Algorithms For Time Series Knowledge Mining

Fabian Morchen
Databionic Research Group
Philipps-University Marburg
Hans-Meerwein-Str., Marburg, Germany
fabian.moerchen@siemens.com

ABSTRACT

Temporal patterns composed of symbolic intervals are commonly formulated with Allen's interval relations originating in temporal reasoning. This representation has severe disadvantages for knowledge discovery. The Time Series Knowledge Representation (TSKR) is a new hierarchical language for interval patterns expressing the temporal concepts of coincidence and partial order. We present effective and efficient mining algorithms for such patterns based on itemset techniques. A novel form of search space pruning effectively reduces the size of the mining result to ease interpretation and speed up the algorithms. On a real data set a concise set of TSKR patterns can explain the underlying temporal phenomena, whereas the patterns found with Allen's relations are far more numerous yet only explain fragments of the data.

Categories and Subject Descriptors: I.5 Computing Methodologies: Pattern Recognition **General Terms:** Algorithms **Keywords:** knowledge discovery, time series, interval patterns

1. INTRODUCTION

Symbolic interval time series are an important data format for discovering temporal knowledge that can be easily communicated to human analysts [4, 6, 18, 3, 7, 5, 12]. Numerical time series are often converted to symbolic interval time series by segmentation [7, 5], discretization [18, 11] or clustering [4, 12]. Patterns mined from symbolic interval data can provide explanation for the underlying temporal processes or anomalous behavior.

Unsupervised pattern discovery in interval time series has largely been performed based on Allen's interval relations [1], e.g. in [6, 3, 5]. The relations were originally developed in the context of temporal reasoning. The input usually consists of exact but incomplete data and temporal constraints. Typical problems include determining the consistency of the data and answering queries about scenarios satisfying all constraints (e.g. [14]). In the data mining context, however, possibly noisy and incorrect interval data is given and meaningful and understandable patterns are searched [5].

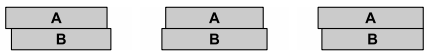
We think that Allen's relations have severe disadvantages when used for pattern discovery from interval time series. As an alternative we proposed the Time Series Knowledge Representation (TSKR) [10], a new hierarchical language for the formulation of temporal knowledge based on interval times series that significantly extends the Unification-based Temporal Grammar (UTG) [16, 17]. We present efficient algorithms for mining patterns expressed with the TSKR using itemset techniques.

KDD'06, August 20–23, 2006, Philadelphia, Pennsylvania, USA.

2. RELATED WORK AND MOTIVATION

The most common tool for expressing temporal patterns from interval data are the 13 interval relations of [1] are: *before*, *meets*, *overlaps*, *starts*, *during*, *finishes*, the corresponding inverses and *equals*. They are commonly used for the formulation of temporal rules involving intervals [6, 3, 5, 9, 13]. Unsupervised rule mining with using Allen's relations is commonly performed with variants of the Apriori algorithm. In [6] and [3] interval patterns are constructed combining two intervals or existing patterns with a single relation. The representation of [5] lists all pairwise interval relations within a pattern. Allen's relations have severe disadvantages when used for pattern discovery from interval time series, as demonstrated with the following examples. For a more detailed analysis see [10].

(1) 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. Figure 1 shows several examples of almost equal intervals.

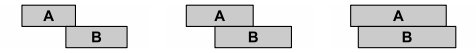


(a) overlaps (b) during (c) finishes

Figure 1: Examples for different patterns according to Allen that are fragments of the same approximate relation *almost equals*.

(2) Patterns expressed with Allen's interval relations are

ambiguous: The same relation of Allen can visually and intuitively represent very different situations. In Figure 2 three very different versions of the *overlaps* relation are shown as an example. Even more ambiguous is the compact representation of patterns from [6, 3], several different descriptions are valid for the exact same pattern (see Figure 3).



(a) small (b) medium (c) large

Figure 2: Intuitively different instances of Allen's *overlaps*.

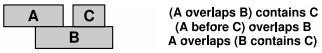


Figure 3: Pattern that can be described by three different compact rules using Allen's relations.

