The representation of sequential patterns and their projections within Formal Concept Analysis

Aleksey Buzmakov, Elias Egho, Nicolas Jay, Sergei O. Kuznetsov, Amedeo Napoli, Chedy Raïssi

The representation of sequential patterns and their projections within Formal Concept Analysis

Aleksey Buzmakov[1,2], Elias Egho[1], Nicolas Jay[1], Sergei O. Kuznetsov[2], Amedeo Napoli[1], and Chedy Raïssi[1]

[1] LORIA (CNRS – Inria Nancy Grand Est – Université de Lorraine) Campus Scientifique, B.P. 70239, Vandœuvre-lès-Nancy, France
[2] Higher School of Economics – National Research University Pokrovskiy Bd. 11 – 109028 Moscow – Russia
{aleksey.buzmakov, elias.egho, nicolas.jay, napoli, chedy.raissi}@inria.fr, skuznetsov@hse.ru

Abstract. Nowadays data sets are available in very complex and heterogeneous ways. The mining of such data collections is essential to support many real-world applications ranging from healthcare to marketing. In this work, we focus on the analysis of *"complex"* sequential data by means of interesting sequential patterns. We approach the problem using an elegant mathematical framework: Formal Concept Analysis (FCA) and its extension based on *"pattern structures"*. Pattern structures are used for mining complex data (such as sequences or graphs) and are based on a subsumption operation, which in our case is defined with respect to the partial order on sequences. We show how pattern structures along with projections (i.e., a data reduction of sequential structures), are able to enumerate more meaningful patterns and increase the computing efficiency of the approach. Finally, we show the applicability of the presented method for discovering and analyzing interesting patients' patterns from a French healthcare data set of cancer patients. The quantitative and qualitative results are reported in this use case which is the main motivation for this work.

Keywords: formal concept analysis, pattern structures, sequential pattern structures, sequences

Introduction

Sequence data is largely present and used in many applications. Consequently, mining sequential patterns from sequence data has become an important and crucial data mining task. In the last two decades, the main emphasis has been on developing efficient mining algorithms and effective pattern representations [1–5]. However, the problem with traditional sequential pattern mining algorithms (and generally with all pattern enumeration algorithms) is that they generate a large number of frequent sequences while few of them are truly relevant. To echo this

challenge, some recent studies try to enumerate patterns using some alternative interestingness measures or by sampling representative patterns. A general idea, which is a framework of finding *statistically significant patterns*, is to extract patterns whose characteristic on a given measure, such as frequency, strongly deviates from its expected value under a null model. In this work, we focus on complementing the statistical approaches with a sound and adequate algebraic approach. That is, *can we develop a framework for enumerating only patterns of required types based solely on data lattices and its associated measures?*

The above question can be answered by addressing the problem of analyzing sequential data with the formal concept analysis framework (FCA), an elegant mathematical approach to data analysis [6], and pattern structures, an extension of FCA that handles complex data [7]. To analyze a dataset of "complex" sequences while avoiding the classical efficiency bottlenecks, we introduce and explain the usage of projections which are mathematical functions that respect certain algebraic properties. This novel usage of projections for sequences allows one to reduce the computational costs and the volume of enumerated patterns, avoiding thus the infamous "pattern flooding". In addition, we provide and discuss several measures to rank patterns with respect to their "interestingness", giving the order in which the patterns may be efficiently analyzed.

In this paper, we develop a novel, rigorous and efficient approach for working with sequential pattern structures in formal concept analysis. The main contributions of this work can be summarized as follows:

Pattern structure specification and analysis. We propose a novel way of dealing with sequences based on complex alphabets by mapping them to pattern structures. The genericity power provided by the pattern structures allows our approach to be directly instantiated with state-of-the-art FCA algorithms, making the final implementation flexible, accurate and scalable.

Projections of Sequential Pattern Structures. We introduce and discuss the notion of "projections" for sequential pattern structures. These mathematical objects significantly decrease (i.e., filter) the number of patterns, while preserving the most interesting ones for an expert. Projections are easily built to answer questions that an expert may have. Moreover, combinations of projections and concept stability index provide an efficient tool for the analysis of complex sequential datasets. The second advantage of projections is its ability to significantly decrease the complexity of a problem, saving thus computational time.

Experimental evaluations. We evaluate our approach on real sequence dataset of a regional healthcare system. The data set contains ordered sets of hospitalizations for cancer patients with information about the hospitals they visited, causes for the hospitalizations and medical procedures. These ordered sets are considered as sequences. The experiments reveal interesting (from a medical point of view) and useful patterns, and show the feasibility and the efficiency of our approach.

The paper is organized as follows. Section 1 introduces formal concept analysis and pattern structures. The specification of pattern structures for the case of

sequences is presented in Section 2. Section 3 describes projections of sequential pattern structures followed in Section 4.1 by the evaluation and experimations. Finally, related works are discussed before concluding the paper.

1 FCA and Pattern Structures

1.1 Formal Concept Analysis

FCA [6] is a formalism for data analysis. FCA starts with a formal context and builds a set of formal concepts organized within a concept lattice. A formal context is a triple (G, M, I), where G is a set of objects, M is a set of attributes and I is a relation between G and M, $I \subseteq G \times M$. In Table 1, a formal context is shown. A Galois connection between G and M is defined as follows:

$$A' = \{m \in M \mid \forall g \in A, (g, m) \in I\}, \qquad A \subseteq G$$
$$B' = \{g \in A \mid \forall m \in M, (g, m) \in I\}, \qquad B \subseteq M$$

The Galois connection maps a set of objects to the maximal set of attributes shared by all objects and reciprocally. For example, $\{g_1, g_2\}' = \{m_4\}$, while $\{m_4\}' = \{g_1, g_2, g_4\}$.

	m_1	m_2	m_3	m_4
g_1	x			x
g_2			x	x
g_3		x		
g_4			x	x

Table 1: A toy FCA context.

