Proceedings of the 5th International Workshop "What can FCA do for Artificial Intelligence?", FCA4AI 2016 (co-located with ECAI 2016, The Hague, Netherlands, August 30th 2016)

Sergei O. Kuznetsov, Amedeo Napoli, Sebastian Rudolph

Workshop Notes

Fifth International Workshop

"What can FCA do for Artificial Intelligence?"

FCA4AI 2016

European Conference on Artificial Intelligence

ECAI 2016

August 30 2016

The Hague, Netherlands

Editors

Sergei O. Kuznetsov (NRU HSE Moscow)

Amedeo Napoli (LORIA Nancy)

Sebastian Rudolph (TU Dresden)

http://fca4ai.hse.ru/2016/

Preface

The four preceding editions of the FCA4AI Workshop showed that many researchers working in Artificial Intelligence are deeply interested by a well-founded method for classification and mining such as Formal Concept Analysis (see `http://www.fca4ai.hse.ru/`). The first edition of FCA4AI was co-located with ECAI 2012 in Montpellier, the second one with IJCAI 2013 in Beijing, the third one with ECAI 2014 in Prague, and finally the forth and last one with IJCAI 2015 in Buenos Aires. In addition, all the proceedings of these preceding editions have been published as CEUR Proceedings (`http://ceur-ws.org/Vol-939/`, `http://ceur-ws.org/Vol-1058/`, `http://ceur-ws.org/Vol-1257/` and `http://ceur-ws.org/Vol-1430/`).

This year, the fifth workshop has again attracted many different researchers working on actual and important topics, e.g. theory, fuzzy FCA, dependencies, classification, mining of linked data, navigation, visualization, and various applications. This shows the diversity and the richness of the relations between FCA and AI.

Formal Concept Analysis (FCA) is a mathematically well-founded theory aimed at data analysis and classification. FCA allows one to build a concept lattice and a system of dependencies (implications) which can be used for many AI needs, e.g. knowledge discovery, learning, knowledge representation, reasoning, ontology engineering, as well as information retrieval and text processing. As we can see, there are many "natural links" between FCA and AI. Recent years have been witnessing increased scientific activity around FCA, in particular a strand of work emerged that is aimed at extending the possibilities of FCA w.r.t. knowledge processing, such as work on pattern structures and relational context analysis. These extensions are aimed at allowing FCA to deal with more complex than just binary data, both from the data analysis and knowledge discovery points of view and as well from the knowledge representation point of view, including, e.g., ontology engineering. All these investigations provide new possibilities for AI activities in the framework of FCA. Accordingly, in this workshop, we are interested in two main issues:

- How can FCA support AI activities such as knowledge processing (knowledge discovery, knowledge representation and reasoning), learning (clustering, pattern and data mining), natural language processing, and information retrieval.
- How can FCA be extended in order to help AI researchers to solve new and complex problems in their domains.

The workshop is dedicated to discuss such issues. This year, the papers submitted to the workshop were carefully peer-reviewed by three members of the program committee and 14 papers with the highest scores were selected. We thank all the PC members for their reviews and all the authors for their contributions.

The Workshop Chairs

Sergei O. Kuznetsov
National Research University Higher School of Economics, Moscow, Russia

Amedeo Napoli
LORIA (CNRS – Inria Nancy Grand Est – Université de Lorraine), Vandoeuvre les Nancy, France

Sebastian Rudolph
Technische Universität Dresden, Germany

Program Committee

Mehwish Alam (Université de Paris-Nord, France)

Gabriela Arevalo (Universidad Nacional de Quilmes, Argentina)

Jaume Baixeries (UPC Barcelona, Catalunya)

Karell Bertet (Université de La Rochelle, France, Germany)

Aleksey Buzmakov (National Research University HSE Perm, Russia)

Mathieu D'Aquin (Open University, UK)

Florent Domenach (University of Nicosia, Cyprus)

Sébastien Ferré (IRISA, Rennes, France)

Marianne Huchard (LIRMM/Université de Montpellier, France)

Dmitry I. Ignatov (National Research University HSE Moscow, Moscow, Russia)

Yuri Kashnitsky (National Research University HSE Moscow, Russia)

Mehdi Kaytoue (INSA-LIRIS Lyon, France)

Jan Konecny (Palacky University, Olomouc, Czech Republic)

Florence Le Ber (ENGEES/Université de Strasbourg, France)

Nizar Messai (Université de Tours, France)

Sergei A. Obiedkov (NRU Higher School of Economics, Moscow, Russia)

Jan Outrata (Palacky University, Olomouc, Czech Republic)

Jean-Marc Petit (INSA-LIRIS Lyon, France)

Uta Priss (Ostfalia University of Applied Sciences, Wolfenbüttel, Germany)

Christian Săcărea (Babes-Bolyai University, Cluj-Napoca, Romania)

Baris Sertkaya (Frankfurt University of Applied Sciences, Germany)

Diana Troancă (Babes-Bolyai University, Cluj-Napoca, Romania)

Renato Vimiero (UFPE Recife, Brazil)

Contents

Invited Talk
Constraint Programming for Constrained Clustering

Christel Vrain

Université d'Orléans
INSA Centre Val de Loire, LIFO EA 4022
45067 Orléans, France
christel.vrain@univ-orleans.fr

Abstract. Several works have shown the interest of declarative frameworks, such as Constraint Programming, SAT, Integer Linear Programming, for Data Mining [9,10,12,8,14,15]. Relying on Constraint Programming (CP) has several advantages: first, its inherent declarativity allows to easily model constraints on the Data Mining problem at hand, second, CP has the ability either to enumerate all the solutions or to find an optimal solution, in case of an optimization problem. Moreover, it relies on constraint propagation, which often allows to efficiently prune the search space.
I will mainly focus on constrained clustering [16,7]: user constraints are put on the solutions, thus allowing to get a clustering closer to the one expected by the user.
After giving some backgrounds on CP, I will present the seminal work of [9] on CP for itemset mining, its extension to k-pattern set mining under constraints [13] and its application to conceptual clustering.
The talk will then mainly be focused on distance-based constrained clustering. I will show how we have modeled this task in CP [1,4,3], the difficulties we have had to face, the solutions we have developed [2,5]. The interest of relying on CP will be illustrated through several extensions [4,6,11].
The work on distance-based constrained clustering in CP is a joint work with T.B.H. Dao and K.C. Duong.