(3) Patterns expressed with Allen's interval relations are not easily comprehensible: The representation of patterns with Allen's relations does not follow the Gricean maxims suggested for the representation of knowledge discovery results to humans [15]. For example, the maxim of quantity is violated. Since the compact format is ambiguous, patterns need to be expressed by listing the pairwise relations of all intervals [5]. This representation grows quickly with the number of intervals.

The UTG was proposed by Ultsch in [16, 17] and realized in [4]. It is a hierarchical pattern language for temporal knowledge discovery. Each level of the hierarchy inhibits an increase in temporal abstraction. The central pattern elements are Events, that describe *more or less simultaneous* intervals, and Sequences, that describe a total order of Events. The UTG is more robust than but not as expressible as Allen's relations [10]. The hierarchical structure of the UTG and the separation of temporal concepts over several mining steps, however, offer unique possibilities in relevance feedback during the knowledge discovery process and in the analysis of the results [17]. The TSKR extends these core ideas achieving higher robustness and expressivity.

3. KNOWLEDGE REPRESENTATION

The TSKR is a hierarchical language for expressing temporal knowledge in interval data. The levels successively describe the temporal concepts of duration, coincidence, and partial order.

DEFINITION 1. *A* symbolic time interval *is a triple* $[\sigma, s, e]$ *with* $\sigma \in \Sigma$, $[s, e] \in \mathbb{T}^2$, $s \leq e$. *The duration of a symbolic interval is* $d([s, e]) = e - s + 1$. *We write* $[\sigma, s, e] \subseteq [\sigma', s', e']$ *if* $s' \leq s$ *and* $e \leq e'$ *with equality iff* $s = s'$ *and* $e = e'$. *A symbolic time interval is* maximal *if* $\forall [\sigma', s', e'] \supset [\sigma, s, e]$ $\sigma' \neq \sigma$. *If* $\{s, ..., e\} \cap \{s', ..., e'\} \neq \emptyset$ *the intervals* $[\sigma, s, e]$ *and* $[\sigma', s', e']$ *overlap.*

DEFINITION 2. *A* symbolic interval series $\mathcal{I}$ *is a set of non-overlapping symbolic time intervals. The duration of* $\mathcal{I}$ *is the sum of the interval durations. A* symbolic interval sequence *is a finite set of symbolic time intervals. An* itemset interval series *is a symbolic interval series where the symbol* σ *is replaces by a subset* $S \subseteq \Sigma$ *of symbols.*

Tones are the basic primitives of the TSKR representing duration. A Tone consists of a label, a symbol, and a symbolic interval series indicating when it is observed. A Tone labeled *temperature high* could be obtained by discretizing a numerical time series with thresholds. Simultaneously occurring Tones form a *Chord*, representing coincidence.

DEFINITION 3. *A* Chord *pattern describes a time interval where* $k > 0$ *Tones coincide. We say the Chord* $c_i \supset c_j$ *is a super-Chord of* c_j *if* c_i *describes the coincidence of a superset of the Tones from* c_j. $c_i \cup c_j$ *is the Chord obtained by merging the Tones,* $|c|$ *is the size of a Chords, i.e. the number of Tones that coincide. The* support $sup_\delta(c)$ *of a Chord* c *is the duration of all maximal observation intervals with duration of at least* δ.

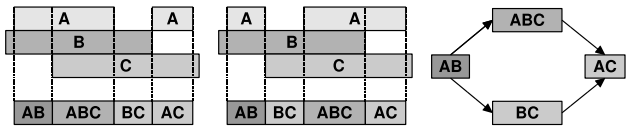
Several Chords are shown in Figure 4(a) in the bottom row describing different coincidences of the Tones A, B, and C in the top rows. The symbolic interval of AC is maximal whereas that of BC is not. A Chord pattern implies all sub-Chords on the same interval. Usually, larger Chords are more interesting, because they are more specific. We therefore introduce the concept of margin-closedness, a generalization of closed itemsets (e.g. [21]).

DEFINITION 4. *A Chord* c_i *is* margin-closed *w.r.t. a threshold* $\alpha < 1$ *if there are no super-Chords that have almost the same support, i.e.,* $\forall c_j \supset c_i, \frac{sup(c_j)}{sup(c_i)} < 1 - \alpha$.

Several Chords connected with a partial order form a *Phrase*.