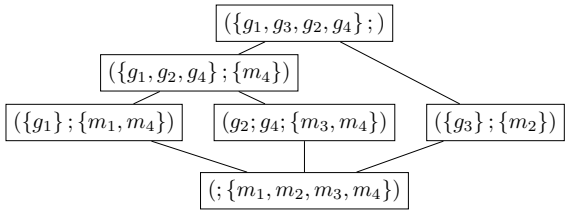


Fig. 1: Concept Lattice for the toy context

A formal concept is a pair (A, B), where A is a subset of objects, B is a subset of attributes, such that $A' = B$ and $A = B'$, where A is called the extent of the concept, and B is called the intent of the concept. A formal concept corresponds to a pair of maximal sets of objects and attributes, i.e. it is not possible to add an object or an attribute to the concept without violating the maximality property. For example a pair $(\{g_1, g_2, g_4\}, \{m_4\})$ is a formal concept.

Formal concepts can be partially ordered w.r.t. the extent inclusion (dually, intent inclusion). For example, $(\{g_1\}; \{m_1, m_4\}) \leq (\{g_1, g_2, g_4\}, \{m_4\})$. This partial order of concepts is shown in Figure 1.

1.2 Stability Index of a Concept

The number of concepts in a lattice for real-world tasks can be considerable. To find the most interesting subset of concepts, different measures can be used such as the stability of the concept [8] or the concept probability and separation [9]. These measures helps extracting the most interesting concepts and were shown to be reliable in noisy data [9].

Definition 1. *Given a concept c, the concept stability Stab(c) is the number of subsets of the concept extent (denoted Ext(c)), whose description is equal to the concept intent (denoted Int(c)). Hereafter $\wp(P)$ is a powerset of P.*

$$Stab(c) := \frac{|\{s \in \wp(Ext(c)) \mid s' = Int(c)\}|}{|\wp(Ext(c))|} \tag{1}$$

Stability measures how much the concept depends on the initial dataset. The bigger the stability the more objects can be deleted from the context without affecting the intent of the concept, i.e. the intent of the most stable concepts are likely to be a characteristic pattern of the studied data set.

To the best of our knowledge the fastest algorithm [10] processes a concept lattice L, in the worse case, in $O(|L|^2)$ where $|L|$ is the size of the concept lattice. For a big lattice, the stability calculation time can be high, and an estimation of the stability is useful. It should be noted that in a lattice the extent of any parent of a concept c is a superset of the extent of c, while the extent of any child is a subset. Given a concept c and its child, $\forall s \subseteq Ext(child), s'' \subseteq Ext(child) \subset Ext(c)$, i.e. $s' \neq Int(c)$. Thus, any subset of any child of the concept c should be excluded from the numerator in Equation 1.

$$Stab(c) \leq 1 - \max_{ch \in Children}(2^{-Diff(c,ch)}), \tag{2}$$

where $Diff(c, ch)$ is the extent difference between concept c and its child ch, $Diff(c, child) = |c.Ext \setminus child.Ext|$. Thus, if we would like to find stable concepts, with stability more than 0.97, we should select among concepts with

$$\max_{ch \in Children}(Diff(c, ch)) \geq -\log(1 - 0.97) = 5.06. \tag{3}$$

1.3 Pattern Structures

Although FCA applies to binary context, more complex data such as sequences or graphs can be directly processed as well. For that, pattern structures were introduced in [7].

Definition 2. *A pattern structure is a triple $(G, (D, \sqcap), \delta)$, where G is a set of objects, $(D, \sqcap)$ is a complete meet-semilattice of descriptions and $\delta : G \to D$ maps an object to a description.*

The lattice operation in the semilattice ($\sqcap$) corresponds to the similarity between two descriptions. Standard FCA can be presented in terms of a pattern structure. In this case, G is the set of objects, the semilattice of descriptions is $(\wp(M), \sqcap)$ and a description is a set of attributes, with the $\sqcap$ operation corresponding to the set intersection. If $x = \{a, b, c\}$ and $y = \{a, c, d\}$ then $x \sqcap y = x \cap y = \{a, c\}$. The mapping $\delta : G \to \wp(M)$ is given by, $\delta(g) = \{m \in M \mid (g, m) \in I\}$, and returns the description for a given object as a set of attributes.

The Galois connection between $\wp(G)$ and D is defined as follows:

$$A^\diamond := \prod_{g \in A} \delta(g), \qquad \text{for } A \subseteq G$$
$$d^\diamond := \{g \in G \mid d \sqsubseteq \delta(g)\}, \qquad \text{for } d \in D$$

The Galois connection makes a correspondence between sets of objects and descriptions. Given a set of objects A, $A^\diamond$ returns the description which is common to all objects in A. And given a description d, $d^\diamond$ is the set of all objects whose description subsumes d. More precisely, the partial order (or the subsumption order) on D ($\sqsubseteq$) is defined w.r.t. the similarity operation $\sqcap$: $c \sqsubseteq d \Leftrightarrow c \sqcap d = c$, and c is subsumed by d.

Definition 3. *A pattern concept of a pattern structure $(G, (D, \sqcap), \delta)$ is a pair (A, d) where $A \subseteq G$ and $d \in D$ such that $A^\diamond = d$ and $d^\diamond = A$, A is called the concept extent and d is called the concept intent.*

A pattern concept corresponds to the maximal set of objects A whose description subsumes the description d, where d is the maximal common description for objects in A. The set of all concepts can be partially ordered w.r.t. partial order on extents (dually, intent patterns, i.e $\sqsubseteq$), within a concept lattice. The stability of a pattern concept can be defined or estimated by the same procedure as for a formal concept, since the stability only depends on extents.

