes in Figure 8(d) indicate the Temporal Complex Pattern *A, C, E more or less simultaneous followed by A, D, E more or less simultaneous*. Most steps of TCon are performed manually based on visualizations and statistical summaries of the patterns of the previous levels, no algorithms for the automatic discovery of the temporal concepts of synchronicity and order are described. The temporal concept of coincidence is not expressible by the UTG patterns, because the intervals are required to start and end almost simultaneously. This makes the UTG less expressive than patterns formulated with Allen's relations [74].

The Time Series Knowledge Representation (TSKR) [74; 73;

77] extends the UTG by replacing the temporal concept of synchronicity with the more general coincidence and relaxing the strict order in Sequences to a partial order. Chord patterns describe a time interval where several observed intervals overlap, no conditions are placed on the interval end points. In Figure 8(e) we indicated eight Chord patterns with the dashed boxes. The letter combinations at the bottom indicate which intervals coincide. An important feature of Chords is that sub-intervals of an observed interval are allowed. Different parts of the observed interval B on the left contribute to the three Chords *A coincides with B*; *A, B, C coincide*; and *B coincides with C*. Phrase patterns describe a partial order of several Chords. The two similar Chord sequences of length four in Figure 8(e) as indicated with the outer dashed boxes are summarized by the partial order graph of the Phrase shown in Figure 8(f).

Using partial order and allowing sub-intervals in a pattern makes the TSKR more expressive than the UTG and the proposed pattern formats using Allen's relations [73]. A more flexible pattern language increases the space and calls for efficient mining algorithms. The Time Series Knowledge Mining (TSKM) framework [74; 72; 77] describes methods to obtain TSKR patterns from numeric time series and symbolic interval series. Chords are similar in structure to the well known itemsets [1]. Each symbolic interval corresponds to an item and a Chord to an itemset. Chords can thus be mined efficiently, e.g., with a version of the CHARM [125] algorithm modified to measure support of Chords as the sum of the durations of the occurrences. Phrases express a partial order similar to the time points patterns in Section 5. The mining can be performed in several steps: The interval sequence of Chords is converted to an itemset sequence with one itemset per interval where no Chords change containing all currently active Chords. Next, an algorithm for closed sequence mining, e.g. CloSpan [115], is applied using a windowing of the itemset sequence. Similar to [19] the closed sequences are grouped according to their transaction lists. Each group is then converted to a partial order.

In Table 2 the properties of the described approaches for pattern discovery in interval series and sequences are listed in order of increasing expressivity of the pattern language and earlier publication. All except the second method work on multivariate interval series and interval sequences.

The first approach is limited to a single temporal concept and may only be useful in certain applications. Allen's interval relations provide a higher temporal expressivity of the resulting patterns and rules. In contrast to [48] the methods of [56] and [22] do not model duration. The pattern format of [48] has been further used by other authors [84; 113]. The UTG/TCon also offers duration but not coincidence. The successor TSKR can express all identified temporal concepts.

7. DISCUSSION

When searching for frequent temporal structure in data one can sometimes choose between the time point and the time interval data model. Numerical time series can be converted to symbolic time series or symbolic interval time series by segmentation [60; 52; 57], discretization [108; 62; 27; 76] or clustering [40; 78]. Which one is more appropriate depends on the data. If the temporal process behaves rather smooth and the discretization contains many repetitions of symbols we recommend using intervals because they more compactly represent the data and make mining more efficient.

For time point data the methods from Section 5 can be used to mine frequent patterns. If the data is univariate, only the temporal concept of order is interesting, and patterns with no or just short gaps are sufficient, using the sequence approach will be most efficient. A summary of the data is built as a preprocessing step such that the support of candidate patterns can be determined more quickly than with repeated scans of the input data. A limited amount of noise can be handled by allowing gaps. Regular expression or the IQL language offer more expressive patterns. Even duration can be modeled by repetition. The mining with Genetic Programming can be quite expensive, however, special hardware is used in [94] to evaluate candidate patterns.

For multivariate data sequential patterns should be considered. They can handle gaps more naturally and express order and synchronicity. There are plenty of algorithms to choose from and this is still a area of active research [8].

In applications where the local order of symbols can be disturbed by noise or is not relevant in the first place, Episodes or full partial order patterns should be used. Considering that partial orders are more expressive and that an efficient mining algorithm is available [89] we think that they are preferable to Episodes. More research is needed to transfer the generation of implication rules from frequent patterns or the implementation of user defined constraints [88] from Episodes to full partial order patterns.

For time interval data patterns based on Allen's relations have been most widely used. The relations were originally developed in the context of temporal reasoning [95] where inference about past, present, and future supports applications in planning, understanding, and diagnosis. The input usually consists of exact but incomplete input data and temporal constraints, often expressed by Allen's relations. Typical problems include determining the consistency of the data and answering queries about scenarios satisfying all constraints. But these problems do not occur in the data mining context: almost the complete interval data is given and meaningful and understandable patterns are searched [52]. One may have to cope with some missing data, but more importantly with possibly noisy and incorrect data.

