Interval-based sequence mining using FCA and the NextPriorityConcept algorithm

Salah Eddine Boukhetta, Jérémy Richard, Christophe Demko, and Karell Bertet

Laboratory L3i, La Rochelle University, La Rochelle, France

Abstract. In this paper, we are interested in sequential data analysis using `GALACTIC`, a new library based on Formal Concept Analysis (FCA) for calculating a concept lattice from heterogeneous and complex data. Inspired by the pattern structure theory, data in `GALACTIC` are described by predicates according to their types and a system of plugins allows an easy integration of new characteristics and new descriptions. We present new ways to analyse interval-based sequences, where items persist in time. Here we address the question of mining relevant sequential patterns, describing a set of sequences, by maximal common subsequences, or shortest supersequences. Experimentation on two real sequential datasets shows the effectiveness of our plugins in term of size of the lattice and of running time.

Keywords: Formal concept analysis · Lattice · Pattern structures · Interval-based sequences · Maximal common subsequences · Shortest common supersequences

1 Introduction

Sequences appear in many areas: sequences of words in a text, trajectories, surfing on the internet, or buying products in a supermarket. A sequence is a succession $\langle x_i \rangle$ of symbols, sets or events. Sequence mining is a topic of data mining which aims at finding frequent patterns in a dataset of sequences. Many algorithms have been proposed for mining sequential patterns, such as GSP [25], PrefixSpan [24], CloSpan [29], etc. These algorithms take as input a dataset of sequences and a minimum support threshold, and generate all frequent subsequences. Some algorithms mine time-point sequences $\langle (t_i, x_i) \rangle$, where an item x_i occurred at a timestamp t_i, for example for discovering episodes in a long time-point sequence [23, 26]. In real world applications, events may persist in time, or in an interval of time $(\underline{t}_i, \overline{t}_i)$, we call these sequences, *interval-based sequences* $\langle (\underline{t}_i, \overline{t}_i, X_i) \rangle$, where X_i is an itemset. They are mostly analysed using Allen's interval relations [1]. To quote from Kam and Fu's work on discovering temporal interval sequences [18], the patterns discovered are of type "event A's occurrence time overlaps with that of event B and both of these events occur before event C appears". Other works also used Allen's relations to discover interval based patterns [17, 28].

Formal Concept Analysis (FCA) appears in 1982 [27], then in the Ganter and Wille's 1999 work [14], it is issued from a branch of applied lattice theory that first appeared in the book of Barbut and Monjardet in 1970 [2]. The lattice property guarantees both a hierarchy of clusters, and a complete and consistent navigation structure for interactive approaches [11]. The formalism of pattern structures [13, 20] and abstract conceptual navigation [10, 9] extend FCA to deal with non-binary data, where data is described by patterns such that the pattern space must be organised as a semi-lattice in order to maintain a Galois connection between objects and their descriptions. By FCA framework, pattern lattice and bases of rules are defined, where a concept is composed of a subset of objects together with their common patterns, and a rule possesses patterns in premises and conclusions. However, pattern lattices are huge, often untractable [19], and the need for approaches to drive the search towards the most relevant patterns is a current challenge. Logical Concept Analysis [12] is a generalization of FCA in which sets of attributes are replaced by logical expressions. The power set of attributes mentioned by the Galois connection is replaced by an arbitrary set of formulas to which are associated a deduction relation (i.e., subsumption), and conjunctive and disjunctive operations, and therefore forms a lattice. Inspired by pattern structures, the NextPriorityConcept algorithm, introduced in a recent article [8] proposes a user-driven pattern mining approach for heterogeneous and complex data as input. This algorithm allows a generic pattern computation through specific *descriptions* of objects by predicates. It also proposes to reduce predecessors of a concept by the refinement of a set of objects into a fewer one through specific user exploration *strategies*, resulting in a reduction of the number of generated patterns. Some algorithms appear within FCA framework for analysing sequence data; we can mention works for mining medical care trajectories using pattern structures [5, 6], sequence mining to discover rare patterns [7], and other studies on demographic sequences [15, 16]. But for discovering interval-based sequence using FCA methods, we found fewer works. We can cite Kaytoue et al.'s work on gene expression data [21].

In this article, we propose a new sequence mining approaches using the NextPriorityConcept algorithm, with descriptions and strategies dedicated to interval-based sequences. We propose two different descriptions that describe a subset of objects by subsequences or supersequences. We also propose five strategies of pattern exploration in order to generate a reduction of a cluster of interval-based sequences, i.e., its predecessors in the pattern lattice.

Section 2 introduces basic definitions related to interval sequence mining and a short description of the NextPriorityConcept algorithm. Section 3 will be dedicated to our new interval-based sequence descriptions and strategies. Experimental results are presented in Section 4.