References

1. Thi-Bich-Hanh Dao, Kanh-Chuong Duong, and Christel Vrain. A Declarative Framework for Constrained Clustering. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 419–434, 2013.
2. Thi-Bich-Hanh Dao, Khanh-Chuong Duong, and Christel Vrain. A Filtering Algorithm for Constrained Clustering with Within-Cluster Sum of Dissimilarities

Criterion. In *Proceedings of the 25th International Conference on Tools with Artificial Intelligence*, pages 1060–1067, 2013.

3. Thi-Bich-Hanh Dao, Khanh-Chuong Duong, and Christel Vrain. Clustering sous contraintes en ppc. *Revue d'Intelligence Artificielle*, 28(5):523–545, 2014.
4. Thi-Bich-Hanh Dao, Khanh-Chuong Duong, and Christel Vrain. Constrained clustering by constraint programming. *Artificial Intelligence Journal*, 2015.
5. Thi-Bich-Hanh Dao, Khanh-Chuong Duong, and Christel Vrain. Constrained minimum sum of squares clustering by constraint programming. In *Principles and Practice of Constraint Programming - 21st International Conference, CP 2015, Cork, Ireland, August 31 - September 4, 2015, Proceedings*, pages 557–573, 2015.
6. Thi-Bich-Hanh Dao, Christel Vrain, Khanh-Chuong Duong, and Ian Davidson. A framework for actionable clustering using constraint programming. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 453–461, 2016.
7. Ian Davidson and S. S. Ravi. The Complexity of Non-hierarchical Clustering with Instance and Cluster Level Constraints. *Data Mining Knowledge Discovery*, 14(1):25–61, 2007.
8. Ian Davidson, S. S. Ravi, and Leonid Shamis. A SAT-based Framework for Efficient Constrained Clustering. In *Proceedings of the 10th SIAM International Conference on Data Mining*, pages 94–105, 2010.
9. Luc De Raedt, Tias Guns, and Siegfried Nijssen. Constraint programming for itemset mining. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 204–212, 2008.
10. Luc De Raedt, Tias Guns, and Siegfried Nijssen. Constraint Programming for Data Mining and Machine Learning. In *Proc. of the 24th AAAI Conference on Artificial Intelligence*, 2010.
11. Tias Guns, Thi-Bich-Hanh Dao, Christel Vrain, and Khanh-Chuong Duong. Repetitive branch-and-bound using constraint programming for constrained minimum sum-of-squares clustering. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 462–470, 2016.
12. Tias Guns, Siegfried Nijssen, and Luc De Raedt. Itemset mining: A constraint programming perspective. *Artificial Intelligence*, 175:1951–1983, 2011.
13. Tias Guns, Siegfried Nijssen, and Luc De Raedt. k-Pattern set mining under constraints. *IEEE Transactions on Knowledge and Data Engineering*, 25(2):402–418, 2013.
14. Marianne Mueller and Stefan Kramer. Integer Linear Programming Models for Constrained Clustering. In *Proceedings of the 13th International Conference on Discovery Science*, pages 159–173, 2010.
15. Jean-Philippe Métivier, Patrice Boizumault, Bruno Cremilleux, Medhi Khiari, and Samir Loudni. A constraint-based language for declarative pattern discovery. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 1112–1119, Dec 2011.
16. Kiri Wagstaff and Claire Cardie. Clustering with instance-level constraints. In *Proceedings of the 17th International Conference on Machine Learning*, pages 1103–1110, 2000.

Axiomatization of General Concept Inclusions from Streams of Interpretations with optional Error Tolerance

Francesco Kriegel

Institute of Theoretical Computer Science,
Technische Universität Dresden, Dresden, Germany
francesco.kriegel@tu-dresden.de

Abstract. We propose applications that utilize the infimum and the supremum of closure operators that are induced by structures occuring in the field of *Description Logics*. More specifically, we consider the closure operators induced by interpretations as well as closure operators induced by TBoxes, and show how we can learn GCIs from streams of interpretations, and how an error-tolerant axiomatization of GCIs from an interpretation guided by a hand-crafted TBox can be achieved.

Keywords: Description Logics · Formal Concept Analysis · Most Specific Consequence · Error Tolerance · General Concept Inclusion · TBox · Interpretation · Model · Stream · Incremental Learning · Automatic Learning

1 Introduction

Description Logics [2, 6–8] are a family of well-founded languages for knowledge representation with a strong logical foundation as well as a widely explored hierarchy of decidability and complexity of common reasoning problems. The several reasoning tasks allow for an automatic deduction of implicit knowledge from given explicitly represented facts and axioms, and many reasoning algorithms have been developed. Description Logics are utilized in many different application domains, and in particular provide the logical underpinning of *Web Ontology Language (OWL)* [16] and its profiles.

An interesting problem is the task of (semi-)automatic generation of terminological axioms, so-called *general concept inclusions (GCIs)*, from given data. For example, in [4, 10] Baader and Distel have generalized the construction of implicational bases from so-called *formal contexts* [12, 15] in the field of *Formal Concept Analysis* [14] to the construction of bases of $\mathcal{EL}^{\bot}$-GCIs from *interpretations* in *Description Logics*. The main difference of the underlying data structures is that interpretations additionally allow the expression of binary relations between objects, which implies a number of technical and theoretical difficulties that have been solved by them. In case of incompleteness of the input data set, a technique of *Attribute Exploration* [11–13] can be utilized to axiomatize implications in a sound and complete manner. This approach furthermore presupposes an expert in the domain of interest that is able to correctly answer all queries posed to her. In [5, 10] Baader and Distel have as well extended this technique from formal contexts to interpretations. A further work in the intersection of *Formal Concept Analysis* and *Description Logics* was developed by Rudolph in [20, 21]. He generalized *Attribute*

Exploration to *Relational Exploration*, a technique that processes an interpretation in a multi-step approach where in each step the role-depth of the involved concept descriptions is increased. In particular, Relational Exploration is a sound and complete deduction calculus for GCIs in the Description Logic $\mathcal{FLE}$. A weakness of the exploration methods is the requirement of an expert that is able to truthfully answer all questions posed to it.