DEFINITION 5. *A* Phrase *pattern is a partial order of* $k > 1$ *Chords. A Phrase starts with the first Chord and ends with the last Chord. The Chords within a Phrase are not allowed to overlap. We say the Phrase* $p_i \supset p_j$ *is a super-Phrase of* p_j *if* p_i *describes the partial order of a superset of of the Chords of* p_j *and all common Chords have the same partial order. The* support $sup(c)$ *of a Phrase* p *is the number of observations.*

The two similar Chord sequences in the bottom rows of Figure 4(a) and Figure 4(b) are summarized by the partial order graph of a Phrase shown in Figure 4(c).



(a) Some Chords (b) Similar Chords (c) Phrase for both

Figure 4: Chords summarize several overlapping Tones. Phrases summarize similar sequences of Chords.

To avoid redundancies one can again only consider margin-closed Phrases analogous to Chords.

4. TIME SERIES KNOWLEDGE MINING

The complete TSKM process [10] describes how TSKR patterns can be obtained from numerical time series. Here, we will concentrate on the core steps of mining coincidence and partial order from symbolic interval data.

4.1 Mining coincidence

The input for mining coincidence in form of Chords is a set of Tones with the respective symbolic interval sequence. A Chord is observed whenever at least s_{min} different Tones coincide for a duration of at least δ. A Chord is frequent, if the total duration of all intervals where it is observed exceeds the minimum support sup_{min}. The mining of Chords thus consists of picking a subset of all Tones and comparing the support with all super-Chords to ensure margin-closedness. This is similar to mining closed frequent itemsets, we therefore adapt the CHARM [21] algorithm for our purpose. We only describe the basic algorithm and refer to [21] for possible performance improvements.

Algorithm 4.1 performs a depth-first search for margin-closed Chords. The function EXTEND is recursively called given a prefix Chord c_p and a set of super-Chords S. The recursion is started with an empty Chord and all trivial frequent Chords as possible extensions in Line 1. The prefix Chord c_p is extended by all combinations of super-Chords from the set S in the double loop starting in Line 3. The variable $\hat{c}_i$ (Line 4) stores the initial super-Chord extension by c_i. The extensions with another Chords c_j are filtered by the minimum support in Line 6. The core part of the algorithm is the comparison of the support of the super-Chord $\hat{c}_i \cup c_j$ with the support of both sub-Chords $\hat{c}_i$ and c_j. The following four cases (Lines 7, 9, 11, and 13) are generalizations of the four properties for itemsets in [21] to margin-closed Chords and will be explained in detail below. In Line 19 the current Chord $\hat{c}_i$ is added to the set of margin-closed Chords if it is not subsumed by an already found margin-closed Chord. The function EXTEND is called recursively with the super-Chords $\hat{S}$ of the current Chord $\hat{c}_i$ in Line 22 unless the maximum Chord size is reached. The recursion stops as soon as the set of extensions S is empty.

The four different cases of the support comparison are depicted for an example in Figure 5. Let A, B, C be some Tone patterns. Only one instance of each Tone pattern is shown for demonstration purposes, the length and overlaps of the intervals are meant to be representative for the whole data set. For our example, let the arguments to the EXTEND function be the trivial Chord $c_p = A$ and the set of super-Chords be $S = \{AB, AC\}$. For the first extension $\hat{c}_i = A \cup AB = AB$ the support of $\hat{c}_i \cup c_j = AB \cup AC = ABC$ needs to be compared to the support of $\hat{c}_i = AB$ and $c_j = AC$.

The first case in Line 7 is shown in Figure 5(a). The super-Chord ABC has almost the same support as both sub-Chords. When AB or AC are observed, so is almost always ABC. We can effectively replace both sub-Chords with the super-Chord by adding $c_j = AC$ to the current extension $\hat{c}_i$ and excluding it from further processing in the inner loop. The second case in Line 9 is shown in Figure 5(b). While AB has the same support as ABC, AC has significantly more. Again we can add AC to the current extension $\hat{c}_i$ because whenever AB occurs, so does AC. We can not delete AC from S in this case, however, because combinations with other Chords still need to be considered. The third case in Line 11 is shown in Figure 5(c). The Chord AB has significantly more support than ABC, while AC does not. The super-Chord ABC is possibly a closed Chord, it needs to be added to the set of immediate super-Chords of $\hat{c}_i = AB$ for the recursive function call. The Chord AC can be excluded from further processing, because whenever it is observed, so is AB. The last case in Line 13 is shown in Figure 5(d). Both AB and AC have significantly more support than ABC. ABC is added to the set of possibly closed super-Chords for the recursion and no pruning can be performed.