2 Preliminaries

2.1 Interval-based Sequences

A sequence s is a succession of itemsets from a dictionary Σ, often in the form of $s =< X_i >_{i \leq n}$, where $X_i \subseteq \Sigma$ is a subset of items i.e., itemset. A temporal sequence is a sequence where each itemset X_i must have an associated timestamp t_i. An *Event* (or *Time frame*) E, is a triple $E = (\underline{t}, \overline{t}, X)$ where $X \subseteq \Sigma$ is an *itemset*, $\underline{t}$ is the starting time and $\overline{t}$ is the ending time, $\underline{t} \leq \overline{t}$. For better readability we refer to $(\underline{t}, \overline{t})$ by T.

Interval-based sequence. An *interval-based sequence* (or *Time frame sequence*) $s = \langle (T_i, X_i) \rangle_{i \leq n}$ is a list of events (or time frames), verifying $\overline{t_i} < \underline{t_{i+1}}$, thus an interval-based sequence is a list of separate intervals containing itemsets. The size of the interval-based sequence is the number of its time frames. We refer to the interval-based sequence by *sequence*.

Consider the example in Figure 1 for an alphabet $\Sigma = \{C, M, P, H\}$ (where C stands for Castle, M for Museum, P for Public Garden and H for Historical Place), the sequences represent trajectories of visits of three tourists $s1, s2$ and $s3$. In this example, visitors may be in two or more different locations at the same interval as the intervals are large enough and we may don't have the exact interval of each location.

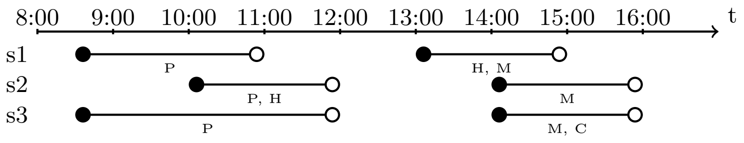

Fig. 1. Example of interval-based sequences

Subinterval. For two intervals, $T = (\underline{t}, \overline{t})$ and $T' = (\underline{t'}, \overline{t'})$, we say that T is subinterval T', if : $\underline{t} \geq \underline{t'}$ and $\overline{t} \leq \overline{t'}$ and we write $T \preceq T'$, that corresponds to the containing relation from Allen's relations [1].

Projections. We introduce the *projection* operator Φ of a sequence s, over a given interval T, that selects all the itemsets of the sequence included in this interval : $\Phi_T(s) = \{X' \ : \ T' \preceq T \text{ and } (T', X') \in s\}$. Dually, the *projection* operator Φ, over an itemset $X \subseteq \Sigma$ selects all the intervals where the items of X may occur: $\Phi_X(s) = \{T' \ : \ X' \subseteq X \text{ and } (T', X') \in s\}$. $\Phi_\Sigma(s)$ represents a set of all the intervals in s.

Subsequence. A sequence s, is *subsequence* of another sequence s', $s \Subset s'$ if for all $(T, X) \in s$, there exists $(T', X') \in s'$ such that $T \preceq T'$ and $X \subseteq X'$. We also say that s' is **supersequence** of s.

Affix. A prefix/suffix of a sequence $s = \langle (T_i, X_i) \rangle_{i \leq n}$ according to a window w, is the subsequence of s composed by the first/last w time frames of s, prefix$(s, w) = \langle (T_i, X_i) \rangle_{1 \leq i \leq w}$, suffix$(s, w) = \langle (T_i, X_i) \rangle_{(n-w) < i \leq n}$.

Cardinality. For a set of sequences A, an item $x \in \Sigma$ and an interval T, the function *card* gives the number of sequences $a \in A$ possessing the item x in the projection of a over T, $x \in \Phi_T(a)$.

$$card(A, T, x) = |\{a \ : \ x \in \Phi_T(a), a \in A\}| \tag{1}$$

When $card(A, T, x)$ is maximal, we denote $card(A, T, x)$ by $card_{max}(A, T)$. We define $card_{min}(A, T)$ in the same maner when $card(A, T, x)$ is minimal.

From example in Figure 1 we have, $\Phi_{(10:00,11:00)}(s2) = \{P, H\}$, the prefix of $s1$ is $\langle(08{:}30, 11{:}00), P)\rangle$, and all three tourists were in the museum from 14:00 to 15:00, so $\langle(14{:}00, 15{:}00), M)\rangle$ is subsequence of $s1, s2$ and $s3$. For $A = \{s1, s2, s3\}$ and $T = (11{:}00, 12{:}00)$ $card(A, T, P) = card_{\max}(A, T) = 2$.