An example of pattern structures is given in Table 2, while the corresponding lattice is depicted in Figure 2.

2 Sequential Pattern Structures

2.1 An Example of Sequential Data

Patient	Trajectory
p^1	$\langle [H_1, \{a\}]; [H_1, \{c, d\}]; [H_1, \{a, b\}]; [H_1, \{d\}] \rangle$
p^2	$\langle [H_2, \{c, d\}]; [H_3, \{b, d\}]; [H_3, \{a, d\}] \rangle$
p^3	$\langle [H_4, \{c, d\}]; [H_4, \{b\}]; [H_4, \{a\}]; [H_4, \{a, d\}] \rangle$

Table 2: Toy sequential data on patient medical trajectories.

Imagine that we have medical trajectories of patients, i.e. sequences of hospitalizations, where every hospitalization is described by a hospital name and a set of procedures. An example of sequential data on medical trajectories with three patients is given in Table 2. There are a set of procedures $P = \{a, b, c, d\}$ a set of hospital names $T_H = \{H_1, H_2, H_3, H_4, CL, CH, *\}$, where hospital names are hierarchically organized (by level of generality), H_1 and H_2 are central hospitals (CH) and H_3 and H_4 are clinics (CL), and $*$ denotes the root of this hierarchy. The least common ancestor in this hierarchy is denoted as $h_1 \sqcap h_2$, for any $h_1, h_2 \in T_H$, i.e. $H_1 \sqcap H_2 = CH$. Every hospitalization is described with

one hospital name and may contain several procedures. The procedure order in each hospitalization is not important. For example, the first hospitalization $[H_2, \{c, d\}]$ for the second patient (p^2) was in hospital H_2 and during this hospitalization patient underwent procedures c and d. An important task is to find the "characteristic" sequences of procedures and associated hospitals in order to improve hospitalization planning, optimize clinical processes or detect anomalies.

The search for characteristic sequences can be performed by finding the most stable concepts in the lattice corresponding to a sequential pattern structure. For the simplification of calculations, subsequences are considered without "gaps", i.e the order of non consequent elements is not taken into account. It is reasonable in this task, because a hospitalization is a rather rare situation in the life of a patient, and, thus, in the most cases a hospitalization has a strong relation to the previous one. Next subsections define partial order on sequences and the corresponding pattern structures.

2.2 Partial Order on Complex Sequences

A sequence is constituted of elements from an alphabet. The classical subsequence matching task requires no special properties of the alphabet. Several generalization of the classical case were made by introducing subsequence relation based on itemset alphabet [11] or on multidimensional and multilevel alphabet [12]. Here, we generalize the previous cases, requiring for an alphabet to form a semilattice $(E, \sqcap_E)$[3]. This generalization allows one to process in a unified way all types of complex sequential data.

Definition 4. *Given an alphabet lattice* $(E, \sqcap_E)$*,*

1. $\langle\rangle$ *is a sequence;*
2. *for any sequence* $s = \langle e_1; ...; e_n\rangle$ *and any element* $e \in E$*,* $s \circ e = \langle e_1; ...; e_n; e\rangle$ *is a sequence.*

Definition 5. *A sequence* $t = \langle t_1; ...; t_k\rangle$ *is a subsequence of a sequence* $s = \langle s_1; ...; s_n\rangle$*, denoted* $t \leq s$*, iff* $k \leq n$ *and there exists* $j_1, ..j_k$ *such that* $1 \leq j_1 < j_2 < ... < j_k \leq n$ *and for all* $i \in \{1, 2, ..., k\}$*,* $t_i \sqsubseteq_E s_{j_i}$*.*

With complex sequences and such kind of subsequences the computation can be hard. Thus, for the sake of simplification, only "restricted" subsequences are considered, where only the order of consequent elements is taken into account, i.e. given j_1 in Definition 5, $j_i = j_{i-1} + 1$ for all $i \in \{2, 3, ..., k\}$. Below the word "subsequence" refers to "restricted" subsequence if not specified otherwise.

In the running example (Section 2.1), the alphabet is $E = T_H \times \wp(P)$ with the similarity operation $(h_1, P_1) \sqcap (h_2, P_2) = (h_1 \sqcap h_2, P_1 \cap P_2)$, where $h_1, h_2 \in T_H$ are hospitals and $P_1, P_2 \in \wp(P)$ are sets of procedures. Thus, the sequence ss^1 in

[3] It should be noted that in this paper we consider two semilattices, the first one is on the characters of the alphabet, $(E, \sqcap_E)$, and the second one is introduced by pattern structures, $(D, \sqcap)$.

Table 3 is a subsequence of p^1 in Table 2 because if we set $j_i = i+1$ (Definition 5) then $ss_1^1 \sqsubseteq p_{j_1}^1$ ('CH' is more general than H_1 and $\{c,d\} \subseteq \{c,d\}$), $ss_2^1 \sqsubseteq p_{j_2}^1$ (the same hospital and $\{b\} \subseteq \{b,a\}$) and $ss_3^1 \sqsubseteq p_{j_3}^1$ ('*' is more general than H_1 and $\{d\} \subseteq \{d\}$).