As discussed in detail in [74; 73; 77] there are several problems when using Allen's relations in the context of data mining. First, patterns from noisy interval data expressed with Allen's interval relations are not robust. Several of Allen's relations require the equality of two or more interval end points. Small disturbances in interval end points can create patterns where a very similar relationship between intervals is fragmented into different relations, see Figure 9 for an example.

There have been attempts to relax the strictness of Allen's

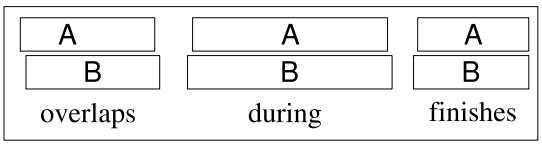

Figure 9: Examples for different patterns according to Allen that are fragments of the same approximate relation *almost equals*.

Author(s)/Year	Method/Keyword	Data model			Temporal concepts			
		univariate numeric interval series	multivariate symbolic interval series	symbolic interval sequence	duration	order	coincidence	synchronicity
Villafane (1999) [107]	Containments		×	×			×	
Kam & Fu (2000) [56]	Allen/A1		×	×		×	×	×
Cohen (2001) [22]	Allen/Fluents		×	×		×	×	×
Guimarães & Ultsch [40]	UTG/TCon		×	×	×	×		×
Höppner (2001) [48]	Allen/pairwise		×	×	×	×	×	×
Mörchen (2006) [74]	TSKR		×	×	×	×	×	×

Table 2: Categorization of unsupervised frequent pattern mining methods based on time interval data models.

interval relations. For fuzzy relations [11] it is not clear how to best determine the membership values for the relations between two given intervals. Threshold can be used to consider temporally close interval boundaries equal [3]. Fragmented patterns are still possible with this approach if noise causes interval boundaries to be shifted around the threshold value.

Further, the patterns are ambiguous because as demonstrated in Figure 10 the same relation of Allen can visually and intuitively represent very different situations. Even more ambiguous is the compact representation of patterns from [56; 22], because as described in Section 6 several different descriptions are valid for the exact same pattern.

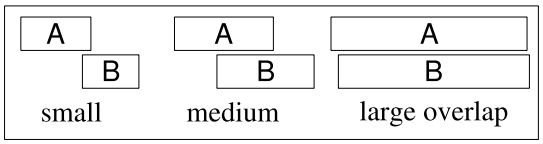

Figure 10: Three instances of Allen's *overlaps* relation with large quantitative differences.

Ambiguity could be reduced by splitting patterns with potential high variability into several different patterns, e.g. using the mid points [92]. But using 49 instead of 13 relations with many additional conditions requiring equality of time points will in turn increase effects of pattern fragmentation.

The UTG was the first method that departed from Allen's relations when forming patterns from intervals. Instead of relying on techniques designed for a different purpose, the UTG was designed from scratch for unsupervised pattern mining involving time intervals. The UTG Event patterns are more robust than Allen's relations, by matching interval endpoints with a threshold. The UTG cannot, however, express all patterns that can be formulated with Allen, because Events cannot express overlapping intervals in general and the number of intervals in an Event is restricted [74].

An important feature of the UTG is the separation of the temporal concepts over several hierarchical levels. Using this divide and conquer strategy reduces the search space for the separate algorithms and offers unique possibilities in relevance feedback during the knowledge discovery process and in the interpretation of the results [105]. An expert can focus on particularly interesting rules and discard valid but known rules before the next level constructs are searched. After obtaining the final results an expert can zoom into each rule to learn about how it is composed and what its meaning and consequences might be supporting the zoom, filter, and details on demand paradigm [98].

The TSKR was built upon these ideas inheriting the advantages while fixing the identified discrepancies. As shown in [74; 77] it is more expressive than the UTG and the proposed pattern formats using Allen's relations. In contrast to Allen's relations the Chord and Phrase patterns of the TSKR are designed to be easily understandable and avoid ambiguities. A TSKR pattern always describes similar situation in the data and can be described with textual rules, with example instances from the input data, or with a visualization of the partial order graph in the Phrase (see Figure 8(f) abstracting over quantitative aspects of the particular observations. The TSKR operator *coincides* is extremely robust against noise in the interval boundaries because it only considers the intersection of all participating intervals, any interval can individually be stretched to infinity without changing the pattern at all.

All described interval pattern languages may suffer from frequent short interruptions of otherwise long intervals caused by noise. Smoothing the original numerical data, if available, can reduce such effects. The interval data can also be filtered [74](Chapter 8.3.5) to create longer intervals where a certain property is almost always observed corresponding to the *true-percentage* operator of [93].

8. RELATED WORK

The unsupervised search for frequent temporal patterns can easily lead to result sets that are too large to analyze. Often, the user has some idea of the type or structure of the patterns he wishes to find. User defined constraints in general and temporal constraints in particular are important tools to guide the search in these cases. Limits on the minimum or maximum time gap between adjacent elements in a sequential pattern can be easily integrated into Apriori-based mining [101]. In [86; 88] it is shown how the temporal con-

straints on the length, duration, and gaps can be efficiently mined with a prefix-growth algorithm.