2.2 Description of the NextPriorityConcept algorithm

The NextPriorityConcept algorithm [8] computes concepts for heterogeneous and complex data for a set of objects G, its main characteristics are:

Heterogeneous data as input, described by specific predicates. The algorithm introduces the notion of *description* δ as an application to provide predicates describing a set of objects $A \subseteq G$. Each concept $(A, \delta(A))$ is composed of a subset of objects A and a set of predicates $\delta(A)$ describing them. Such generic use of predicates makes it possible to consider heterogeneous data as input, i.e., numerical, discrete or more complex data. However, unlike classical pattern structures, predicates are not globally computed in a preprocessing step, but locally for each concept.

Concept lattice generation. The NextPriorityConcept algorithm is inspired by Bordat's algorithm[3], also found in Linding's work [22], that recursively computes the immediate successors of a concept, starting with the bottom concept. It is a dual version that computes the immediate predecessors of a concept, starting with the top concept $(G, \delta(G))$ containing the whole set of objects, until no more concepts can be generated. The use of a priority queue ensures that each concept is generated before its predecessors, and a mechanism of propagation of constraints ensures that meets will be computed. NextPriorityConcept computes a concept lattice and therefore is positioned in FCA framework, with the possibility of extraction of rules, closure computations or navigation in the lattice, that can be useful in many fields of pattern mining and discovery.

Predecessors selection by specific strategies. The algorithm also introduces the notion of *strategy* σ to provide predicates (called *selectors*) describing candidates for an object reduction of a concept $(A, \delta(A))$ i.e., predecessors of $(A, \delta(A))$ in the pattern lattice. A selector proposes a way to refine the description $\delta(A)$ to a reduced set $A' \subset A$ of objects. Several strategies are possible to generate predecessors of a concept, going from the naive strategy classically used in FCA that considers all the possible predecessors, to strategies reducing the number of predecessors in order to obtain smaller lattices. Selectors are only used for the predecessors' generation, they are not kept either in the description or in the final set of predicates. Therefore, choosing or testing several strategies at each iteration in a user-driven pattern discovery approach would be interesting.

The main result in [8] states that the NextPriorityConcept algorithm computes the formal context $\langle G, P, I_P\rangle$ and its concept lattice (where P is the set of predicates describing the objects in G, and $I_P = \{(a, p),\ a \in G, p \in P\ :\ p(a)\}$ is the relation between objects and predicates) if description δ verifies $\delta(A) \sqsubseteq \delta(A')$

for $A' \subseteq A$. The run-time of the NextPriorityConcept algorithm has a complexity $O(|\mathcal{B}|\,|G|\,|P|^2\,(c_\sigma + c_\delta))$ (where $\mathcal{B}$ is the number of concepts, c_σ is the cost of the strategy and c_δ is the cost of the description), and a space memory in $O(w\,|P|^2)$ (where w is the width of the concept lattice).

3 NextPriorityConcept for sequences

In order to mine interval-based sequences with NextPriorityConcept algorithm, we have to define descriptions and strategies for sequences. Consider a set G of sequences whose size is smaller than n, defined on an alphabet Σ as input:

A description δ is a mapping $\delta : 2^G \to 2^P$ which defines a set of predicates $\delta(A)$ describing any subset $A \subseteq G$ of sequences. Predicates are of form, *"is subsequence/supersequence of"*.

A strategy σ is a mapping $\sigma : 2^G \to 2^P$ which defines a set of selectors $\sigma(A)$ to select strict subset A' of A as predecessor candidates of any concept $(A, \delta(A))$ in the pattern lattice.

Predicates are computed using the subsequence relation in the form *"is subsequence of "* or *"is supersequence of"*. For better readability, the sets $\delta(A)$ and $\sigma(A)$ will be treated either as sets of predicates/selectors, or as sets of sequences, they can reciprocally be deduced from each other.

3.1 Description for interval sequences

We define two descriptions for a subset $A \subseteq G$ of sequences. The maximal common time frame description `MCTF` refers to the classical maximal common subsequence description [4] and corresponds to the set of maximal subsequences of all sequences in A. The shortest supersequence time frame description `SSTF` contains all minimal supersequences of sequences in A.

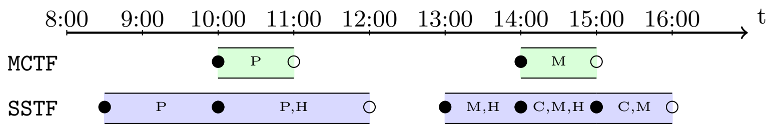

Fig. 2. $\delta_{\text{MCTF}}(\{s1, s2, s3\})$ and $\delta_{\text{SSTF}}(\{s1, s2, s3\})$ for $s1, s2$ and $s3$ in Figure 1.