In [21] the CHARM algorithm is reported to scale up linearly with the number of transactions. We observed similar behavior with Algorithm 4.1. The parameter δ needs to be chosen according to the application domain. The minimum duration of a Chord needs to be long enough for an expert to consider the coincidence of several properties meaningful. The minimum support and the threshold for margin-closedness directly control the number of Chords found. They should be set to larger values first and be made

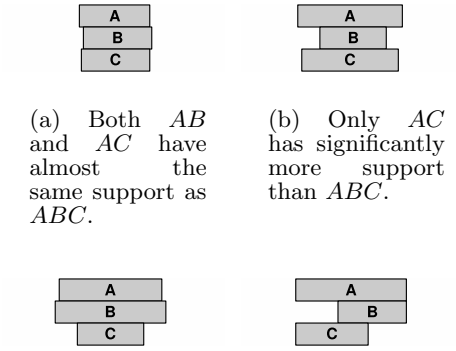
Algorithm 4.1 Depth-first margin-closed Chord mining.

Input:
Set of Tones T.
Minimum Chord duration δ.
Minimum Chord support sup_{min} (default 0.01).
Minimum Chord size s_{min} (default 2).
Maximum Chord size s_{max} (default ∞).
Threshold α for margin-closedness (default 0.1).

Output:
Set R of margin-closed Chords.

1: $S := \{t \in T | sup(t) >= sup_{min}\}$
2: $R := \text{EXTEND}(\emptyset, S, \emptyset)$

$\text{EXTEND}(c_p, S, R)$
3: **for all** $c_i \in S$ **do**
4: $\quad \hat{c}_i := c_p \cup c_i \quad \hat{S} := \emptyset$
5: $\quad$ **for all** $c_j \in S$ with $j > i$ **do**
6: $\quad\quad$ **if** $sup_\delta(\hat{c}_i \cup c_j) \geq sup_{min}$ **then**
7: $\quad\quad\quad$ **if** $\frac{sup_\delta(\hat{c}_i \cup c_j)}{max(sup_\delta(\hat{c}_i), sup_\delta(c_j))} \geq 1 - \alpha$ **then**
8: $\quad\quad\quad\quad \hat{c}_i := \hat{c}_i \cup c_j \quad S := S \setminus \{c_j\}$
9: $\quad\quad\quad$ **else if** $\frac{sup_\delta(\hat{c}_i \cup c_j)}{sup_\delta(\hat{c}_i)} \geq 1-\alpha$ and $\frac{sup_\delta(\hat{c}_i \cup c_j)}{sup_\delta(c_j)} < 1-\alpha$ **then**
10: $\quad\quad\quad\quad \hat{c}_i := \hat{c}_i \cup c_j$
11: $\quad\quad\quad$ **else if** $\frac{sup_\delta(\hat{c}_i \cup c_j)}{sup_\delta(\hat{c}_i)} < 1-\alpha$ and $\frac{sup_\delta(\hat{c}_i \cup c_j)}{sup_\delta(c_j)} \geq 1-\alpha$ **then**
12: $\quad\quad\quad\quad \hat{S} := \hat{S} \cup \{\hat{c}_i \cup c_j\} \quad S := S \setminus \{c_j\}$
13: $\quad\quad\quad$ **else**
14: $\quad\quad\quad\quad \hat{S} := \hat{S} \cup \{\hat{c}_i \cup c_j\}$
15: $\quad\quad\quad$ **end if**
16: $\quad\quad$ **end if**
17: $\quad$ **end for**
18: $\quad$ **if** $|\hat{c}_i| \geq s_{min}$ and $\forall c \in R$ with $\hat{c}_i \subseteq c$ holds $\frac{sup_\delta(c)}{sup_\delta(\hat{c}_i)} < 1 - \alpha$ **then**
19: $\quad\quad R := R \cup \{\hat{c}_i\}$
20: $\quad$ **end if**
21: $\quad$ **if** $|\hat{c}_i| < s_{max}$ **then**
22: $\quad\quad R := \text{EXTEND}(\hat{c}_i, \hat{S}, R)$
23: $\quad$ **end if**
24: **end for**



(a) Both AB and AC have almost the same support as ABC.