In order theory, *closure operators (clop)* denote mappings in a powerset – or more generally, in a lattice – which are extensive, monotone, and idempotent. Many types of data sets give rise to an induced closure operator in such a way that an implication is valid in the data set if, and only if, it is valid in the closure operator. For example, each formal context (G, M, I) induces the clop $X \mapsto X^{II}$ on the powerset $\wp(M)$, and each interpretation $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ induces the clop $C \mapsto C^{\mathcal{II}}$ in the lattice of all $\mathcal{EL}^{\bot}$-concept descriptions ordered by subsumption $\sqsubseteq$ and factorized by equivalence $\equiv$. In a recent paper [19], the author investigated how implicational bases for closure operators can be computed in a parallel manner. Furthermore, it was shown that the set of all clops in a complete lattice constitutes a complete lattice itself, and formulae for computing infima and suprema of clops were provided. In this document, we will introduce a closure operator $C \mapsto C^{\mathcal{T}}$ induced by a TBox $\mathcal{T}$, and will furthermore provide applications for computing bases of GCIs for the infimum as well as the supremum of $C \mapsto C^{\mathcal{T}}$ and $C \mapsto C^{\mathcal{II}}$.

This document is structured as follows. Section 2 gives a brief overview on the easy description logic $\mathcal{EL}^{\bot}$. Then, Section 3 cites some related work, and recalls important notions. Section 4 introduces the notion of a most specific consequence with respect to a TBox, and shows how to define a closure operator induced by a TBox. Section 5 then discusses applications of infima and suprema of clops induced by interpretations and TBoxes, and Section 7 draws some conclusions. Note that proofs are not included, but the interested reader can find them in the corresponding technical report [17].

2 The Description Logic $\mathcal{EL}^{\bot}$

The syntax and semantics of the light-weight description logic $\mathcal{EL}^{\bot}$ are introduced as follows. Throughout the whole document assume that (N_C, N_R) is a signature, i.e., N_C is a set of *concept names*, and N_R is a set of *role names*. An $\mathcal{EL}^{\bot}$*-concept description* is a term that is constructed by means of the following inductive rule:

$$C ::= \bot \mid \top \mid A \mid C \sqcap C \mid \exists r. C.$$

A *general concept inclusion* (abbr. *GCI*) is an expression $C \sqsubseteq D$ where both the *premise* C as well as the *conclusion* D are $\mathcal{EL}^{\bot}$-concept descriptions. A *TBox* is a set of GCIs.

The *role depth* $\mathsf{rd}(C)$ of an $\mathcal{EL}^{\bot}$-concept description C is inductively defined as follows:

$$\mathsf{rd}(\bot) := \mathsf{rd}(\top) := \mathsf{rd}(A) := 0, \quad \mathsf{rd}(C \sqcap D) := \mathsf{rd}(C) \vee \mathsf{rd}(D), \\ \text{and} \quad \mathsf{rd}(\exists r. C) := 1 + \mathsf{rd}(C).$$

An *interpretation* $\mathcal{I} := (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty set $\Delta^{\mathcal{I}}$, called the *domain*, and an *extension function* $\cdot^{\mathcal{I}}$ that maps concept names $A \in N_C$ to subsets $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and maps role names $r \in N_R$ to binary relations $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. Then, the extension function is canonically extended to all $\mathcal{EL}^{\bot}$-concept descriptions by the following definitions:

$$\bot^{\mathcal{I}} := \emptyset, \quad \top^{\mathcal{I}} := \Delta^{\mathcal{I}}, \quad (C \sqcap D)^{\mathcal{I}} := C^{\mathcal{I}} \cap D^{\mathcal{I}}, \\ \text{and} \quad (\exists r. C)^{\mathcal{I}} := \{\, d \in \Delta^{\mathcal{I}} \mid \exists e \in \Delta^{\mathcal{I}} \colon (d, e) \in r^{\mathcal{I}} \text{ and } e \in C^{\mathcal{I}} \,\}.$$

A GCI $C \sqsubseteq D$ is *valid* in $\mathcal{I}$ if $C^\mathcal{I} \subseteq D^\mathcal{I}$. We then also refer to $\mathcal{I}$ as a *model* of $C \sqsubseteq D$, and denote this by $\mathcal{I} \models C \sqsubseteq D$. Furthermore, $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$, symbolized as $\mathcal{I} \models \mathcal{T}$, if each GCI in $\mathcal{T}$ is valid in $\mathcal{I}$. The entailment relation is lifted to TBoxes as follows: A GCI $C \sqsubseteq D$ is *entailed* by a TBox $\mathcal{T}$, denoted as $\mathcal{T} \models C \sqsubseteq D$, if each model of $\mathcal{T}$ is a model of $C \sqsubseteq D$, too. We then also say that C is *subsumed* by D with respect to $\mathcal{T}$. A TBox $\mathcal{T}$ *entails* a TBox $\mathcal{U}$, symbolized as $\mathcal{T} \models \mathcal{U}$, if $\mathcal{T}$ entails each GCI in $\mathcal{U}$, or equivalently if each model of $\mathcal{T}$ is also a model of $\mathcal{U}$. Two $\mathcal{EL}^\bot$-concept descriptions C and D are *equivalent* with respect to $\mathcal{T}$, and we shall write $\mathcal{T} \models C \equiv D$, if $\mathcal{T} \models \{C \sqsubseteq D, D \sqsubseteq C\}$. In case $\mathcal{T} = \emptyset$ we may ommit the prefix "$\emptyset \models$". However, then we have to carefully interpret an expression $C \sqsubseteq D$ – it either just denotes a general concept inclusion, i.e., an axiom, without stating where it is valid; or it expresses that C is subsumed by D (w.r.t. $\emptyset$), i.e., $C^\mathcal{I} \subseteq D^\mathcal{I}$ is satisfied in all interpretations $\mathcal{I}$. An analogous hint applies to concept equivalences $C \equiv D$.