Figure 2 represents $\delta_{\text{MCTF}}(A)$ and $\delta_{\text{SSTF}}(A)$ for $A = \{s1, s2, s3\}$ from Figure 1. We can observe that `MCTF` could be interpreted as "conjunction" where $(14{:}00, 15{:}00, \{M\})$ in δ_{MCTF} means that all $s1, s2$ and $s3$ contain $(14{:}00, 15{:}00, \{M\})$. Dually, `SSTF` could be interpreted as "disjunction". More formally, `MCTF` and `SSTF` are defined for a subset $A \subseteq G$ of sequences by:

Maximal Common Time Frame (MCTF) description.

$$\delta_{\text{MCTF}}(A) = \{\langle (T, X) \rangle \ : \forall a \in A, X \subseteq \Phi_T(a)\} \tag{2}$$

Shortest Shared Time Frame (SSTF) description.

$$\delta_{\text{SSTF}}(A) = \{\langle (T, X) \rangle \ : \forall a \in A, \Phi_T(a) \subseteq X\} \tag{3}$$

To compute the two descriptions of a set A of sequences, we iterate on the sequences of A, and update the resulting sequences of $\delta(A)$ with the common parts. Therefore the complexity of the description is $c_\delta^{MCTF} = c_\delta^{SSTF} = O(s\,|A|\,log(|A|)) \leq O(s\,|G|\,log(|G|))$ where s is the maximal size of the computed sequences. We have to ensure that the NextPriorityConcept algorithm generates a concept lattice. These descriptions must verify $\delta(A) \sqsubseteq \delta(A')$ for $A' \subseteq A$:
Proposition 1 *For $A' \subseteq A \subseteq G$, we have the following two properties:*

1. $\delta_{MCTF}(A) \sqsubseteq \delta_{MCTF}(A')$
2. $\delta_{SSTF}(A) \sqsubseteq \delta_{SSTF}(A')$

Proof: Let A and A' be two subsets of G such that $A' \subseteq A$.

1. Let $c \in \delta_{\text{MCTF}}(A)$, i.e., c is a maximal subsequence of A. From $A' \subseteq A$ we can deduce that c is also a subsequence of sequences in A', but c is not necessarily a maximal subsequence for A'. If c is a maximal subsequence in A' then $c \in \delta_{\text{MCTF}}(A')$. Otherwise, there exists $c' \in \delta_{\text{MCTF}}(A')$ such that c is a subsequence of c'. In these two cases, we can deduce that, $\delta_{\text{MCTF}}(A) \sqsubseteq \delta_{\text{MCTF}}(A')$.
2. Let $s \in \delta_{\text{SSTF}}(A)$, i.e., s is the supersequence of all sequences in A. From $A' \subseteq A$ we can deduce that s is also a supersequence of sequences in A' but not necessarily the shortest one. If s is a shortest supersequence in A' then $s \in \delta_{\text{SSTF}}(A')$. Otherwise, there exists $s' \in \delta_{\text{SSTF}}(A')$ such that s is supersequence of s'. We can deduce that, $\delta_{\text{SSTF}}(A) \sqsubseteq \delta_{\text{SSTF}}(A')$.

$\square$

3.2 Strategies and selectors for time frame sequences

Strategies are used by the NextPriorityConcept algorithm to refine each concept $(A, \delta(A))$ into concepts with fewer objects (sequences) and more specific descriptions. It is important to clarify that a strategy must be used with a description composed of predicates of the same kind. Recall that `MCTF` description defines "is subsequence of" predicates whereas `SSTF` description defines "is supersequence of" predicates. We define one subseqeunce strategy for the `MCTF` description and four supersequence strategies for the `SSTF` description.

Strategy with subsequence selectors for `MCTF` description: The *Augmented Minimum Cardinality* strategy computes all the possible refinements of a concept $(A, \delta_{\text{MCTF}}(A))$ by adding in the events of sequences of δ_{MCTF} any item with a minimal cardinality $card(A, T, x)$ for each time frame T. More formally, σ_{AMC} is defined for $A \subseteq G$ by:

Augmented Minimum Cardinality.

$$\begin{aligned} \sigma_{\text{AMC}}(A) = \{\langle (T, X)\rangle :\ & \forall a \in A, \Phi_T(a) \subseteq X \text{ and } \forall x \in X \\ & card(A,T,x) = |A| \ \vee\ card(A,T,x) = card_{min}(A,T)\} \end{aligned} \qquad (4)$$

The cost for this strategy is clearly equal to the cost of `MCTF` description, $c_\sigma^{AMC} = c_\delta^{MCTF}$.

Figure 3, represents the generated Hasse diagram of sequences in Figure 1 using the `MCTF` description and the `AMC` strategy, where in each concept the symbol \$ represents the identifier of the concept, and the symbol # represents the number of sequences inside the concept, i.e., its support. The concept \$0 contains the description of the 3 visits. The concept \$1 describes the two visits, $s1$ and $s3$ as they were in the Public Garden from 08:30 to 11:00 then in the Museum from 14:00 to 15:00.