(b) Only AC has significantly more support than ABC.

(c) Only AB has significantly more support than ABC.

(d) Both AB and AC have significantly more support than ABC.

Figure 5: The four cases of Algorithm 4.1 for mining margin-closed Chords. The support of the Chords AB and AC is compared to the support of the common super-Chord ABC.

smaller if the results are too coarse.

4.2 Mining partial order

Phrases are similar to Episodes [8] but describe a partial order of time intervals instead of time points. The mining of margin-closed Phrases can be performed in several steps similar to the closed partial orders of [2]. Algorithm 4.2 shows the high level overview of our proposal.

Algorithm 4.2 High level algorithm to find margin-closed Phrases.

Input:
Set of Chords C.
Minimum duration ϵ of Chords in Phrase (default $\frac{\delta}{2}$).
Minimum Phrase support sup_{min} (default 1).
Minimum path length in Phrase s_{min} (default 2).
Threshold α for margin-closedness (default 0.1).
Transaction windows $W = \{[s_i, e_i] | i = 1, ..., w\}$.

Output:
Set of Phrases.

1: Convert $\mathcal{C}$ to itemset interval series $\mathcal{I}$ using ϵ.
2: Mine pairs (s, T) composed of a closed sequential pattern s occurring within the transaction windows $T \subseteq W$ using sup_{min} and s_{min}
3: Create margin-closed maximal pairs (S, T) where S is the set of all closed sequential patterns occurring in all transaction windows $T \subseteq W$ using sup_{min} and α.
4: Build partial order of Chords for each set S.

First, the interval sequence of Chords is converted to an itemset interval series in Step 1. One itemset is created for each interval of minimum duration ϵ where no Chords change. The itemset contains all symbols of the currently active Chords.

For Step 2 the single long itemset interval series needs to be implicitly converted to a set of shorter sub series by creating a sequence of intervals. Any standard algorithm for closed sequence mining, e.g., CloSpan [20], can then be used with the restriction to pick at most one item per itemset to exclude overlapping Chords. This also speeds up the mining of closed sequential patterns, because it reduces the search space to total orders of single items. As a result we obtain pairs (s, T) composed of a closed sequential pattern s occurring in the transaction windows T.

The mining of margin-closed partial orders (Step 3) is the largest deviation from [2] where pairs are formed consisting of a set of closed sequential patterns and the list of transactions in which they all occur. The pairs were required to be maximal in the sense, that there is no additional sequential pattern, that occurs in all transactions and no additional transaction in which all patterns occur. The sequential patterns were grouped by first considering all patterns with the same list of transactions and then adding any pattern with a superset of transactions that do not make the group redundant. Such groups are closed w.r.t. frequency and form a concept lattice. Each group of closed sequences can then be converted to a partial order.

When adding the constraints for margin-closed pairs, this algorithm cannot be used anymore. For each group not only sequential patterns with the same or longer transaction lists need to be considered, but also patterns with only slightly shorter lists. For a valid group of patterns, the intersection of all transaction lists is allowed to be at most $\alpha * 100$ percent shorter than the longest. Since all sequential patterns in a group influence the intersection, the search space for the groups consists of all 2^k combinations of k closed sequences. Interestingly, this corresponds to the problem of finding margin-closed itemsets, if we consider each closed sequential pattern an item and each group of sequential patterns an itemset. We can therefore again adapt the CHARM [21] algorithm to tackle this problem efficiently. Only minor modifications to Algorithm 4.1 are necessary, we therefore refer the interested reader to [10] for the complete listing.

For the last step of converting a set of sequences into a partial order necessary conditions are given in [2] making the construction of the final Phrase representation straightforward, see [10] for a simple algorithm. The mining of margin-closed Phrases is the computationally most demanding task of the TSKM, because the number of closed sequential patterns used as items can be quite large. Reducing the number of closed sequential patterns with constraints on minimum length and minimum frequency is advisable. Similar to Chord mining, using higher values of α will dramatically reduce the search space and the runtime.