It is readily verified that the *subsumption* $\sqsubseteq$ constitutes a quasi-order on the set $\mathcal{EL}^\bot(N_C, N_R)$ of all $\mathcal{EL}^\bot$-concept descriptions over the signature (N_C, N_R). Hence, the quotient of $\mathcal{EL}^\bot(N_C, N_R)$ with respect to the induced *equivalence* $\equiv$ is a partially ordered set (a *poset*). (In the following we will not distinguish between the equivalence classes and their representatives.) The poset is even a bounded lattice. Of course, $\bot$ is the smallest element, and $\top$ is the greatest element. Furthermore, the conjunction $\sqcap$ corresponds to the *infimum* operation, and the least common subsumer mapping $\vee$ corresponds to the *supremum* operation. Remark that the *least common subsumer* (abbr. *lcs*) $C \vee D$ of two $\mathcal{EL}^\bot$-concept descriptions C and D is (up to equivalence) uniquely defined by the following conditions: 1. $C \sqsubseteq C \vee D$ as well as $D \sqsubseteq C \vee D$, and 2. for each $\mathcal{EL}^\bot$-concept description E, if $C \sqsubseteq E$ and $D \sqsubseteq E$, then $C \vee D \sqsubseteq E$. It is easy to see that the equivalence $\equiv$ is compatible with both $\sqcap$ and $\vee$. In the sequel of this document, we shall denote this bounded lattice by $\boldsymbol{\mathcal{EL}}^\bot(N_C, N_R) \coloneqq {}^{(\mathcal{EL}^\bot(N_C, N_R), \sqsubseteq)}\!/_{\equiv}$. For a role-depth bound $\delta \in \mathbb{N}$, $\mathcal{EL}^\bot(N_C, N_R)\restriction_\delta$ is the set of all $\mathcal{EL}^\bot$-concept descriptions with a role depth of at most δ, and accordingly $\boldsymbol{\mathcal{EL}}^\bot(N_C, N_R)\restriction_\delta \coloneqq {}^{(\mathcal{EL}^\bot(N_C, N_R)\restriction_\delta, \sqsubseteq)}\!/_{\equiv}$ symbolizes the corresponding bounded lattice of (equivalence classes of) $\mathcal{EL}^\bot$-concept descriptions. Note that $\boldsymbol{\mathcal{EL}}^\bot(N_C, N_R)\restriction_\delta$ is complete if the underlying signature (N_C, N_R) is finite.

3 Related Work

Baader and Distel introduced a technique for the axiomatization of general concept inclusions that are valid in a given interpretation. More specifically, they interconnected Formal Concept Analysis with Description Logics by defining so-called *induced* formal contexts such that its canonical base can directly be converted into a base of GCIs for the underlying interpretation. However, there was no possibility to include existing knowledge. For the case where a set of GCIs valid in the given interpretation is available, a solution for the computation of a *relative* base of GCIs has been proposed in [18]. However, it was not clear how to proceed in presence of an interpretation $\mathcal{I}$ and a TBox $\mathcal{T}$ where $\mathcal{I} \not\models \mathcal{T}$. This problem will be tackled in the following section.

Beforehand, we recall the notion of a model-based most specific concept description. Let $\mathcal{I}$ be an interpretation, and consider a set $X \subseteq \Delta^\mathcal{I}$ as well as a role-depth bound $\delta \in \mathbb{N}$. Then an $\mathcal{EL}^\bot$-concept description C is called a *role-depth-bounded model-based most specific concept description* (abbr. *mmsc*) of X with respect to $\mathcal{I}$ and δ if the following statements hold: 1. $\mathsf{rd}(C) \leq \delta$, 2. $X \subseteq C^\mathcal{I}$, and 3. for all concept descriptions

D, if $X \subseteq D^{\mathcal{I}}$, then $\emptyset \models C \sqsubseteq D$. As an immediate consequence of the definition we infer that the mmsc of X w.r.t. $\mathcal{I}$ and δ is unique up to equivalence. Hence, we may speak of *the* mmsc, and denote it by $X^{\mathcal{I}_\delta}$.

4 Most Specific Consequences with respect to a TBox

For a given $\mathcal{EL}^\bot$-TBox $\mathcal{T}$, and an $\mathcal{EL}^\bot$-concept description C, one may ask for a concept description D which is *most specific* with respect to the condition that C is subsumed by D w.r.t. $\mathcal{T}$. Such a concept description is called *most specific consequence*, or *most specific subsumer* (abbr. *mss*). Note that Distel has investigated a dual notion in [10, Chapter 7], namely that of a *minimal possible consequence*, which he utilized to constitute an algorithm for the exploration of ontologies, called *ABox Exploration*. To emphasize this duality, it is reasonable to use the name of a *minimal certain consequence*.

In this section, the notion of a most specific consequence shall be formally introduced, and necessary as well as sufficient conditions for its existence will be explored. Furthermore, we investigate its relationship to entailment with respect to a TBox. The next section then provides at least one application that utilizes most specific consequences to construct a base of GCIs that are both valid in a given interpretation as well as are entailed by a given TBox.

As an exemplary TBox, consider $\mathcal{T} \coloneqq \{A \sqsubseteq \exists r.\, A\}$. It can be readily verified that for each $n \in \mathbb{N}$, the concept description $(\exists r.)^n A$ is a *consequence* (i.e., a subsumer) of A with respect to $\mathcal{T}$. However, $(\exists r.)^{n+1} A$ is more specific than $(\exists r.)^n A$, and thus a most specific consequence of A w.r.t. $\mathcal{T}$ does not exist in the description logic $\mathcal{EL}^\bot$ with *descriptive semantics* (as introduced in Section 2). There are two solutions to tackle this problem of existence of most specific consequences. The first one is to use the extension of $\mathcal{EL}^\bot$ with *greatest fixpoint semantics*. This extension has been extensively studied in [1, 3, 10], and in particular it has been shown that this extension can handle terminological cycles (as present in the given TBox $\mathcal{T}$ above). In particular, $\mathcal{EL}^\bot_{\mathsf{gfp}}$-concept descriptions are pairs $C \coloneqq (A_C, \mathcal{T}_C)$ where $\mathcal{T}_C$ is a TBox, and A_C is a defined concept name of $\mathcal{T}_C$. We do not introduce the full machinery of the semantics of $\mathcal{EL}^\bot_{\mathsf{gfp}}$ here, but rather refer the interested reader to [1, 3]. However, it can be shown that the most specific consequence of the example above indeed exists in $\mathcal{EL}^\bot_{\mathsf{gfp}}$, and is given by $(A, \mathcal{T})$. It is straight-forward to claim that most specific consequences always exist in $\mathcal{EL}^\bot_{\mathsf{gfp}}$, but nevertheless a corresponding proof is outstanding.