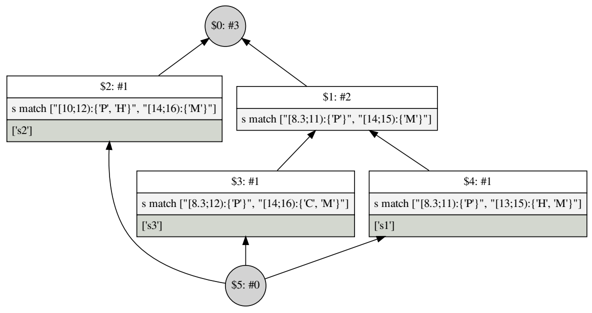
$0: #3
$2: #1
s match ["[10;12):{'P', 'H'}", "[14;16):{'M'}"]
['s2']
$1: #2
s match ["[8.3;11):{'P'}", "[14;15):{'M'}"]
$3: #1
s match ["[8.3;12):{'P'}", "[14;16):{'C', 'M'}"]
['s3']
$4: #1
s match ["[8.3;11):{'P'}", "[13;15):{'H', 'M'}"]
['s1']
$5: #0

Fig. 3. Hasse diagram of the reduced concept lattice for the AMC strategy and the MCTF description

Strategies with supersequence selectors for `SSTF` description: First, the *Simple Time Frame* strategy consists in simply generating selectors by deleting one item from each itemset on the `SSTF` description. More formally, σ_{STF} is defined for a subset of sequences $A \subseteq G$, and the `SSTF` description, by:

Simple Time Frame strategy (STF).

$$\sigma_{\text{STF}}(A, \delta_{\text{SSTF}}) = \{\langle (T, X \backslash \{x\}) \rangle \; : \forall x \in X, \; (T, X) \in s, s \in \delta_{\text{SSTF}}(A)\} \quad (5)$$

To implement this strategy, we have to consider any item of any sequence of the description, thus a complexity $c_\sigma^{STF} = O(s\, m\, c_\delta^{SSTF}) \leq O(s\, |\Sigma|\, c_\delta^{SSTF})$ where m is the maximal number of items in any time frame in the sequences of the description.

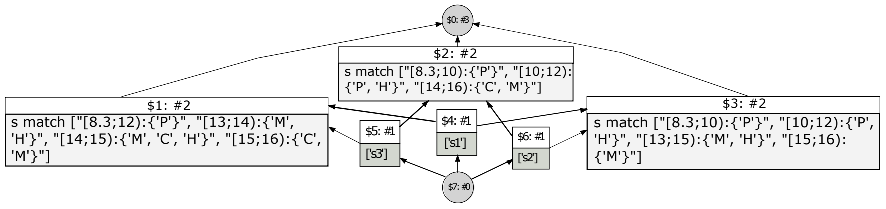
$0: #3
$2: #2
s match ["[8.3;10):{'P'}", "[10;12):{'P', 'H'}", "[14;16):{'C', 'M'}"]
$1: #2
s match ["[8.3;12):{'P'}", "[13;14):{'M', 'H'}", "[14;15):{'M', 'C', 'H'}", "[15;16):{'C', 'M'}"]
$5: #1
['s3']
$4: #1
['s1']
$6: #1
['s2']
$3: #2
s match ["[8.3;10):{'P'}", "[10;12):{'P', 'H'}", "[13;15):{'M', 'H'}", "[15;16):{'M'}"]
$7: #0

Fig. 4. Hasse diagram of the reduced concept lattice for the STF strategy and the SSTF description

Figure 4 represents the generated Hasse diagram using the `SSTF` description and the `STF` strategy. Concepts \$1, \$2, and \$3 contain descriptions of $\{s1, s3\}$, $\{s2, s3\}$, and $\{s1, s2\}$. Concept \$2 shows that at least one of $s2$ or $s3$ visited P from 08:30 to 10:00, then H or P from 10:00 to 12:00, and finally C or M from 14:00 to 16:00. The strategy constructs a lattice with all concepts, hence it is time consuming. So, we thought about strategies that may reduce the size of the lattice, and the time complexity. The *Bounds Time Frame* strategy consists in deleting only the items of cardinalities minimal or maximal. The *Window Affix Time Frame* strategy uses a *window* parameter w and generates the prefix and suffix. The *Alphabet Time Frame* strategy, deletes one item of the alphabet from all the time frames. More Formally, these strategies are defined for a subset A of sequences and the description δ_{SSTF} by:

Bounds Time Frame (BTF), for an integer *card*:

$$\begin{aligned}\sigma_{\text{BTF}}(A, c) = \{\langle (T, X \backslash \{x\}) \rangle \ : \ (T, X) \in s,\\ card(A, T, x) = c, s \in \delta_{\text{SSTF}}(A)\}\end{aligned} \tag{6}$$