5. EXPERIMENTS

We demonstrate the efficacy and efficiency of the TSKM on a dataset from sports medicine. More experiments with video data can be found in [10]. We compare the resulting patterns to results obtained using Allen's relations. For Allen's relations we used the pattern format of [5] to avoid ambiguity. Support was measured based on counting occurrences in windows to be comparable with the Phrase mining. We further pruned the set of Allen patterns applying the concept of margin-closedness in a brute force post-processing step.

The data was collected from tests in professional In-Line Speed Skating. An athlete performed a standardized indoor test at a speed of 7,89m/s on a large motor driven tread-

mill. EMG (Electromyography) and kinematic data were measured for 30 seconds. The 6-dimensional kinematic time series with angles and angular speeds of the hip, knee, and ankle joints was clustered into 3 states. The Tones were labeled *swing*, *glide+push*, and *recovery*. Three EMG time series describe the activation of leg muscles mainly responsible for forward propulsion: Medial Gastrocnemius (*Gastroc*), Vastus Medialis (*Vastus*), and Gluteus Maximus (*Gluteus*). Each was discretized with the Persist algorithm [11] to obtain the Tones *low*, *high*, and *very high*. One series describes the foot contact with Tones *on* and *off* obtained from a pressure sensor in the skate. Non-overlapping windows corresponding to the 19 movement cycles of the experiment were used. The goal was to identify typical muscle activation patterns during the cyclic movement with complex inter-muscular coordination patterns. The experimental results were evaluated by an expert.

For the TSKR patterns we first mined Chords with a minimum duration of 50ms. Shorter phenomena related to muscle activation are physiologically not plausible. The minimum size was set to 3, the maximum size is naturally bounded by the 5 dimensions of the interval series. With a minimum support of 2% a total of 70 Chords was found. Restricting the patterns to closed Chords reduced the number to 60, mining margin-closed Chords with $\alpha = 0.1$ returned 18 patterns. At this point it is difficult to decide automatically whether rare large or small frequent Chords are better. The lattice of Chords was presented to the expert, who selected and labeled 9 as the most interesting.

Using only these Chords we found 30 closed sequences with a minimum length of 2 and a minimum frequency of 10 that were merged into 20 closed Phrases. When using $\alpha = 0.1$ 15 margin-closed Phrases were obtained. Again we presented the lattice of Phrases to the expert who selected the most specific and the second most general patterns as the most interesting. The general Phrase present in 17 out of the 19 movement cycles is shown in Figure 6(a) with images from the skating experiments. The advantages of the hierarchical structure of the TSKR is shown by unfolding one Chord from the Phrase to show the coinciding Tones and one Tone to show the original numerical time series with the thresholds for discretization. The more detailed Phrase shown in Figure 7 was observed in 10 cycles, it describes a total order of 8 Chords and provides details on the successive activation of the three major leg muscles.

The pruning by margin-closedness largely reduced the number of patterns so they could easily be analyzed manually to trade off pattern size vs. pattern frequency. We analyzed the search space of Chords in dependence of the parameters to quantify the pruning effects. In Figure 8 the number of Chords found is shown on the left for different values of the parameter α determining margin-closedness. The value of 0.1 as used for the analysis above, effectively prunes the size of the result down to a manageable number of patterns that can be analyzed manually. Using lower values would result in a dramatic increase of patterns, using higher values results in less than five patterns for $\alpha > 0.18$. We also measured the number of recursive calls to the EXTEND function as an estimate of the runtime of the algorithm independent of implementation and hardware. As is visible on the right of Figure 8 the concept of margin-closedness does not only reduce the size of the result set, it also effectively prunes the search space during mining and speeds up the computation. This is because notion of margin-closedness more often enables the activation of the cases 1-3 in Algorithm 4.1 that have a pruning effect. For $\alpha \geq 0.2$ less than 10% of the functions call of strict closed Chord mining are needed.