2.3 Sequential Meet-semilattice

Now, we can precisely define the sequential pattern structure that is used for representing and managing sequences. For that, we make an analogy with the pattern structures for graphs [13] where the meet-semilattice operation $\sqcap$ respects subgraph isomorphism. Thus, we introduce a sequential meet-semilattice respecting subsequence relation. Given an alphabet lattice $(E, \sqcap_E)$, $\mathfrak{S}$ is the set of all sequences based on $(E, \sqcap_E)$. $\mathfrak{S}$ is partially ordered w.r.t. Definition 5. $(D, \sqcap)$ is a semilattice on sequences $\mathfrak{S}$, where $D \subseteq \wp(\mathfrak{S})$ such that if $d \in D$ contains a sequence s then all subsequences of s should be included into d, $\forall s \in d, \nexists \tilde{s} \leq s : \tilde{s} \notin d$, and similarity operation is the set intersection for two set of sequences. Given two patterns $d_1, d_2 \in D$, the set intersection operation ensures that if a sequence s belongs to $d_1 \sqcap d_2$ then any subsequence of s belongs to $d_1 \sqcap d_2$ and thus $(d_1 \sqcap d_2) \in D$. As the set intersection operation is idempotent, commutative and associative, $(D, \sqcap)$ is a valid semilattice.

However, the set of all possible subsequence for a given sequence can be rather large. Thus, it is more efficient and representable to keep a pattern $d \in D$ as a set of all maximal sequences $\tilde{d}$, $\tilde{d} = \{s \in d \mid \nexists s^* \in d : s^* \geq s\}$. Furthermore, every pattern will be given only by the set of all maximal sequences. For example, $\{p^2\} \sqcap \{p^3\} = \{ss^6, ss^7, ss^8\}$ (see Tables 2 and 3), i.e. $\{ss^6, ss^7, ss^8\}$ is the set of all maximal sequences specifying the intersection result of two sets of sequences specified by sequences p^2 and p^3, correspondingly $\{ss^6, ss^7, ss^8\} \sqcap \{p^1\} = \{ss^4, ss^5\}$. Note that representing a pattern by the set of all maximal sequences allows for an efficient implementation of the intersection "$\sqcap$" of two patterns (see Section 4.1 for more details).

Example 1. The sequential pattern structure for our example (Subsection 2.1) is $(G, (D, \sqcap), \delta)$, where $G = \{p^1, p^2, p^3\}$, $(D, \sqcap)$ is the semilattice of sequential descriptions, and δ is the mapping associating an object in G to a description in D shown in Table 2. Figure 2 shows the resulting lattice of sequential pattern concepts for this particular pattern structure $(G, (D, \sqcap), \delta)$.

3 Projections of Sequential Pattern Structures

Pattern structures can be hard to process due to the usually large number of concepts in the concept lattice, the complexity of the involved descriptions and the similarity operation. Moreover, a given pattern structure can produce a lattice with a lot of patterns which are not interesting for an expert. *Can we save computational time by avoiding to compute useless patterns?* Projections of pattern structures "simplify" to some degree the computation and allow one to work

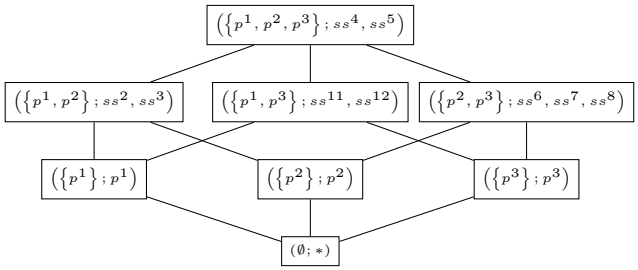


Fig. 2: The concept lattice for the pattern structure given by Table 2. Concept intents reference to sequences in Tables 2 and 3.

	Subsequences
ss^1	$\langle[CH, \{c, d\}]; [H_1, \{b\}]; [*, \{d\}]\rangle$
ss^3	$\langle[CH, \{\}]; [*, \{d\}]; [*, \{a\}]\rangle$
ss^5	$\langle[*, \{a\}]\rangle$
ss^7	$\langle[CL, \{d\}]; [CL, \{\}]\rangle$
ss^9	$\langle[CH, \{c, d\}]\rangle$
ss^{11}	$\langle[*, \{c, d\}]; [*, \{b\}]\rangle$

	Subsequences
ss^2	$\langle[CH, \{c, d\}]; [*, \{b\}]; [*, \{d\}]\rangle$
ss^4	$\langle[*, \{c, d\}]; [*, \{b\}]\rangle$
ss^6	$\langle[*, \{c, d\}]; [CL, \{b\}]; [CL, \{a\}]\rangle$
ss^8	$\langle[CL, \{\}]; [CL, \{a, d\}]\rangle$
ss^{10}	$\langle[CL, \{b\}]; [CL, \{a\}]\rangle$
ss^{12}	$\langle[*, \{a\}]; [*, \{d\}]\rangle$

Table 3: Subsequences of patient sequences in Table 2.

with a reduced description. In fact, projections can be considered as filters on patterns respecting mathematical properties. These properties ensure that the projection of a semilattice is a semilattice and that projected concepts have a correspondence to original ones. Moreover, the stability measure of projected concepts never decreases w.r.t the original concepts [7].

A possible projection for sequential pattern structures comes from the following observation. In many cases it may be more interesting to analyze long subsequences. We call these projections *Minimal Length Projection* (MLP) and they depend on the minimal allowed length l for the sequences in a pattern. To project a pattern structure w.r.t. MLP, a pattern should be substituted with the pattern where any sequence of length less then l is removed.