In particular, we can consider $\sigma_{\text{BTF}}(A, card_{max})$ and $\sigma_{\text{BTF}}(A, card_{min})$
Window Affix Time Frame strategy (WATF), for a window size w:

$$\begin{aligned}\sigma_{\text{WATF}}(A, w) = \{\langle (T, X) \rangle \ : \ (T, X) \in s - \text{prefix}(s, w),\\ (T, X) \in s - \text{suffix}(s, w), s \in \delta_{\text{SSTF}}(A)\}\end{aligned} \tag{7}$$

Here the $s - \text{prefix}(s, w)$ means s without time frames in $\text{prefix}(s, w)$, same for $s - \text{prefix}(s, w)$.
Alphabet Time Frame strategy (ATF):

$$\sigma_{\text{ATF}}(A) = \{\langle (T, X \backslash \{x\}) \rangle \ : \forall (T, X) \in s, \ \forall x \in \Sigma, \ s \in \delta_{\text{SSTF}}(A)\} \tag{8}$$

The cost of the `BTF` strategy is $c_\sigma^{BTF} = O(s \, c_\delta^{SSTF})$, as it must calculate the predicates of `SSTF`, then iterate on the resulting sequences. For the `WATF` strategy, $c_\sigma^{WATF} = c_\delta^{SSTF}$, the w parameter is constant, so the cost is equal only to the cost of δ_{SSTF}. For the `ATF` strategy $c_\sigma^{ATF} = O(s \, c_\delta^{SSTF})$.

The NextPriorityConcept allows a user-driven approach for the data analyst to choose strategies that respond the best to the specifications of the data. With the `SSTF` description, the data analyst have a choice of 4 strategies. The `BTF` strategy allows a generation of predecessors where frequent or non-frequent events may not appear (maximum and minimum cardinalities). The `WATF` strategy focuses of events that appear first or last at the same interval. The `ATF` strategy focuses on clusters where some events may not appear. The use of these strategies reduces the time complexity of the lattice generation process and generate a smaller lattice than the `STF` strategy.

4 Experiments

In this section, we experimentally evaluate our descriptions and strategies for mining interval sequences. Our approach is different from previous works. We are not mining all frequent interval-based sequences, but we use our strategies and descriptions to mine only the relevant ones. To experimentally assess the effectiveness of our descriptions and strategies, we use `GALACTIC`[1] (**GA**lois **LA**ttices, **C**oncept **T**heory, **I**mplicational systems and **C**losures), a development platform

[1] https://galactic.univ-lr.fr

of the NextPriorityConcept algorithm, which mixed with a system of plugins, makes it possible easy integration of new kinds of data (descriptions and strategies). We have implemented new plugins for sequences. Experiments were performed on an Intel Core i7 2.20GHz machine with 32GB main memory. We run our experiments on two real datasets:

GeoLuciole **dataset** is issued from classical GPS trajectories of people's deplacements in the city of La Rochelle in France. By matching the GPS coordinates to districts of the city, raw data are transformed into semantic sequences. The data have been collected by a specific application named GeoLuciole that we have developed for the DA3T[2] project. The data contains only 15 trajectories with an average size of sequences equals to 2[3].

Wine-City **dataset** is issued from the museum "La cité du vin" in Bordeaux, France[4], gathered from the visits over a period of one year (May 2016 to May 2017). The museum is a large "open-space", where visitors are free to explore the museum the way they want. The trajectories in this dataset are of size 9 on average.

Comparison of descriptions

Here we compare our two descriptions in terms of running time and lattice size. We use the `MCTF` description with the `AMC` strategy, and the `SSTF` description with the `STF` strategy. These two strategies generate all possibles subsequences/supersequences. Results for the Wine-City dataset are given in Table 1. We can observe that `MCTF` is far faster than `SSTF`. It generates a lattice of 149 concepts in about 2 minutes from 500 sequences, while the `SSTF` description, stops with 1024 concepts generated in about 30 minutes from only 10 sequences. These descriptions are two different ways of representing the data. The `SSTF` description is clearly richer than the `MCTF` description that is the classical maximal common subsequences description extended to intervals, but `SSTF` is not adapted to huge datasets.

	data size	4	6	10	100	500	1000
$\sigma_{\mathrm{STF}}(\delta_{\mathrm{SSTF}})$	# concepts	16	33	**1024**			
	time(ms)	2878	11760	**1927039**			
	time/concept(ms)	179,87	356,36	**1881,87**			
$\sigma_{\mathrm{AMC}}(\delta_{\mathrm{MCTF}})$	# concepts	6	12	13	67	**149**	**132**
	time(ms)	267	771	838	17198	**141281**	**246437**
	time/concept(ms)	44.5	64.25	64.46	256.68	948.19	1866.94

Table 1. # of concepts and execution time for δ_{MCTF} and δ_{SSTF} descriptions for the Wine city dataset