Another solution for ensuring the existence of most specific consequences is to *restrict the role-depth* of the concept descriptions under consideration, as this has been done in [9] to ensure the existence of model-based most specific concept descriptions in $\mathcal{EL}^\bot$ with descriptive semantics. We introduce the following definition.

Definition 1. *Let $\mathcal{T}$ be an $\mathcal{EL}^\bot$-TBox, C an $\mathcal{EL}^\bot$-concept description, and $\delta \in \mathbb{N}$ a* role-depth bound. *Then an $\mathcal{EL}^\bot$-concept description D is called a* role-depth-bounded most specific consequence *of C with respect to $\mathcal{T}$ and δ if it satisfies the following conditions: 1. $\mathsf{rd}(D) \leq \delta$, 2. $\mathcal{T} \models C \sqsubseteq D$, and 3. for each $\mathcal{EL}^\bot$-concept description E, if $\mathsf{rd}(E) \leq \delta$ and $\mathcal{T} \models C \sqsubseteq E$, then $\emptyset \models D \sqsubseteq E$.*

Provided that such role-depth-bounded most specific consequences exist, they are unique up to equivalence, and hence we may then speak of *the* most specific consequence, and we shall denote it by $C^{\mathcal{T}_\delta}$ for a given concept description C, a TBox $\mathcal{T}$, and a

role-depth bound δ. By Definition 1, $\mathcal{T} \models C \sqsubseteq C^{\mathcal{T}_\delta}$. Furthermore, from $\mathcal{T} \models C \sqsubseteq C$ we conclude that $\emptyset \models C^{\mathcal{T}_\delta} \sqsubseteq C$. In summary, $\mathcal{T} \models C \equiv C^{\mathcal{T}_\delta}$.

Lemma 2. *Role-depth-bounded most specific consequences always exist, provided that the signature is finite.*

Lemma 3. *Let $\mathcal{T} \cup \{C \sqsubseteq D\}$ be an $\mathcal{EL}^\bot$-TBox such that D has a role depth of at most δ. Then the following statements are equivalent:*

1. $\mathcal{T} \models C \sqsubseteq D$.
2. $\emptyset \models C^{\mathcal{T}_\delta} \sqsubseteq D$.
3. $\{ E \sqsubseteq E^{\mathcal{T}_\delta} \mid E \in \mathcal{EL}^\bot(N_C, N_R)\restriction_\delta \} \models C \sqsubseteq D$.

If all conclusions of GCIs in $\mathcal{T}$ have role depths not exceeding δ, then furthermore the following statement is equivalent to the previous ones:

4. $\{ E \sqsubseteq E^{\mathcal{T}_\delta} \mid \exists F \colon E \sqsubseteq F \in \mathcal{T} \} \models C \sqsubseteq D$.

Corollary 4. *Let $\mathcal{T} \cup \{C \sqsubseteq D\}$ be an $\mathcal{EL}^\bot$-TBox such that $\mathcal{T} \models C \sqsubseteq D$, and both C and D have role depths not exceeding δ. Then for each $\mathcal{EL}^\bot$-concept description E, if $\emptyset \models E^{\mathcal{T}_\delta} \sqsubseteq C$, then $\emptyset \models E^{\mathcal{T}_\delta} \sqsubseteq D$.*

This document does not include a method for the computation of most specific consequences, and leaves this problem open for future research. However, we have shown that their existence is guaranteed in the case of descriptive semantics when the candidate concept descriptions are restricted in their role depth. In particular, the notion defined in Definition 1 always exists. As a next step, we provide a technique that allows for the computation of a TBox from a stream of interpretations and that utilizes those most specific consequences.

Lemma 5. *The mapping $C \mapsto C^{\mathcal{T}_\delta}$ is a closure operator in the dual of $\boldsymbol{\mathcal{EL}}^\bot(N_C, N_R)\restriction_\delta$.*

5 Axiomatization of General Concept Inclusions from Streams of Interpretations

Consider a setting where a stream of interpretations $\mathcal{I}_n$, $n \in \mathbb{N}$, can be observed, and furthermore for each time point $n \in \mathbb{N}$, a TBox $\mathcal{T}_n$ shall be constructed such that for each GCI $C \sqsubseteq D$, $\mathcal{T} \models C \sqsubseteq D$ if, and only if, $\mathcal{I}_k \models C \sqsubseteq D$ for all previous time points $k \leq n$. Of course, for the initial moment $n = 0$, we may simply compute $\mathcal{T}_0$ as a base of GCIs for $\mathcal{I}_0$, utilizing the approach of Baader and Distel [4, 10]. For the following moments $n \geq 1$, we may of course also construct a base of GCIs for the disjoint union of the interpretations $\mathcal{I}_0, \mathcal{I}_1, \ldots, \mathcal{I}_n$. However, since the method requires the construction of so-called *induced contexts* the size of which may be exponential in the size of the domain of the interpretation, this technique could possibly be infeasible for late time points. Furthermore, it requires the storing of all interpretations observed so far. We want to present another technique for solving the above mentioned task. Please note that this problem was already addressed in [18] for the case of $\mathcal{I}_{n+1} \models \mathcal{T}_n$. Here, we propose a solution that circumvents this rather restrictive precondition.

The infimum of $\cdot^{\mathcal{T}}$ and $\cdot^{\mathcal{II}}$ is the greatest closure operator below both $\cdot^{\mathcal{T}}$ and $\cdot^{\mathcal{II}}$. The following lemma recalls an important property of infima of clops, specifically tailored to the case of implications of concept descriptions, i.e., of general concept inclusions as they are more commonly called.