In addition to the temporal concepts used above, time point data can be mined for periodicity [43]. Given a period length, patterns are mined with an Apriori algorithm. For a period k the patterns consist of k positions filled with symbols or wild cards and cover the temporal concepts of order and periodicity. The period is mined along with the patterns [119]. The method allows missing occurrences as well as shifted periods between segments of occurrence. In subsequent work surprising periodic patterns are mined using an information measure rather than support [120; 121]. Recently, extensions for more robust pattern matching allowing random replacements [117] and meta patterns [118] have been proposed. In [14] the Fast Fourier Transform (FFT, [97]) of binary vectors for each symbol is used to obtain candidates for the algorithm of [43]. In contrast [30] uses a single pass of the data to mine periodic patterns with a similar method. An incremental version of [43] is proposed in [7]. A different approach to periodicity mining based on inter arrival times of a symbol is presented in [64]. Mining candidate periods before association rules is fast, while the opposite is more robust, because random occurrences of a pattern are less likely than those of a single symbol. Periodical patterns are typically searched in retail data, where discovered periodicity can be used for supply chain management or advertisement.

Many of the above described frequent temporal patterns can be used to generate implication rules similar to association rules [69; 107; 22; 48; 94; 71]. Often, only implications forward in time are used to ensure the use of the rule for prediction. Other method directly mine implication rules. In [26] the sequential implication rules of two symbols are mined in univariate symbolic time series. In [81] this is extended to multivariate time series and concurrency. The search space is systematically searched in a general to specific manner. Possible performance problems can be overcome by parallel implementations [80]. In [47] the author presents a method for directly mining Episode rules from multivariate symbolic time series or symbolic time sequences [47; 45]. The antecedent and consequent part of the rule must be separated by a time lag. In addition to order and concurrency, synchronicity is supported by creating new symbols for synchronous pairs.

In [60] association rules are mined from interval data. Each interval is associated with several numeric or symbolic features and a single symbolic target attribute. Association rules on pairs of adjacent intervals predicting the target attribute are mined using the Info-Fuzzy Network (IFN, [65]). Duration can be expressed by using it as a numerical feature for the intervals.

9. CONCLUSION

We reviewed pattern languages and algorithms for unsupervised mining of symbolic temporal data. Elementary temporal concepts and data models were defined to implement the comparison. In Section 7 we discussed the merits of different methods trying to help researchers in selecting the appropriate method. For time points the concept of partial order has recently drawn much attention [66; 36; 18; 89]. Interesting further research direction include transferring mining techniques from sequences to partial orders, e.g., rule generation, clustering [116], or indexing [21]. For interval data Allen's relation have been omnipresent. The recently proposed TSKR offers alternative semantics for interval patterns that are more expressive and more robust. One way to increase the expressivity of patterns using Allen's relations would be to allow patterns with only partially related intervals. In [96] a temporal reasoning framework based on time points and time intervals is described and applied to medical domains [114] using domain knowledge. To the best of our knowledge there has been no data mining approach to unsupervised pattern mining for data with time points and time intervals as may be appropriate for applications like fault monitoring where single events and states with durations can be observed. The five point-interval relations of [106] as used in [92] could be used to formulate patterns.

10. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In P. Buneman and S. Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of data*, pages 207–216. ACM Press, 1993.

[2] R. Agrawal and R. Srikant. Mining sequential patterns. In P. S. Yu and A. S. P. Chen, editors, *Proceedings of the 11th International Conference on Data Engineering (ICDE'95)*, pages 3–14. IEEE Press, 1995.

[3] M. Aiello, C. Monz, L. Todoran, and M. Worring. Document understanding for a broad class of documents. *International Journal on Document Analysis and Recognition*, 5(1):1–16, 2002.

[4] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[5] A. Apostolico, M. E. Bock, S. Lonardi, and X. Xu. Efficient detection of unusual words. *Journal of Computational Biology*, 7(1-2):71–94, 2000.

[6] A. Apostolico, F. Gong, and S. Lonardi. Verbumculus and the discovery of unusual words. *Journal of Computer Science and Technology*, 19(1):22–41, 2003.

[7] W. G. Aref, M. G. Elfeky, and A. K. Elmagarmid. Incremental, online, and merge mining of partial periodic patterns in time-series databases. *IEEE Transactions on Knowledge and Data Engineering*, 16(3):332–342, 2004.

[8] S. Aseervatham, A. Osmani, and E. Viennet. bitSPADE: A lattice-based sequential pattern mining algorithm using bitmap representation. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM'06)*, 2006.

[9] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu. Sequential pattern mining using a bitmap representation. In D. Hand, D. Keim, and R. Ng, editors, *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)*, pages 429–435. ACM Press, 2002.

[10] F. Bacchus and F. Kabanza. Using temporal logics to express search control knowledge for planning. *Artificial Intelligence*, 16(1-2):123–191, 2000.

[11] S. Badaloni and M. Giacomin. A fuzzy extension of Allen's interval algebra. In E. Lamma and P. Mello, editors, *AI*IA99: Advances in Artificial Intelligence*, pages 155–165. Springer, 2000.