[2] System for the Analysis of Numerical Traces for the development of Tourist Territories (Dispositif d'Analyse des Traces numériques pour la valorisation des Territoires Touristiques)

[3] It was planned to collect more data during the holidays on Mars and April, but unfortunately, this was impossible due to the world pandemic Covid-19

[4] https://www.laciteduvin.com/en

Comparison of strategies

We focus now on the SSTF description to compare the four strategies; STF, BTF, WATF and ATF. Recall that the STF strategy generates all possible supersequences whereas BTF, WATF and ATF focus on special supersequences (prefix, suffix, according to a window). Figure 5 shows the running time and the number of concepts generated using the Wine-City dataset. Compared to the STF strategy, we can clearly see that the other strategies are faster, and generate fewer concepts. The WATF strategy is the best in this example, especially with $w = 1$, with $w = 2$: the result approximates that of the BTF strategy. We run the BTF strategy with $card_{min}$ and $card_{max}$: we observe that the running time is better, and we obtain fewer concepts with $card_{min}$. The complexity of NextPriorityConcept depends on the size of the lattice, therefore a reduction to more relevant concepts also reduce the running time. Table 2 presents a comparaison between the

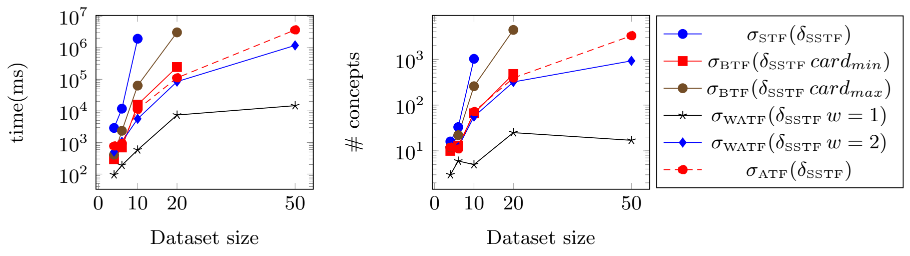

Fig. 5. Running time and size of the lattice using the δ_{SSTF} description and the four strategies for the Wine-City dataset

four strategies of the SSTF description with the GeoLuciole dataset. The BTF, ATF and WATF strategies are compared to the STF in terms of compression ratio i.e., the ratio between the number of concepts obtained with the STF strategy by the number of concepts obtained with each of the other strategies. Table 2 shows the effectiveness of our strategies in reducing the number of concepts. We can also observe that the compression ratio is improved as we increase the size of the data for all strategies. ATF strategy performs better and generates fewer concepts compared to BTF. The compression ratio is low with the WATF strategy, because the size of sequences is close to the window, and thus as we raise the windows the number of concepts get closer to the STF strategy. The behaviour of WATF strategy is linked to the average size of the sequences. The data analyst can variate parameters such as the cardinality for BTF, or the window for WATF, to generate only relevant concepts.

5 Conclusion

In this paper, we presented a sequence mining approach using the NextPriorityConcept algorithm. This algorithm allows a generic pattern computation through specific *descriptions* and *strategies*.

We presented two descriptions and five strategies for analysing interval-based sequences. The two descriptions represent two different approaches for representing a set of sequences. The first one MCTF is the classical maximal common

dataset size	4	5	6	7	10	15
	# Concepts					
σ_{STF}	16	32	64	**128**	**672**	**16640**
	Compression ratio					
$\sigma_{\text{BTF}}(card_{min})$	2.28	**4.57**	4.92	4.74	18.66	96.18
$\sigma_{\text{BTF}}(card_{max})$	**5.33**	2.66	**7.11**	3.45	28	92.96
σ_{ATF}	4	2.13	4	8.53	24.88	130
$\sigma_{\text{WATF}}(w = 1)$	1.77	3.2	3.76	**10.66**	**32**	**386.97**
$\sigma_{\text{WATF}}(w = 2)$	1.45	2.13	1.42	4.26	7.38	49.52

Table 2. # of concepts and compression ratio using δ_{SSTF} description and the four strategies `STF`, `BTF`, `ATF` and `WATF` for the GeoLuciole dataset

subsequences, whereas the second one `SSTF` provides a richer description of interval sequences. We presented one strategy for the `MCTF` description, and four strategies for the `SSTF` description that can be tested in a user-driven approach in order to generate fewer concepts and more relevant data. Therefore, we will focus on reducing the time complexity of our plugins, and create more configurable ones that respond the best to the particularity of the data we want to treat.