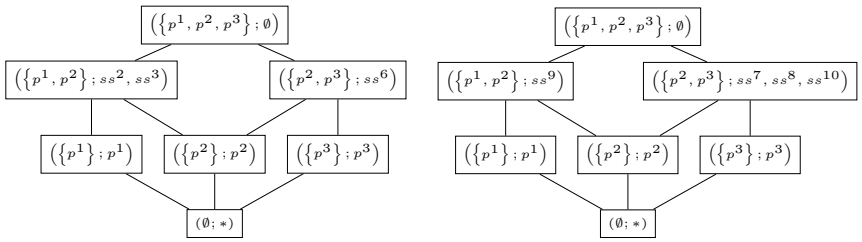
Example 2. If we set the minimal length threshold to 3, then there is only one maximal common subsequence ss^6 in Table 3 between p^2 and p^3 in Table 2, while ss^7 and ss^8 are too short to be considered. Figure 3a shows the corresponding projected lattice for the pattern structure in Table 2.

Another important type of projections is connected to a variation of the lattice alphabet $(E, \sqcap_E)$. The simplest variation is to ignore of certain fields in the elements. For example, if a hospitalization is described by a hospital name and a set of procedures, then procedures can be ignored in similarity computation. For that, in any element a set of procedures can be substituted by $*$ which is the most general element of the taxonomy of hospitals.

Another variation of the alphabet, is to require that some field(s) should not be empty. For example, we want to find patterns with non-empty set of procedures, or we want to have information about hospital (the element $*$ of hospital taxonomy is not allowed in an element of a sequence). Such variations are

easy to realize within our approach. For this, computing the similarity operation between elements of the alphabet, one should check if the result contains empty fields and, if yes, should substitute the result by $\perp$. This variation is useful, as shown in the experimental section, but this variation is rather difficult to define within classical frequent sequence mining approaches.

Example 3. An expert is interested in finding sequential patterns describing how a patient changes hospitals, without interest in procedures. Thus, any element of the alphabet lattice containing a non-empty set of procedures is projected to the corresponding element with the same hospital and an empty set of procedures. Moreover, an expert is interested in finding sequential patterns containing information about the hospital in every hospitalization, i.e. hospital field in the patterns cannot be $*$, e.g. ss^5 is an invalid pattern, while ss^6 is a valid pattern in Table 3. Figure 3b shows the lattice corresponding to the projected pattern structure (Table 2) by changing the alphabet semilattice.



(a) MLP projection, $l = 3$ (b) Projection removing '*' hospitals

Fig. 3: The projected concept lattices for the pattern structure given by Table 2. Concept intents refer to the sequences in Tables 2 and 3.

4 Sequential Pattern Structure Evaluation

4.1 Implementation

Nearly any state-of-the-art FCA algorithm can be adapted to process pattern structures instead of standard FCA contexts. We adapted `AddIntent` algorithm [14], as the lattice structure is important for us to calculate stability (see the algorithm for calculating stability in [10]). To adapt the algorithm to our needs, every set intersection operation on attributes should be substituted with semilattice operation $\sqcap$ on corresponding patterns, while every subset checking operation should be substituted with semilattice order checking $\sqsubseteq$, in particular all $(\cdot)'$ should be substituted with $(\cdot)^\diamond$.

The next question is how the semilattice operations ($\sqcap$, $\sqsubseteq$) can be implemented. Given two sets of sequences $S = \{s^1, ...s^n\}$ and $T = \{t^1, ..., t^m\}$, the similarity between these sets, $S \sqcap T$, is calculated according to Section 2.3, i.e. maximal sequences among all common subsequences for any pair of s^i and t^j.

To find all common subsequences of two sequences, the following observations can be useful. If $ss = \langle ss_1; ...; ss_l \rangle$ is a subsequence of $s = \langle s_1; ...; s_n \rangle$ with

$j_i^s = k^s + i$ (Definition 5: k^s is the index difference from which ss is a subsequence of s) and a subsequence of $t = \langle t_1; ...; t_m \rangle$ with $j_i^t = k^t + i$ (likewise), then for any index $i \in \{1, 2, ..., l\}$, $ss_i \sqsubseteq_E (s_{j_i^s} \sqcap t_{j_i^t})$. Thus, to find all maximal common subsequences between s and t, we first align s and t in all possible ways. For each alignment of s and t we compute the resulting intersection. Finally, we keep only the maximal intersected subsequences.

Let us consider two possible alignments of s^1 and s^2:

$s^1 = \langle \{a\}\,; \{c,d\}\,; \{b,a\};\ \{d\}\ \rangle$	$s^1 = \langle \{a\}\,; \{c,d\}; \{b,a\};\ \{d\}\ \rangle$
$s^2 = \langle \{c,d\}; \{b,d\}\,; \{a,d\} \rangle$	$s^2 = \langle \{c,d\}; \{b,d\}; \{a,d\} \rangle$
$ss^l = \langle\ \emptyset\ ;\ \{d\}\ \rangle$	$ss^r = \langle \{c,d\};\ \{b\}\ ;\ \{d\}\ \rangle$

The left intersection ss^l is not retained, as it is not maximal, while the right intersection ss^r is kept.

4.2 Experiments and Discussion

The experiments are carried out on an "Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz" computer with 8Gb of memory under the Ubuntu 12.04 operating system. The algorithms are not parallelized and are coded in C++.