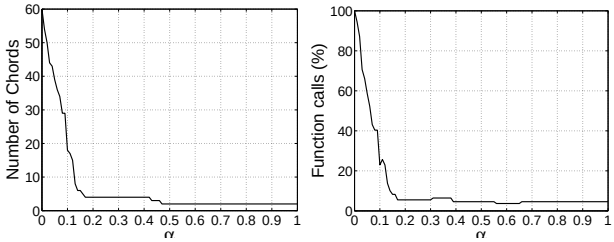


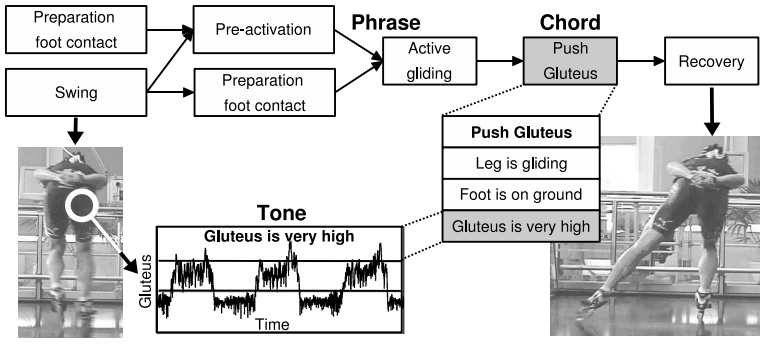
Figure 8: Effects of different values of α determining margin-closedness on the number of Chords (left) and the complexity of the search (right).

We searched the same data for patterns expressed with Allen's relations using a minimum frequency of 10 and a minimum size of 2 resulting in 4208 distinct patterns. Pruning by enforcing margin-closed patterns with $\alpha = 0.1$ returned 487 patterns. There were 8 patterns of size 9 and one of size 10. The analysis was performed by the expert looking at an example of each pattern as shown for the size 10 pattern in Figure 6(b) because the listing of all 45 or 36 pairwise relations did not offer a comprehensible description. While the relation of the Gluteus muscle activity to the movement phases was recognized as valid, the fact that no information on the Vastus muscle was given and the Gastroc muscle appeared only with low activity rendered the pattern useless to the expert. The fact that the three movement phases appear in the displayed order and the two Foot states alternate is already obvious from the input data and offers no information gain.

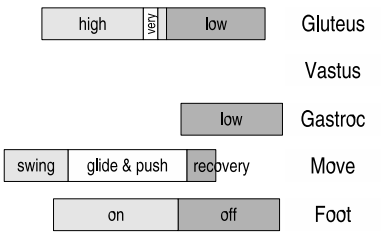
Because a correspondence of muscle activity to movement related information was sought we filtered the pattern set explicitly for patterns containing the states *high* and *very high* for the *Vastus* and *Gluteus* muscles. None of the 5 patterns found contained useful information on the foot contact. Further analysis with less restrictive constraints suggested, that patterns with many muscle states offer only very limited information on movement and foot contact and the other way around. The relation of muscle related interval series to movement related interval series, all of which are subject to noise from the recording process and the discretization procedure, did not seem to be well representable with Allen's relations.

6. DISCUSSION

The mining algorithms for Allen's relations are all based on the A-priori principle from association rule mining. The patterns in [5] are build in a breadth first manner. It is well known, however, that this approach is inferior in performance to depth first search for long patterns [19] as performed by our algorithms. Nevertheless, the search for closed frequent itemsets can become infeasible for a large number of items. The novel concept of margin closedness helps to partly alleviate this complexity. It serves as a sampling of the results and a pruning of the search space to avoid redundancy. Coarse results can be found very fast and refined



(a) Phrase explaining most observed movement cycles. Chords (e.g. *Push Gluteus*) can be expanded to show the coinciding Tones. Tones (e.g. *Gluteus is very high*) are obtained from the numerical sensor time series.



(b) One instance of the largest Allen pattern. The 10 intervals are connected with 45 pairwise relations.

Figure 6: Patterns found in inline skating data.

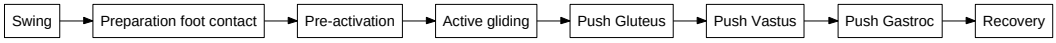


Figure 7: Detailed Phrase of skating data with additional information on muscle activation.

if necessary. We expect margin-closed patterns to be useful in other itemset mining tasks as well.