[12] J. Baixeries, G. Casas-Garriga, and J. L. Balcazar. Mining unbounded episodes from sequential data. Technical Report NC-TR-01-091, NeuroCOLT, Royal Holloway University of London, UK, 2001.

[13] R. J. Bayardo. Efficiently mining long patterns from databases. In A. Tiwary and M. Franklin, editors, *Proceedings of the 17th ACM SIGMOD symposium on Principles of database systems (PODS'98)*, pages 85–93. ACM Press, 1998.

[14] C. Berberidis, I. Vlahavas, W. G. Aref, M. Atallah, and A. K. Elmagarmid. On the discovery of weak periodicities in large time series. In T. Elomaa, H. Mannila, and H. Toivonen, editors, *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'02)*, pages 51–61. Springer, 2002.

[15] C. Bettini, X. Sean Wang, S. Jajodia, and J.-L. Lin. Discovering frequent event patterns with multiple granularities in time sequences. *IEEE Transactions on Knowledge and Data Engineering*, 10(2):222–237, 1998.

[16] M. H. Böhlen, R. Busatto, and C. S. Jensen. Point-versus interval-based temporal data models. In *Proceedings of the 14th International Conference on Data Engineering (ICDE'98)*, pages 192–200. IEEE Press, 1998.

[17] M. Calin and D. Galea. A fuzzy relation for comparing intervals. In B. Reusch, editor, *Proceedings of the 7th Fuzzy Days on Computational Intelligence, Theory and Applications*, pages 904–916. Springer, 2001.

[18] G. Casas-Garriga. Discovering unbounded episodes in sequential data. In N. Lavrac, D. Gamberger, H. Blockeel, and L. Todorovski, editors, *Proceedings of the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'03)*, pages 83–94. Springer, 2003.

[19] G. Casas-Garriga. Summarizing sequential data with closed partial orders. In H. Kargupta, J. Srivastava, C. Kamath, and A. Goodman, editors, *Proceedings of the 5th SIAM International Conference on Data Mining (SDM'05)*, pages 380–391. SIAM, 2005.

[20] G. Chen, X. Wu, and X. Zhu. Mining sequential patterns across data streams. Technical Report CS-05-04, University of Vermont, Burlington, VT, USA, 2005.

[21] H. Cheng, X. Yan, and J. Han. SeqIndex: Indexing sequences by sequential pattern analysis. In H. Kargupta, J. Srivastava, C. Kamath, and A. Goodman, editors, *Proceedings of the 5th SIAM International Conference on Data Mining (SDM'05)*, pages 84–93. SIAM, 2005.

[22] P. R. Cohen. Fluent learning: Elucidating the structure of episodes. In F. Hoffmann, D. Hand, N. Adams, D. Fisher, and G. Guimarães, editors, *Proceedings of the 4th International Conference in Intelligent Data Analysis (IDA'01)*, pages 268–277. Springer, 2001.

[23] P. R. Cohen, C. Sutton, and B. Burns. Learning effects of robot actions using temporal associations. In *Proceedings of the 2nd International Conference on Development and Learning*, pages 96–101. IEEE Press, 2002.

[24] P. Cotofrei and K. Stoffel. Rule extraction from time series databases using classification trees. In *Proceedings of the 20th IASTED Conference on Applied Informatics*, pages 327–332. ACTA Press, 2002.

[25] P. Cotofrei and K. Stoffel. First-order logic based formalism for temporal data mining. In T. Lin, S. Ohsuga, C.-J. Liau, X. Hu, and S. Tsumoto, editors, *Foundations of Data Mining and Knowledge Discovery*, pages 193–218. Springer, 2005.

[26] G. Das, K.-I. Lin, H. Mannila, G. Renganathan, and P. Smyth. Rule discovery from time series. In R. Agrawal, P. E. Stolorz, and G. Piatetsky-Shapiro, editors, *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pages 16–22. AAAI Press, 1998.

[27] C. S. Daw, C. E. A. Finney, and E. R. Tracy. A review of symbolic analysis of experimental data. *Review of Scientific Instruments*, 74(2):916–930, 2003.

[28] D. DuBois and H. Prade. Processing fuzzy temporal knowledge. *IEEE Transactions on Systems, Man and Cybernetics*, 19(4):729–744, 1989.

[29] A. Dugarjapov and G. Lausen. Mining sets of time series: Description of time points. In M. Schwaiger and O. Opitz, editors, *Proceedings of the 25th Annual Conference of the German Classification Society (GfKl'01)*, pages 41–49. Springer, 2002.

[30] M. G. Elfeky, W. G. Aref, and A. K. Elmagarmid. Using convolution to mine obscure periodic patterns in one pass. In E. Bertino, S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Böhm, and E. Ferrari, editors, *Proceedings of the 9th International Conference on Extending Database Technology (EDBT'04)*, pages 605–620. Springer, 2004.

[31] A. Fern, R. Givan, and J. M. Siskind. Specific-to-general learning for temporal events with application to video event recognition. *Journal of Artificial Intelligence Research*, 17:379–449, 2002.