Lemma 6. *Let $\mathcal{I}$ be an interpretation, $\mathcal{T}$ an $\mathcal{EL}^{\bot}$-TBox, and $C \sqsubseteq D$ a general concept inclusion such that both premise and conclusion have a role-depth not exceeding δ. Then the following statements are equivalent:*

1. *$C \sqsubseteq D$ is both valid in $\mathcal{I}$ as well as entailed by $\mathcal{T}$.*
2. *$\emptyset \models \{C^{\mathcal{I}\mathcal{I}_\delta} \sqsubseteq D, C^{\mathcal{T}_\delta} \sqsubseteq D\}$.*
3. *$\emptyset \models C^{\mathcal{I}\mathcal{I}_\delta} \vee C^{\mathcal{T}_\delta} \sqsubseteq D$.*
4. *$C \sqsubseteq D$ is valid in the infimum of $C \mapsto C^{\mathcal{I}\mathcal{I}_\delta}$ and $C \mapsto C^{\mathcal{T}_\delta}$.*

As a conclusion, we infer the following incremental method for the computation of a sequence of TBoxes from a sequence of interpretations:

1. Upon availability of the first observed interpretation $\mathcal{I}_0$, compute its canonical base of GCIs, as proposed by Baader and Distel in [4, 10]. If only concept descriptions up to a certain role depth shall be considered, then the variant described by Borchmann, Distel, et al., in [9] is sufficient.
2. For each new interpretation $\mathcal{I}_{n+1}$, compute the canonical base of the infimum of the clops that are induced by the current TBox $\mathcal{T}_n$ as well as by the newly observed interpretation $\mathcal{I}_{n+1}$.

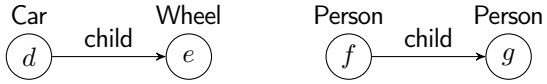
It is readily verified that – by construction – for each time point $n \in \mathbb{N}$, the TBox $\mathcal{T}_n$ entails a GCI $C \sqsubseteq D$ if, and only if, $C \sqsubseteq D$ is valid in the interpretations $\mathcal{I}_0, \ldots, \mathcal{I}_n$.

6 Error-Tolerant Axiomatization of General Concept Inclusions from Interpretations

Assume that we were given an interpretation $\mathcal{I}$ as well as a TBox $\mathcal{T}$ such that $\mathcal{I}$ contains observations that may be possibly faulty due to inaccurate generation methods, and that $\mathcal{T}$ is certainly valid in the domain of interest, e.g., as it has been hand-crafted by experts. In particular, we assume that $\mathcal{I}$ is not a model of $\mathcal{T}$, i.e., that at least one domain element in $\mathcal{I}$ exists which serves as a counterexample against at least one GCI from $\mathcal{T}$. However, we are expected to axiomatize terminological knowledge from $\mathcal{I}$, which is valid in the unknown domain of interest. As a solution, we will construct the implicational base of the supremum of the clops that are induced by $\mathcal{I}$, and by $\mathcal{T}$, respectively. It is then ensured that those implications are considered which are valid for all those domain elements of $\mathcal{I}$ that respect the GCIs in $\mathcal{T}$, i.e., that we axiomatize general concept inclusions from $\mathcal{I}$ that are compatible with the axioms contained in $\mathcal{T}$. In a certain sense this yields a method for an error correction in $\mathcal{I}$ when learning GCIs. We will describe a short motivating example. Define a TBox $\mathcal{T}$ and an interpretation $\mathcal{I}$ as follows:

$$N_C := \{\mathsf{Person}, \mathsf{Car}, \mathsf{Wheel}\}, \quad N_R := \{\mathsf{child}\}$$
$$\mathcal{T} := \{\exists\, \mathsf{child}.\, \top \sqsubseteq \mathsf{Person}, \mathsf{Person} \sqcap \mathsf{Car} \sqsubseteq \bot\}$$

$\mathcal{I}$:



Consider the GCI $\mathsf{Car} \sqsubseteq \exists\, \mathsf{child}.\, \mathsf{Wheel}$. Of course, it is valid in $\mathcal{I}$, and furthermore is contained in the canonical base of $\mathcal{I}$ when applying the construction from [4, 10].

We can show that the above mentioned GCI is also valid $\mathcal{II}_\delta \curlyvee \mathcal{T}_\delta$ for $\delta \geq 1$:

$$\mathsf{Car}^{\mathcal{I}\mathcal{I}_\delta} \equiv \mathsf{Car} \sqcap \exists\, \mathsf{child}.\, \mathsf{Wheel}$$
$$(\mathsf{Car} \sqcap \exists\, \mathsf{child}.\, \mathsf{Wheel})^{\mathcal{T}_\delta} \equiv \mathsf{Car} \sqcap \exists\, \mathsf{child}.\, \mathsf{Wheel} \sqcap \mathsf{Person} \sqcap \bot \equiv \bot,$$

The closure of Car with respect to the supremum is the least common subsumer of those concept descriptions that are closures of both $C \mapsto C^{\mathcal{I}\mathcal{I}_\delta}$ and $C \mapsto C^{\mathcal{T}_\delta}$, as well as are subsumed by Car. It is easy to see that this supremum closure can be computed by an exhaustive repeated application of both closure operators until a fixpoint is reached. As we have seen above, the fixpoint $\bot$ is reached after the first iteration, and hence $\bot$ is the supremum-closure.

However, the GCI is a consequence of the more specific valid GCI $\mathsf{Car} \sqsubseteq \bot$, and hence would not have been axiomatized upon construction of the canonical base. In particular, we see that d is not compatible with $\mathcal{T}$ – in contrast to the other domain elements e, f, and g. Eventually, Car is a pseudo-closure of the supremum, and hence the canonical base contains the axiom expressing the non-existence of cars.

7 Conclusion

We have defined the notion of most specific consequences with respect to TBoxes, and considered the corresponding closure operator. We have investigated the interplay of this closure operator induced by a given TBox with the closure operator induced by an interpretation – more specifically, we have shown how their infimum can be utilized for learning from streams of interpretations, and have motivated how their supremum can be used for an error-tolerant axiomatization of general concept inclusions from interpretations in the presence of a hand-crafted TBox that indicates errors in the observed interpretation.

Acknowledgements. The author gratefully thanks the anonymous reviewers for their constructive hints and helpful remarks.

References

[1] Franz Baader. "Least Common Subsumers and Most Specific Concepts in a Description Logic with Existential Restrictions and Terminological Cycles". In: *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*. Ed. by Georg Gottlob and Toby Walsh. Morgan Kaufmann, 2003, pp. 319–324.

[2] Franz Baader. "Logic-Based Knowledge Representation". In: *Artificial Intelligence Today*. 1999, pp. 13–41.

[3] Franz Baader. "Terminological Cycles in a Description Logic with Existential Restrictions". In: *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*. Ed. by Georg Gottlob and Toby Walsh. Morgan Kaufmann, 2003, pp. 325–330.