First, the public available database from UCI repository on anonymous web data is used as a benchmark data set for scalability tests. This database contains around 10^6 transactions, and each transaction is a sequence based on "simple" alphabet, i.e. with no order on the elements. The overall time changes from 37279 seconds for the sequences of length $MLP \geq 5$ upto 97042 seconds for the sequences of length $MLP \geq 3$. For more details see the web-page.[4]

Our use-case data set comes from PMSI[5], a French healthcare system [15]. Each elements of a sequence has a "complex" nature. The dataset contains 2400 patients suffering from *cancer*. Every patient is described as a sequence of hospitalizations without any timestamps. The hospitalization is a tuple with three elements: (i) healthcare institution (e.g. university hospital of Paris (CHU_{Paris})), (ii) reason of the hospitalization (e.g. a cancer disease), and (iii) set of medical procedures that the patient underwent. An example of a medical trajectory of a patient is provided below:

$$\langle [\mathrm{CHU}_{Paris}, \mathrm{Cancer}, \{P_1, P_2\}]; [\mathrm{CH}_{Lyon}, \mathrm{Chemo}, \{\}]; [\mathrm{CH}_{Lyon}, \mathrm{Chemo}, \{\}] \rangle\,.$$

.This sequence represents a patient trajectory with three hospitalizations. It expresses that one patient was first admitted to the university hospital of Paris (CHU_{Paris}) for a cancer problem as reason, and underwent procedures P_1 and P_2. Then he had two consequent hospitalizations in Central hospital of Lyon (CH_{Lyon}) for doing chemotherapy with no additional procedures. We substituted the same consequent hospitalizations by the number of repetitions. With this substitution, we have shorter and more understandable trajectory. For example, the above pattern should be transformed into two hospitalizations where the first hospitalization repeats once and the second twice:

$$\langle [\mathrm{CHU}_{Paris}, \mathrm{Cancer}, \{P_1, P_2\}][1]; [\mathrm{CH}_{Lyon}, \mathrm{Chemo}, \{\}][2] \rangle\,.$$

[4] `http://www.loria.fr/~abuzmako/PKDD2013/experiment-uci.html`

[5] Programme de Médicalisation des Sytèmes d'Information.

The healthcare institution was associated with a geographical taxonomy of 4 levels of granularity (i.e. Root, Region, Department and Healthcare institution). This taxonomy has 304 node. Where hospitalization reasons and medical procedures are simple sets without any associated subsumption relation. The set of hospitalisation reasons has 1939 items and the set of medical procedures has 723 items. The distribution of sequence lengths' is shown in Figure 4.

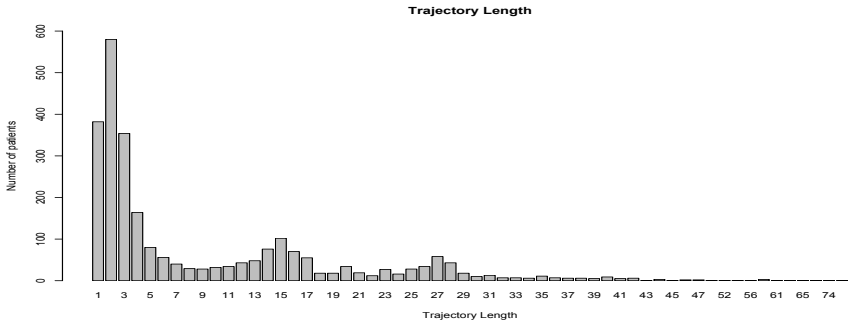


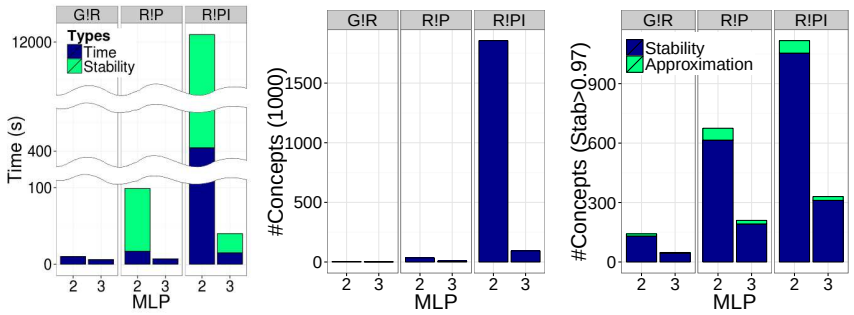
Fig. 4: The length distribution of sequences in the dataset

For this dataset the computation of the whole lattice is infeasible. However our medical expert is not interested in all possible patterns, but rather in patterns which answer his analysis question(s). First of all, an expert may know the minimal size of sequences he is interested in, i.e. setting the MLP projection. If an expert is interested in sequential patterns, the patterns of length 1 are unlikely to be of interest for him (knowing that people go to hospitals when they are sick is not a valuable new knowledge). Thus, we use the MLP projection of length 2 and 3 and take into account the small average length of the sequences.

Figure 5 shows computational time, the number of concepts in the lattice, and the number of stable concepts for different projections. For example, computation of the lattice for projection with name "`R!PI`" takes 400 seconds and calculation of stability for every concept in the lattice takes 12000 seconds (Figure 5a), the size of the lattice is $1.8 \cdot 10^6$ concepts (Figure 5b) where around 1000 concepts have stability index more than 0.97 while an approximated solution to find stable concepts (Formula 3) return only few unstable ones (Figure 5c).

Table 4 shows some interesting concept intents with the corresponding support and ranking w.r.t. to concept stability. For example the concept #1 is obtained under the projection `R!P` for $MLP \geq 2$, with the intent containing a *Cancer* hospitalization followed by a *Chemotherapy*. This concept is the most stable concept in the lattice for the given projection, and the cardinality of the concept extent is 452 patients.

The first question that the analyst would like to address here is *"What are the sequential relations between hospitalization reasons and corresponding procedures?"*. To answer this question, we are not interested about healthcare institutions. Thus, any alphabet element should be projected by substituting healthcare institution fields by the '*' hospital. As hospitalization reason is important in



(a) Calculation time (b) Concepts (c) Stable concepts

Fig. 5: Parameters of the result for different projections.