[32] C. Freksa. Temporal reasoning based on semi-intervals. *Artificial Intelligence*, 54(1):199–227, 1992.

[33] B. Ganter and R. Wille. *Formal Concept Analysis. Mathematical Foundations*. Springer, 1998.

[34] M. N. Garofalakis, R. Rastogi, and K. Shim. SPIRIT: Sequential pattern mining with regular expression constraints. In M. P. Atkinson, M. E. Orlowska, P. Valduriez, S. B. Zdonik, and M. L. Brodie, editors, *Proceedings of the 25th International Conference on Very*

Large Data Bases (VLDB'99), pages 223–234. Morgan Kaufmann, 1999.

[35] F. Gianotti, M. Nanni, and D. Pedreschi. Efficient mining of temporally annotated sequences. In J. Ghosh, D. Lambert, D. B. Skillicorn, and J. Srivastava, editors, *Proceedings of the 6th SIAM International Conference on Data Mining (SDM'06)*, pages 346–357. SIAM, 2006.

[36] A. Gionis, T. Kujala, and H. Mannila. Fragments of orders. In L. Getoor, T. E. Senator, P. Domingos, and C. Faloutsos, editors, *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'03)*, pages 129–136. ACM Press, 2003.

[37] G. Guimarães. *Eine Methode zur Entdeckung von komplexen Mustern in Zeitreihen mit Neuronalen Netzen und deren Überführung in eine symbolische Wissensrepräsentation*. PhD thesis, Philipps-University Marburg, Germany, 1998. German.

[38] G. Guimarães, J. Peter, T. Penzel, and A. Ultsch. A method for automated temporal knowledge acquisition applied to sleep-related breathing disorders. *Artificial Intelligence in Medicine*, 23(3):211–237, 2001.

[39] G. Guimarães and A. Ultsch. A symbolic representation for pattern in time series using definitive clause grammars. In R. Klar and O. Opitz, editors, *Proceedings of the 20th Annual Conference of the German Classification Society (GfKl'96)*, pages 105–111. Springer, 1997.

[40] G. Guimarães and A. Ultsch. A method for temporal knowledge conversion. In D. J. Hand, J. N. Kok, and M. R. Berthold, editors, *Proceedings of the 3rd International Conference in Intelligent Data Analysis (IDA'99)*, pages 369–380. Springer, 1999.

[41] D. Gusfield. *Algorithms on strings, trees, and sequences: computer science and computational biology*. Cambridge University Press, 1997.

[42] R. Gwadera, M. J. Atallah, and W. Szpankowski. Markov models for identification of significant episodes. In H. Kargupta, J. Srivastava, C. Kamath, and A. Goodman, editors, *Proceedings of the 5th SIAM International Conference on Data Mining (SDM'05)*. SIAM, 2005.

[43] J. Han, W. Gong, and Y. Yin. Mining segment-wise periodic patterns in time-related databases. In R. Agrawal, P. E. Stolorz, and G. Piatetsky-Shapiro, editors, *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pages 214–218. AAAI Press, 1998.

[44] J. Han and J. Pei. Pattern growth methods for sequential pattern mining: Principles and extensions. In *Workshop on Temporal Data Mining, 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'01)*. ACM Press, 2001.

[45] S. K. Harms and J. Deogun. Sequential association rule mining with time lags. *Journal of Intelligent Information Systems, Special issue on Data Mining*, 22(1):7–22, 2004.

[46] 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 N. Cercone, T. Y. Lin, and X. Wu, editors, *Proceedings of the 1st IEEE International Conference on Data Mining (ICDM'01)*, pages 603–606. IEEE Press, 2001.

[47] S. K. Harms, J. S. Deogun, and T. Tadesse. Discovering sequential association rules with constraints and time lags in multiple sequences. In *Proceedings of the 13th International Symposium on Foundations of Intelligent Systems*, pages 432–441. Springer, 2002.

[48] F. Höppner. Discovery of temporal patterns - learning rules about the qualitative behaviour of time series. In L. D. Raedt and A. Siebes, editors, *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'01)*, pages 192–203. Springer, 2001.

[49] F. Höppner. Discovery of core episodes from sequences - using generalization for defragmentation of rule sets. In D. Hand, R. Bolton, and N. Adams, editors, *Proceedings ESF Exploratory Workshop on Pattern Detection and Discovery in Data Mining*, pages 199–213. Springer, 2002.

[50] F. Höppner. Handling feature ambiguity in knowledge discovery from time series. In *Proceedings of the 7th International Conference on Discovery Science (DS'02)*, pages 398–405. Springer, 2002.

[51] F. Höppner. Learning dependencies in multivariate time series. In *Workshop on Knowledge Discovery in (Spatio-) Temporal Data at the 15th Eureopean Conference on Artificial Intelligence (ECAI'02)*, pages 25–31, 2002.

[52] F. Höppner. *Knowledge Discovery from Sequential Data*. PhD thesis, Technical University Braunschweig, Germany, 2003.

[53] F. Höppner and F. Klawonn. Finding informative rules in interval sequences. *Intelligent Data Analysis*, 6(3):237–255, 2002.