[4] Franz Baader and Felix Distel. "A Finite Basis for the Set of EL-Implications Holding in a Finite Model". In: *Formal Concept Analysis, 6th International Conference, ICFCA 2008, Montreal, Canada, February 25-28, 2008, Proceedings*. Ed. by Raoul Medina and Sergei A. Obiedkov. Vol. 4933. Lecture Notes in Computer Science. Springer, 2008, pp. 46–61.

[5] Franz Baader and Felix Distel. "Exploring Finite Models in the Description Logic". In: *Formal Concept Analysis, 7th International Conference, ICFCA 2009, Darmstadt, Germany, May 21-24, 2009, Proceedings*. Ed. by Sébastien Ferré and Sebastian Rudolph. Vol. 5548. Lecture Notes in Computer Science. Springer, 2009, pp. 146–161.

[6] Franz Baader, Ian Horrocks, and Ulrike Sattler. "Description Logics". In: *Handbook on Ontologies*. Ed. by Steffen Staab and Rudi Studer. International Handbooks on Information Systems. Springer, 2004, pp. 3–28.

[7] Franz Baader, Ian Horrocks, and Ulrike Sattler. "Description Logics". In: *Handbook on Ontologies*. Ed. by Steffen Staab and Rudi Studer. International Handbooks on Information Systems. Springer, 2009, pp. 21–43.

[8] Franz Baader and Carsten Lutz. "Description Logic". In: *Handbook of Modal Logic*. Ed. by Patrick Blackburn, Johan van Benthem, and Frank Wolter. Elsevier, 2006. Chap. 13.

[9] Daniel Borchmann, Felix Distel, and Francesco Kriegel. "Axiomatisation of General Concept Inclusions from Finite Interpretations". In: *Journal of Applied Non-Classical Logics* 26.1 (2016), pp. 1–46.

[10] Felix Distel. "Learning description logic knowledge bases from data using methods from formal concept analysis". PhD thesis. Dresden, Germany: Technische Universität Dresden, 2011.

[11] Bernhard Ganter. "Attribute Exploration with Background Knowledge". In: *Theor. Comput. Sci.* 217.2 (1999), pp. 215–233.

[12] Bernhard Ganter. *Two Basic Algorithms in Concept Analysis*. FB4-Preprint 831. Darmstadt, Germany: Technische Hochschule Darmstadt, 1984.

[13] Bernhard Ganter and Sergei A. Obiedkov. *Conceptual Exploration*. Springer, 2016.

[14] Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. 1st ed. Springer, Dec. 1999.

[15] Jean-Luc Guigues and Vincent Duquenne. "Famille minimale d'implications informatives résultant d'un tableau de données binaires". In: *Mathématiques et Sciences Humaines* 95 (1986), pp. 5–18.

[16] Pascal Hitzler, Markus Krötzsch, and Sebastian Rudolph. *Foundations of Semantic Web Technologies*. Chapman and Hall/CRC Press, 2010.

[17] Francesco Kriegel. *Axiomatization of General Concept Inclusions from Streams of Interpretations with optional Error Tolerance*. LTCS-Report 16-05. Dresden, Germany: Chair for Automata Theory, Institute for Theoretical Computer Science, Technische Universität Dresden, 2016.

[18] Francesco Kriegel. "Incremental Learning of TBoxes from Interpretation Sequences with Methods of Formal Concept Analysis". In: *Proceedings of the 28th International Workshop on Description Logics, Athens,Greece, June 7-10, 2015*. Ed. by Diego Calvanese and Boris Konev. Vol. 1350. CEUR Workshop Proceedings. CEUR-WS.org, 2015.

[19] Francesco Kriegel. "NextClosures with Constraints". In: *Proceedings of the Thirteenth International Conference on Concept Lattices and Their Applications, Moscow, Russia, July 18-22, 2016*. Ed. by Marianne Huchard and Sergei Kuznetsov. Vol. 1624. CEUR Workshop Proceedings. CEUR-WS.org, 2016, pp. 231–243.

[20] Sebastian Rudolph. "Exploring Relational Structures Via FLE". In: *Conceptual Structures at Work: 12th International Conference on Conceptual Structures, ICCS 2004, Huntsville, AL, USA, July 19-23, 2004. Proceedings*. Ed. by Karl Erich Wolff, Heather D. Pfeiffer, and Harry S. Delugach. Vol. 3127. Lecture Notes in Computer Science. Springer, 2004, pp. 196–212.

[21] Sebastian Rudolph. "Relational exploration: combining description logics and formal concept analysis for knowledge specification". PhD thesis. Dresden University of Technology, 2006.

Towards a sequent calculus for formal contexts

Ondrej Krídlo[1] and Manuel Ojeda-Aciego[2]

[1] University of Pavol Jozef Šafárik, Košice, Slovakia*
[2] Universidad de Málaga. Departamento de Matemática Aplicada. Spain**

Abstract. This work focuses on the definition of a consequence relation between contexts with which we can decide whether certain contextual information is a logical consequence from a set of contexts considered as underlying hypotheses.

1 Introduction

In real life, one often faces situations in which the underlying knowledge is given as a set of tables which can be interpreted as formal contexts, and we should decide on whether certain contextual information is a consequence from them.

This problem clearly resembles the notion of a formula being a logical consequence of a set of hypotheses, and suggests the possibility or convenience of defining a formal (mathematical) notion of logical consequence between contexts.

The only attempts to introduce logical content within the machinery of Formal Concept Analysis (FCA), apart from its ancient roots anchored in the Port-Royal logic, are the so-called logical information systems and the logical concept analysis [2, 9].

Of course, different links between FCA and logic have been studied but, to the best of our knowledge, the problem considered in this paper has not been explicitly studied in the literature. Nevertheless, it is worth to remark that the concluding section of [6] states that dual bonds could be given a proof-theoretical interpretation in terms of consequence relations.

In the present work, we consider the fact that the category ChuCors of contexts and Chu correspondences is *-autonomous, and hence a model of linear logic, in order to build some preliminary links with the conjunctive fragment of this logic.

* Partially supported by grants VEGA 1/0073/15 and APVV-15-0091
** Partially supported by Spanish Ministry of Science project TIN2015-70266-C2-1-P, co-funded by the European Regional Development Fund (ERDF)

2 Preliminaries

In order to make the manuscript self-contained, the fundamental notions and their main properties are recalled in this section.