Phrases are similar in structure to sequential patterns. Each time point of a symbolic interval sequence could be considered a transaction with an itemset holding one item for each symbolic interval that contains the time point. This leads to very long transactions (for the Skating about 1,500 itemsets per transaction) that are highly redundant. Applying standard sequential pattern mining techniques to this data would be a large waste of computing resources. Using our novel data model conversion to itemset intervals greatly reduces this redundancy. Still, test with algorithms for sequential pattern were extremely slow compared to the search for Phrases with our semantically motivated search space restrictions.

In [2] it is shown how grouping of closed sequential patterns leads to closed partial orders. It is tempting to assume that margin-closed sequential patterns will also lead to margin-closed partial orders, but simple examples show that this is not the case (see [10] for details).

The Phrases were considered valid and interesting by the expert. Of particular importance is the connection of external variables describing the movement with the internal observations on muscle activation made by the Chord patterns that could not be discovered with Allen's relations. The graphical representation of the Phrases offered a great benefit over simply viewing the input time series, as is common practice in the analysis of such data in sports medicine.

7. SUMMARY

We presented efficient and effective unsupervised mining algorithms for the new temporal pattern language TSKR with algorithms adapted from itemset and sequential pattern mining. The pruning of the search space with margin-closedness reduced the number of patterns presented to the analyst and speeded up the mining. Mining TSKR patterns was demonstrated to be more efficient, more effective, and more meaningful than Allen's relations in a real life example.

Acknowledgments: We thank Alfred Ultsch for guidance and Olaf Hoos for providing the Skating data and images and interpreting the results. This work was partly supported by Siemens Corporate Research, Princeton, NJ, USA.

8. REFERENCES

[1] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.
[2] G. Casas-Garriga. Summarizing sequential data with closed partial orders. In *Proc. SDM'05*, pages 380–391, 2005.
[3] P. R. Cohen. Fluent learning: Elucidating the structure of episodes. In *Proc. IDA'01*, pages 268–277, 2001.
[4] G. Guimarães and A. Ultsch. A method for temporal knowledge conversion. In *Proc. IDA'99*, pages 369–380, 1999.
[5] F. Höppner. *Knowledge Discovery from Sequential Data*. PhD thesis, Technical University Braunschweig, Germany, 2003.
[6] P.-S. Kam and A. W.-C. Fu. Discovering temporal patterns for interval-based events. In *Proc. DaWaK'00*, pages 317–326, 2000.
[7] 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.
[8] H. Mannila, H. Toivonen, and I. Verkamo. Discovery of frequent episodes in event sequences. In U. M. Fayyad and R. Uthurusamy, editors, *Proc. KDD'96*, pages 210–215, 1995.
[9] C. Mooney and J. F. Roddick. Mining relationships between interacting episodes. In *Proc. SDM'04*, 2004.
[10] F. Mörchen. *Time Series Knowledge Mining*. PhD thesis, Philipps-University Marburg, Germany, 2006.
[11] F. Mörchen and A. Ultsch. Optimizing time series discretization for knowledge discovery. In *Proc. KDD'05*, pages 660–665, 2005.
[12] 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.
[13] J. F. Roddick and C. H. Mooney. Linear temporal sequences and their interpretation using midpoint relationships. *IEEE TKDE*, 17(1):133–135, 2005.
[14] E. Schwalb and L. Vila. Temporal constraints: A survey. Technical report, ICS, University of California at Irvine, 1997.

[15] S. G. Sripada, E. Reiter, and J. Hunter. Generating English summaries of time series data using the Gricean maxims. In *Proc. KDD'03*, pages 187–196, 2003.

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

[17] A. Ultsch. Unification-based temporal grammar. Technical Report 37, CS Dept., Philipps-University Marburg, Germany, 2004.

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

[19] J. Wang and J. Han. BIDE: Efficient mining of frequent closed sequences. In *Proc. ICDE'04*, pages 79–90, 2004.

[20] X. Yan, J. Han, and R. Afshar. CloSpan: Mining closed sequential patterns in large datasets. In *Proc. SDM'03*, pages 166–177, 2003.

[21] M. J. Zaki and C.-J. Hsiao. Efficient algorithms for mining closed itemsets and their lattice structure. *IEEE TKDE*, 17(4):462–478, 2005.