[54] Interagon. The Interagon Query Language - A reference guide, 2002. Interagon AS, Trondheim, Norway, http://www.interagon.com.

[55] L. Jiang and H. J. Hamilton. Methods for mining frequent sequential patterns. In Y. Xiang and B. Chaibdraa, editors, *Proceedings of the 16th Conference of the Canadian Society for Computational Studies of Intelligence (AI'03)*, pages 486–491. Springer, 2003.

[56] P.-S. Kam and A. W.-C. Fu. Discovering temporal patterns for interval-based events. In Y. Kambayashi, M. K. Mohania, and A. M. Tjoa, editors, *Proceedings of the 2nd International Conference on Data Warehousing and Knowledge Discovery (DaWaK'00)*, pages 317–326. Springer, 2000.

[57] E. Keogh, S. Chu, D. Hart, and M. Pazzani. Segmenting time series: A survey and novel approach. In M. Last, A. Kandel, and H. Bunke, editors, *Data Mining In Time Series Databases*, chapter 1, pages 1–22. World Scientific, Singapore, 2004.

[58] J. R. Koza. Genetic programming. In J. G. Williams and A. Kent, editors, *Encyclopedia of Computer Science and Technology*, volume 39, pages 29–43. Marcel-Dekker, 1998.

[59] H.-C. Kum, J. Pei, W. Wang, and D. Duncan. ApproxMAP: Approximate mining of consensus sequential patterns. In D. Barbará and C. Kamath, editors, *Proceedings of the 3rd SIAM International Conference on Data Mining (SDM'03)*, pages 311–315. SIAM, 2003.

[60] M. Last, Y. Klein, and A. Kandel. Knowledge discovery in time series databases. *IEEE Transactions on Systems, Man, and Cybernetics*, 31(1):160–169, 2001.

[61] S. Laxman, P. Sastry, and K. Unnikrishnan. Discovering frequent episodes and learning hidden markov models: A formal connection. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1505–1517, 2005.

[62] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 2003 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 2–11. ACM Press, 2003.

[63] M.-Y. Lin and S.-Y. Lee. Fast discovery of sequential patterns by memory indexing. In Y. Kambayashi, W. Winiwarter, and M. Arikawa, editors, *Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery (DaWaK'02)*, pages 150–160. Springer, 2002.

[64] S. Ma and J. L. Hellerstein. Mining partially periodic event patterns with unknown periods. In *Proceedings of the 17th International Conference on Data Engineering (ICDE'01)*, pages 205–214. IEEE Press, 2001.

[65] O. Maimon and M. Last. *Knowledge Discovery and Data Mining, the Info-Fuzzy Network (IFN) Methodology*. Kluwer, 2000.

[66] H. Mannila and C. Meek. Global partial orders from sequential data. In R. Ramakrishnan, S. Stolfo, R. Bayardo, and I. Parsa, editors, *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'00)*, pages 161–168. ACM Press, 2000.

[67] H. Mannila and H. Toivonen. Discovering generalized episodes using minimal occurrences. In E. Simoudis, J. Han, and U. M. Fayyad, editors, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96)*, pages 146–151. AAAI Press, 1996.

[68] 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.

[69] H. Mannila, H. Toivonen, and I. Verkamo. Discovery of frequent episodes in event sequences. In U. M. Fayyad and R. Uthurusamy, editors, *Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining (KDD'96)*, pages 210–215. AAAI Press, 1995.

[70] N. Méger and C. Rigotti. Constraint-based mining of episode rules and optimal window sizes. In J.-F. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi, editors, *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'04)*, pages 313–324. Springer, 2004.

[71] C. Mooney and J. F. Roddick. Mining relationships between interacting episodes. In M. W. Berry, U. Dayal, C. Kamath, and D. B. Skillicorn, editors, *Proceedings of the 4th SIAM International Conference on Data Mining (SDM'04)*. SIAM, 2004.

[72] F. Mörchen. Algorithms for time series knowledge mining. In T. Eliassi-Rad, L. H. Ungar, M. Craven, and D. Gunopulos, editors, *Proceedings The Twelveth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 668–673, 2006.

[73] F. Mörchen. A better tool than allen's relations for expressing temporal knowledge in interval data. In T. Li, C. Perng, H. Wang, and C. Domeniconi, editors, *Workshop on Temporal Data Mining at the Twelveth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 25–34, 2006.

[74] F. Mörchen. *Time Series Knowledge Mining*. PhD thesis, Philipps-University Marburg, Germany, 2006.

[75] F. Mörchen and A. Ultsch. Mining hierarchical temporal patterns in multivariate time series. In S. Biundo, T. W. Frühwirth, and G. Palm, editors, *Proceedings of the 27th Annual German Conference in Artificial Intelligence (KI'04)*, pages 127–140. Springer, 2004.

[76] F. Mörchen and A. Ultsch. Optimizing time series discretization for knowledge discovery. In R. Grossman, R. Bayardo, and K. P. Bennett, editors, *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'05)*, pages 660–665. ACM Press, 2005.