References

1. Allen, J.F.: An interval-based representation of temporal knowledge. In: IJCAI. vol. 81, pp. 221–226 (1981)
2. Barbut, M., Monjardet, B.: Ordres et classifications : Algèbre et combinatoire. Hachette, Paris (1970), 2 tomes
3. Bordat, J.P.: Calcul pratique du treillis de Galois d'une correspondance. Mathématiques et Sciences humaines **96**, 31–47 (1986)
4. Boukhetta, D.C., Bertet K., R.J.: Sequence mining using FCA and NextPriorityConcept algorithm. In: The 15th International Conference on Concept Lattices and Their Applications (CLA 2020) (2020)
5. Buzmakov, A., Egho, E., Jay, N., Kuznetsov, S.O., Napoli, A., Raïssi, C.: Fca and pattern structures for mining care trajectories (2013)
6. Buzmakov, A., Egho, E., Jay, N., Kuznetsov, S.O., Napoli, A., Raïssi, C.: On projections of sequential pattern structures (with an application on care trajectories) (2013)
7. Codocedo, V., Bosc, G., Kaytoue, M., Boulicaut, J.F., Napoli, A.: A proposition for sequence mining using pattern structures. In: International Conference on Formal Concept Analysis. pp. 106–121. Springer (2017)
8. Demko, Ch., Bertet, K., Faucher, C., Viaud, J.F., Kuznetsov, S.O.: NextPriorityConcept: A new and generic algorithm computing concepts from complex and heterogeneous data. arXiv preprint arXiv:1912.11038 (2019)
9. Ferré, S.: Systèmes d'information logiques : un paradigme logico-contextuel pour interroger, naviguer et apprendre. Doctorat, University of Rennes 1, France (Oct 2002)
10. Ferré, S.: Reconciling Expressivity and Usability in Information Access - From Filesystems to the Semantic Web. Habilitation, University of Rennes 1, France (november 2014)

11. Ferré, S.: Reconciling expressivity and usability in information access from file systems to the semantic web (2014)
12. Ferré, S., Ridoux, O.: A logical generalization of formal concept analysis. vol. 1867, pp. 371–384 (03 2000)
13. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: LNCS of International Conference on Conceptual Structures (ICCS'01). pp. 129–142 (2001)
14. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical foundations. Springer Verlag, Berlin (1999)
15. Gizdatullin, D., Baixeries, J., Ignatov, D.I., Mitrofanova, E., Muratova, A., Espy, T.H.: Learning interpretable prefix-based patterns from demographic sequences. In: International Conference on Intelligent Data Processing: Theory and Applications. pp. 74–91. Springer (2016)
16. Gizdatullin, D., Ignatov, D., Mitrofanova, E., Muratova, A.: Classification of demographic sequences based on pattern structures and emerging patterns. In: Supplementary Proceedings of 14th International Conference on Formal Concept Analysis, ICFCA. pp. 49–66 (2017)
17. Guyet, T., Quiniou, R.: Extracting temporal patterns from interval-based sequences. In: Twenty-Second International Joint Conference on Artificial Intelligence (2011)
18. Kam, P.s., Fu, A.W.C.: Discovering temporal patterns for interval-based events. In: International Conference on Data Warehousing and Knowledge discovery. pp. 317–326. Springer (2000)
19. Kaytoue, M.: Contributions to Pattern Discovery. Habilitation, University of Lyon, France (february 2020)
20. Kaytoue, M., Codocedo, V., Buzmakov, A., Baixeries, J., Kuznetsov, S.O., Napoli, A.: Pattern structures and concept lattices for data mining and knowledge processing. In: In Proceedings of ECML-PKDDl (2015)
21. Kaytoue, M., Duplessis, S., Kuznetsov, S.O., Napoli, A.: Two fca-based methods for mining gene expression data. In: International Conference on Formal Concept Analysis. pp. 251–266. Springer (2009)
22. Linding, C.: Fast concept analysis. In: Working with Conceptual Structures-Contributions to ICC. pp. 235–248 (2002)
23. Mannila, H., Toivonen, H., Verkamo, A.I.: Discovery of frequent episodes in event sequences. Data mining and knowledge discovery **1**(3), 259–289 (1997)
24. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.C.: Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In: icccn. p. 0215. IEEE (2001)
25. Srikant, R., Agrawal, R.: Mining sequential patterns: Generalizations and performance improvements, edbt (1996)
26. Sun, X., Orlowska, M.E., Zhou, X.: Finding event-oriented patterns in long temporal sequences. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 15–26. Springer (2003)
27. Wille, R.: Restructuring lattice theory : an approach based on hierarchies of concepts. Ordered sets pp. 445–470 (1982), i. Rival (ed.), Dordrecht-Boston, Reidel.
28. Winarko, E., Roddick, J.F.: Armada–an algorithm for discovering richer relative temporal association rules from interval-based data. Data & Knowledge Engineering **63**(1), 76–90 (2007)
29. Yan, X., Han, J., Afshar, R.: Clospan: Mining: Closed sequential patterns in large datasets. In: Proceedings of the 2003 SIAM international conference on data mining. pp. 166–177. SIAM (2003)