#	Projection	Intent	Stab. Rank	Support
1	`R!P2`	$\langle[Cancer, \{\}]; [Chemo, \{\}]\rangle$	1	452
2	`R!P2`	$\langle[Cancer, \{App.\}]; [Ch.Prep, \{\}]; [Chemo, \{\}]\rangle$	4	293
3	`R!P3`	$\langle[Cancer, \{App.\}]; [Ch.Prep, \{\}]; [Chemo, \{\}]\rangle$	2	293
4	`R!PI3`	$\langle[Cancer, \{\}] * 1; [Ch.Prep, \{\}] * 1; [Chemo, \{\}] * [8, 24]\rangle$	4	193
5	`G!R!3`	$\langle[Bourgogne, Cancer]; [Bourgogne, Ch.Prep]; [A\ clinic\ in\ Dijon, Chemo]\rangle$	5	29

Table 4: Interesting concepts, for different projections. `Chemo` is chemotherapy, `Ch.Prep` is preparation for chemotherapy, `App.` is an operation for appendicitis.

each hospitalization so any alphabet element without the hospitalization reason is of no use and should be projected to the bottom element $\perp$ of the alphabet lattice. This is a projection of the hospitalization alphabet and, thus, gives us the projection of the pattern structure. Such projections are called `R!P2` or `R!P3`, meaning that we consider the fields "Reason" and "Procedures", while the reason should not be empty and the MLP parameter is 2 or 3. *Patterns #1 and #2* are obtained under the `R!P2` projection. *Pattern #1* trivially states that in the Bourgogne region, *"When a patient has a cancer, he undergoes chemotherapy"* which is one of the standard procedure followed by french physicians. This pattern gives a general viewpoint about the cancer treatment.

The next accurate question is *"How do the doctors detect colon cancer?"*. *Pattern #2 and #3* answer our question, they show that cancer is detected during an appendicitis surgical intervention which is followed by preparation for chemotherapy and chemotherapy itself. These two patterns highlight a recently discovered fact that acute appendicitis has been shown to occur antecedent to cancer [16] within three years because of a carcinoma in colon or rectum. Therefore, any patient over the age of 40 presenting with acute appendicitis is carefully checked for carcinoma in the colon. We can also note that *patterns #2 and #3* have the same form, but pattern #3 was obtained under `R!P3` projection, and has higher stability rank (2) than pattern #2 (4). *Pattern #4* can help healthcare managers and doctors quantify on average the number of usually required chemotherapies for a patient. It shows that *"After detecting cancer, the patients require chemotherapy between 8 and 24 times in many cases"*. This pattern has

been extracted by the projection `R!PI3` (i.e. involving interval information). Figure 5a and 5b shows that this task is time and memory expensive.

"Where do patients prefer staying (i.e. hospital location) during their treatment, and why ?". To answer this expert question, we consider only healthcare institutions and reason fields, requiring both to "have" some information, i.e. projections `G!R!2` and `G!R!3`. Nearly all patterns show that patients usually prefer to be treated in the same region, without any preferences about the exact hospital. However, *pattern #5* obtained under `G!R!3` projection shows us that a good proportion of patients prefer to undergo Chemotherapy in *a precise private clinic in Dijon*[6], while cancer detection and preparation is usually done everywhere in the Bourgogne region, depending on the patient preferences.

Figure 5 shows that with the increase of the minimal length of a pattern (from 2 to 3), the memory and the time consumption is reduced, in some cases significantly. Figure 5a shows that the precise stability calculation can take more time than the calculation of the lattice, correspondingly the lattice computation for projection `R!PI2` takes 400 seconds, while the stability calculation procedure takes 30 times more (12000). However, the approximation of concept stability that is presented in the beginning of the paper (Formula 3) is fast and does filter only few unstable concepts (less then 5%), while finding all stable (Figure 5c).

5 Related Work

The most widely used approach for analyzing sequences is, probably, mining frequent subsequences [2–4, 12]. The most general type of sequences among them is described in [12], where every element of the sequence is multidimensional and multilevel, i.e. every element can be characterized by several components, and for every component a kind of hierarchy can be applied. Then, every element e in a sequences is substituted by all the most specific elements, which are more general than e and, thus, the task is reduced to sequences of itemsets. In our approach, the elements of sequences are considered to be even more general, for example, beside multidimensional and multilevel sequences, sequences of graphs fall under the definition. Moreover, frequent subsequences mining gives birth to a lot of subsequences which can be hardly analyzed by an expert.

Formal Concept Analysis (FCA) [6] allows one to measure several indexes, related to the importance of a pattern. One of the FCA approaches is [17], where authors process sequential dataset based on "simple" alphabet without involving any partial order on it, in this approach maximal common subsequences (with no gaps) were mined and analyzed with FCA. In the work [11] only sequences of itemsets were considered. All closed subsequences were, first, mined and then regrouped by specialized algorithm in order to obtain a lattice similar to the FCA lattice. Comparing with both approaches, our approach suggests a more general definition of sequences and, thanks to pattern structures, there is no 'premining' step to find frequent (or maximal) subsequences. This allows us to apply different "projections" specializing the request of an expert and simplifying

[6] the name of the clinic is anonymized.

the calculation. In addition, in our approach nearly all state-of-the-art FCA algorithms can be used in order to efficiently process a dataset.

Another type of the FCA generalization is based on well-known LCM algorithm [18]. Authors of [19] process multirelational databases by extending LCM. Although this approach perfectly works for special kinds of multirelational databases, it cannot process sequential datasets for the same reason why it cannot process graph datasets in the settings of frequent graph mining.

Projections is an essential part of our approach and can be considered as a special kind of constraints. Many constraints that do not change subsequence relation have a corresponding projection. Authors of survey [20] (Section 5) enumerate 8 types of constraints, two of them, i.e. "item constraint" and "length constraint", correspond the introduced projections.