[77] F. Mörchen and A. Ultsch. Efficient mining of understandable patterns from multivariate interval time series. *to appear in Data Mining and Knowledge Discovery*, 2007.

[78] F. Mörchen, A. Ultsch, and O. Hoos. Extracting interpretable muscle activation patterns with Time Series Knowledge Mining. *International Journal of Knowledge-Based & Intelligent Engineering Systems*, 9(3):197–208, 2006.

[79] G. Nagypal and B. Motik. A fuzzy model for representing uncertain, subjective and vague temporal knowledge in ontologies. In R. Meersman, Z. Tari, and D. C. Schmidt, editors, *Proceedings International Conference on Ontologies, Databases and Applications of Semantics, (ODBASE'03)*, pages 906–923. Springer, 2003.

[80] T. Oates, M. D. Schmill, and P. R. Cohen. Parallel and distributed search for structure in multivariate time series. Technical Report UM-CS-1996-023, Experimental Knowledge Systems Laboratory, University of Massachusetts Amherst, MA, USA, 1996.

[81] T. Oates, M. D. Schmill, D. Jensen, and P. R. Cohen. A family of algorithms for finding temporal structure in data. In *Proceedings 6th International Workshop on Artificial Intelligence and Statistics*, 1997.

[82] H. J. Ohlbach. Relations between fuzzy time intervals. In *Proceedings 11th International Symposium on Temporal Representation and Reasoning (TIME'04)*, pages 44–51. IEEE Press, 2004.

[83] B. Padmanabhan and A. Tuzhilin. Pattern discovery in temporal databases: a temporal logic approach. In E. Simoudis, J. Han, and U. M. Fayyad, editors, *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96)*, pages 351–354. AAAI Press, 1996.

[84] P. Papaterou, G. Kollios, S. Sclaroff, and D. Gunopoulos. Discovering frequent arrangements of temporal intervals. In *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM'05)*, pages 354–361, 2005.

[85] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *Proceeding of the 7th International Conference on Database Theory (ICDT'99)*, pages 398–416. Springer, 1999.

[86] J. Pei and J. Han. Constrained frequent pattern mining: a pattern-growth view. *ACM SIGKDD Explorations Newsletter*, 4(1):31–39, 2002.

[87] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings of the 17th International Conference on Data Engineering (ICDE'01)*, pages 215–224. IEEE Press, 2001.

[88] J. Pei, J. Han, and W. Wang. Constraint-based sequential pattern mining: the pattern-growth methods. *Journal of Intelligent Information Systems*, 28(2):133–160, 2007.

[89] 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.

[90] H. Pinto, J. Han, J. Pei, K. Wang, Q. Chen, and U. Dayal. Multi-dimensional sequential pattern mining. In *CIKM*, pages 81–88, 2001.

[91] C. Rainsford and J. Roddick. Adding temporal semantics to association rules. In J. M. Zytkow and J. Rauch, editors, *Proceedings of the 3rd European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'99)*, pages 504–509. Springer, 1999.

[92] J. F. Roddick and C. H. Mooney. Linear temporal sequences and their interpretation using midpoint relationships. *IEEE Transactions on Knowledge and Data Engineering*, 17(1):133–135, 2005.

[93] J. J. Rodriguez, C. J. Alonso, and H. Boström. Learning first order logic time series classifiers: Rules and boosting. In D. A. Zighed, H. J. Komorowski, and J. M. Zytkow, editors, *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'00)*, pages 299–308. Springer, 2000.

[94] P. Saetrom and M. L. Hetland. Unsupervised temporal rule mining with genetic programming and specialized hardware. In M. A. Wani, K. J. Cios, and K. Hafeez, editors, *Proceedings of the 2003 International Conference on Machine Learning and Applications (ICMLA'03)*, pages 145–151. CSREA Press, 2003.

[95] E. Schwalb and L. Vila. Temporal constraints: A survey. Technical report, ICS, University of California at Irvine, CA, USA, 1997.

[96] Y. Shahar. A framework for knowledge-based temporal abstraction. *Artificial Intelligence*, 90(1-2):79–133, 1997.

[97] H. Shatkay. The Fourier transform - A primer. Technical Report CS-95-37, Department of Computer Science, Brown University, Providence, RI, USA, 1995.

[98] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*, page 336. IEEE Press, 1996.

[99] J. M. Siskind. Grounding the lexical semantics of verbs in visual perception using force dynamics and event logic. *Journal of Artificial Intelligence Research*, 15:31–90, 2001.

[100] C. Snoek and M. Worring. Multimedia event based video indexing using time intervals. *IEEE Transactions on Multimedia*, 7(4):638–647, 2004.

[101] R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. In P. M. G. Apers, M. Bouzeghoub, and G. Gardarin, editors, *Proceedings of the 5th International Conference on Extending Database Technology (EDBT'96)*, pages 3–17. Springer, 1996.

[102] Z. Troníček. Episode matching. In A. Amir and G. M. Landau, editors, *Proceedings of the 12th Annual Symposium of Combinatorial Pattern Matching*, pages 143–146. Springer, 2001.

[103] 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'03)*, pages 347–354. IEEE Press, 2003.

[104] A. Ultsch. Eine unifikationsbasierte Grammatik zur Beschreibung von komplexen Mustern in multivariaten Zeitreihen. personal notes, 1996. German.

[105] A. Ultsch. Unification-based temporal grammar. Technical Report 37, Department of Mathematics and Computer Science, Philipps-University Marburg, Germany, 2004.

[106] M. Vilain. A system for reasoning about time. In *Proceedings of the 2nd National Conference on Artificial Intelligence (AAAI'82)*, pages 197–201. AAAI Press / MIT Press, 1982.

[107] R. Villafane, K. A. Hua, D. Tran, and B. Maulik. Mining interval time series. In *Proceedings of the 1st International Conference on Data Warehousing and Knowledge Discovery (DaWaK'99)*, pages 318–330. Springer, 1999.

[108] R. Villafane, K. A. Hua, D. Tran, and B. Maulik. Knowledge discovery from series of interval events. *Journal of Intelligent Information Systems*, 15(1):71–89, 2000.

[109] J. Vilo. Discovering frequent patterns from strings. Technical Report C-1998-9, Department of Computer Science, University of Helsinki, Finland, 1998.

[110] J. Vilo. *Pattern Discovery from Biosequences*. PhD thesis, Department of Computer Science, University of Helsinki, Finland, 2002.

[111] J. Wang and J. Han. BIDE: Efficient mining of frequent closed sequences. In *Proceedings of the 20th International Conference on Data Engineering (ICDE'04)*, pages 79–90. IEEE Press, 2004.

[112] G. M. Weiss. Timeweaver: A genetic algorithm for identifying predictive patterns in sequences of events. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*, pages 718–725. Morgan Kaufmann, 1999.

[113] E. Winarko and J. F. Roddick. Armada - an algorithm for discovering richer relative temporal association rules from interval-based data. *Data & Knowledge Engineering*, 2007.

[114] S. Y. and M. M.A. Knowledge-based temporal abstraction in clinical domains. *Artifical Intelligende in Medicine*, 8(3):267–98, 1996.

[115] X. Yan, J. Han, and R. Afshar. CloSpan: Mining closed sequential patterns in large datasets. In D. Barbará and C. Kamath, editors, *Proceedings of the 3rd SIAM International Conference on Data Mining (SDM'03)*, pages 166–177. SIAM, 2003.

[116] J. Yang and W. Wang. CLUSEQ: Efficient and effective sequence clustering. In U. Dayal, K. Ramamritham, and T. M. Vijayaraman, editors, *Proceedings of the 19th International Conference on Data Engineering (ICDE'03)*, pages 101–112. IEEE Press, 2003.

[117] J. Yang, W. Wang, and P. Yu. STAMP: Discovery of statistically important pattern repeats in a long sequence. In D. Barbará and C. Kamath, editors, *Proceedings of the 3rd SIAM International Conference on Data Mining (SDM'03)*, pages 224–238. SIAM, 2003.

[118] J. Yang, W. Wang, and P. Yu. Discovering high order periodic patterns. *Knowledge and Information Systems*, 6(3):243–268, 2004.

[119] J. Yang, W. Wang, and P. S. Yu. Mining asynchronous periodic patterns in time series data. In R. Ramakrishnan, S. Stolfo, R. Bayardo, and I. Parsa, editors, *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'00)*, pages 275–279. ACM Press, 2000.

[120] J. Yang, W. Wang, and P. S. Yu. InfoMiner: Mining surprising periodic patterns. In F. Provost and R. Srikant, editors, *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'01)*, pages 395–400. ACM Press, 2001.

[121] J. Yang, W. Wang, and P. S. Yu. InfoMiner+: Mining partial periodic patterns with gap penalties. In *Proceedings of the 2nd IEEE International Conference on Data Mining (ICDM'02)*, pages 725–728. IEEE Press, 2002.

[122] M. Yoshida, T. Iizuka, H. Shiohara, and M. Ishiguro. Mining sequential patterns including time intervals. In B. V. Dasarathy, editor, *Proceedings of SPIE – Data Mining and Knowledge Discovery: Theory, Tools, and Technology II*, volume 4057, pages 213–220, 2000.

[123] C.-C. Yu and Y.-L. Chen. Mining sequential patterns from multidimensional sequence data. *IEEE Transactions on Knowledge and Data Engineering*, 17(1):136–140, 2005.

[124] M. J. Zaki. Efficient enumeration of frequent sequences. In G. Gardarin, J. C. French, N. Pissinou, K. Makki, and L. Bouganim, editors, *Proceedings of the 7th International Conference on Information and Knowledge Management (CIKM'98)*, pages 68–75. ACM Press, 1998.

[125] M. J. Zaki and C.-J. Hsiao. Efficient algorithms for mining closed itemsets and their lattice structure. *IEEE Transaction on Knowledge and Data Engineering*, 17(4):462–478, 2005.

[126] Q. Zhao and S. Bhowmick. Sequential pattern mining: A survey. Technical report, Nanyang Technichal University, Singapore, 2003.