Conclusion

In this paper, we present a novel approach for analyzing complex sequential data. This kind of data is a generalization of data considered in previous approaches. The approach is based on the formalism of sequential pattern structures and projections. Our work complements the general orientations towards *statistically significant patterns* by presenting strong formal results on the notion of interestingness from a concept lattice point of view. Using pattern structures leads to the construction of a pattern concept lattice, which does not require the setting of a support threshold, as usually needed in classical sequential pattern mining. Moreover, the use of projections gives a lot of flexibility especially for mining and interpreting special kinds of patterns.

Our framework was tested on a large-scale benchmark dataset and on a real-world dataset with patient hospitalization trajectories. Interesting patterns answering to the questions of an expert are extracted and interpreted, showing the feasibility and usefulness of the approach and the importance of the stability as a pattern-selection procedure.

For future work, we are planning to more deeply investigate projections, their potentialities w.r.t. the types of patterns. Finally, another research direction is mining of association rules or building a Horn approximation [21] from the stable part of the pattern lattice.

Acknowledgements: this research received funding from the Basic Research Program at the National Research University Higher School of Economics (Russia) and from the BioIntelligence project (France).

References

1. Han, J., Pei, J., Mortazavi-Asl, B., Chen, Q., Dayal, U., Hsu, M.: FreeSpan: frequent pattern-projected sequential pattern mining. In: Proc. of the 6th ACM SIGKDD Int'l Conf. on Knowledge discovery and data mining. (2000) 355–359

2. 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: 17th International Conference on Data Engineering. (2001) 215–226
3. Yan, X., Han, J., Afshar, R.: CloSpan: Mining Closed Sequential Patterns in Large Databases. In: Proc. of SIAM Int'l Conf. Data Mining (SDM'03). (2003) 166–177
4. Ding, B., Lo, D., Han, J., Khoo, S.C.: Efficient Mining of Closed Repetitive Gapped Subsequences from a Sequence Database. In: Proc. of IEEE 25th International Conference on Data Engineering, IEEE (March 2009) 1024–1035
5. Raïssi, C., Calders, T., Poncelet, P.: Mining conjunctive sequential patterns. Data Min. Knowl. Discov. **17**(1) (2008) 77–93
6. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. 1st edn. Springer, Secaucus, NJ, USA (1997)
7. Ganter, B., Kuznetsov, S.O.: Pattern Structures and Their Projections. In Delugach, H., Stumme, G., eds.: Conceptual Structures: Broadening the Base SE - 10. Volume 2120 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2001) 129–142
8. Kuznetsov, S.O.: On stability of a formal concept. Annals of Mathematics and Artificial Intelligence **49**(1-4) (2007) 101–115
9. Klimushkin, M., Obiedkov, S.A., Roth, C.: Approaches to the Selection of Relevant Concepts in the Case of Noisy Data. In: Proc. of the 8th International Conference on Formal Concept Analysis. ICFCA'10, Springer (2010) 255–266
10. Roth, C., Obiedkov, S., Kourie, D.G.: On succinct representation of knowledge community taxonomies with formal concept analysis A Formal Concept Analysis Approach in Applied Epistemology. International Journal of Foundations of Computer Science **19**(02) (April 2008) 383–404
11. Casas-Garriga, G.: Summarizing Sequential Data with Closed Partial Orders. In: Proc. of the 5th SIAM Int'l Conf. on Data Mining (SDM'05). (2005)
12. Plantevit, M., Laurent, A., Laurent, D., Teisseire, M., Choong, Y.W.: Mining multidimensional and multilevel sequential patterns. ACM Transactions on Knowledge Discovery from Data **4**(1) (January 2010) 1–37
13. Kuznetsov, S.O.: Learning of Simple Conceptual Graphs from Positive and Negative Examples. In Żytkow, J., Rauch, J., eds.: Principles of Data Mining and Knowledge Discovery SE - 47. Volume 1704 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (1999) 384–391
14. Merwe, D.V.D., Obiedkov, S., Kourie, D.: AddIntent: A new incremental algorithm for constructing concept lattices. In Goos, G., Hartmanis, J., Leeuwen, J., Eklund, P., eds.: Concept Lattices. Volume 2961. Springer (2004) 372–385
15. Fetter, R.B., Shin, Y., Freeman, J.L., Averill, R.F., Thompson, J.D.: Case mix definition by diagnosis-related groups. Med Care **18**(2) (February 1980) 1–53
16. Arnbjörnsson, E.: Acute appendicitis as a sign of a colorectal carcinoma. Journal of Surgical Oncology **20**(1) (May 1982) 17–20
17. Ferré, S.: The Efficient Computation of Complete and Concise Substring Scales with Suffix Trees. In Kuznetsov, S.O., Schmidt, S., eds.: Formal Concept Analysis SE - 7. Volume 4390 of Lecture Notes in Computer Science. Springer (2007) 98–113
18. Uno, T., Asai, T., Uchida, Y., Arimura, H.: An efficient algorithm for enumerating closed patterns in transaction databases. Discovery Science (2004) 16–31
19. Garriga, G., Khardon, R., De Raedt, L.: Mining closed patterns in relational, graph and network data. Ann. Math. Artif. Intell. (November 2012) 1–28
20. Mooney, C.H., Roddick, J.F.: Sequential pattern mining – approaches and algorithms. ACM Computing Surveys **45**(2) (February 2013) 1–39

21. Balcázar, J.L., Casas-Garriga, G.: On Horn Axiomatizations for Sequential Data. In: ICDT. (2005) 215–229