2.1 Context, concept and concept lattice

Definition 1. *A* formal context *is any triple $\mathcal{C} = \langle B, A, R\rangle$ where B and A are finite sets and $R \subseteq B \times A$ is a binary relation. It is customary to say that B is a set of* objects*, A is a set of* attributes *and R represents a relation between objects and attributes.*

Given a formal context $\langle B, A, R\rangle$, the derivation (or concept-forming) operators are a pair of mappings $\uparrow\colon 2^B \to 2^A$ and $\downarrow\colon 2^A \to 2^B$ such that if $X \subseteq B$, then $\uparrow X$ is the set of all attributes which are related to every object in X and, similarly, if $Y \subseteq A$, then $\downarrow Y$ is the set of all objects which are related to every attribute in Y.

Definition 2. *A* formal concept *of a formal context $\mathcal{C} = \langle B, A, R\rangle$ is a pair of sets $\langle X, Y\rangle \in 2^B \times 2^A$ which is a fixpoint of the pair of concept-forming operators, namely, $\uparrow X = Y$ and $\downarrow Y = X$. The object part X is called the* extent *and the attribute part Y is called the* intent*. The set of all formal concepts of a context $\mathcal{C}$ will be denoted by* $\mathrm{CL}(\mathcal{C})$*, set of all extents or intents of $\mathcal{C}$ will be denoted by* $\mathrm{Ext}(\mathcal{C})$ *or* $\mathrm{Int}(\mathcal{C})$ *respectively.*

2.2 Intercontextual structures

Two main constructions have been traditionally considered in order to relate two formal contexts: the bonds and the Chu correspondences.

Definition 3. *Let $\mathcal{C}_1 = \langle B_1, A_1, R_1\rangle$ and $\mathcal{C}_2 = \langle B_2, A_2, R_2\rangle$ be two formal contexts. A* bond *between $\mathcal{C}_1$ and $\mathcal{C}_2$ is any relation $\beta \in 2^{B_1 \times A_2}$ such that, when interpreted as a table, its columns are extents of $\mathcal{C}_1$ and its rows are intents of $\mathcal{C}_2$. All bonds between such contexts will be denoted by* $\mathrm{Bonds}(\mathcal{C}_1, \mathcal{C}_2)$.

Another equivalent definition of bond between $\mathcal{C}_1$ and $\mathcal{C}_2$ defines it as any relation $\beta \in 2^{B_1 \times A_2}$ such that $\mathrm{Ext}(\langle B_1, A_2, \beta\rangle) \subseteq \mathrm{Ext}(\mathcal{C}_1)$ *and* $\mathrm{Int}(\langle B_1, A_2, \beta\rangle) \subseteq \mathrm{Int}(\mathcal{C}_2)$

Dual bonds *between $\mathcal{C}_1$ and $\mathcal{C}_2$ are bonds between $\mathcal{C}_1$ and transposition of $\mathcal{C}_2$. Transposition of any context $\mathcal{C} = \langle B, A, R\rangle$ is defined as a new context $\mathcal{C}^* = \langle A, B, R^t\rangle$ with $R^t(a, b)$ holds iff $R(b, a)$ holds.*

The notion of Chu correspondence between contexts can be seen as an alternative inter-contextual structure which, instead, links intents of $\mathcal{C}_1$ and extents of $\mathcal{C}_2$.

Definition 4. *Consider* $\mathcal{C}_1 = \langle B_1, A_1, R_1\rangle$ *and* $\mathcal{C}_2 = \langle B_2, A_2, R_2\rangle$ *two formal contexts. A* Chu correspondence *between* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *is any pair* $\varphi = \langle \varphi_L, \varphi_R\rangle$ *of mappings* $\varphi_L\colon B_1 \to \mathrm{Ext}(\mathcal{C}_2)$ *and* $\varphi_R\colon A_2 \to \mathrm{Int}(\mathcal{C}_1)$ *such that for all* $(b_1, a_2) \in B_1 \times A_2$ *it holds that* $\uparrow_2\big(\varphi_L(b_1)\big)(a_2) = \downarrow_1 \big(\varphi_R(a_2)\big)(b_1)$.

All Chu correspondences between such contexts will be denoted by $\mathrm{Chu}(\mathcal{C}_1, \mathcal{C}_2)$.

The notions of bond and Chu correspondence are interchangeable; specifically, we can consider the bond β_φ associated to a Chu correspondence φ from $\mathcal{C}_1$ to $\mathcal{C}_2$ defined for $b_1 \in B_1, a_2 \in A_2$ as follows

$$\beta_\varphi(b_1, a_2) = \uparrow_2\big(\varphi_L(b_1)\big)(a_2) = \downarrow_1\big(\varphi_R(a_2)\big)(b_1)$$

Similarly, we can consider the Chu correspondence φ_β associated to a bond ρ defined by the following pair of mappings:

$$\varphi_{\beta L}(b_1) = \downarrow_2\big(\beta(b_1)\big)\ \ \varphi_{\beta R}(a_2) = \uparrow_1\big(\beta^t(a_2)\big) \text{ for all } a_2 \in A_2 \text{ and } o_1 \in B_1$$

The set of all bonds (resp. Chu correspondences) between two formal contexts endowed with the ordering given by set inclusion is a complete lattice. Moreover, both complete lattices are dually isomorphic.

2.3 Categorical products in ChuCors

Recall that it is possible to consider a category in which the objects are formal contexts and morphisms between two contexts are the Chu correspondences between them. This category, denoted ChuCors, has been proved to be *-autonomous and equivalent to the category of complete lattices and isotone Galois connections, more results on this category and its L-fuzzy extensions can be found in [4, 3, 5, 7].

Cartesian product in ChuCors The following definition provides a specific construction of the notion of (binary) cartesian product in the category ChuCors.

Definition 5. *Consider* $\mathcal{C}_1 = \langle B_1, A_1, R_1\rangle$ *and* $\mathcal{C}_2 = \langle B_2, A_2, R_2\rangle$ *two formal contexts. The* product *of such contexts is a new formal context* $\mathcal{C}_1 \times \mathcal{C}_2 = \langle B_1 \uplus B_2, A_1 \uplus A_2, R_{1\times 2}\rangle$ *where the relation* $R_{1\times 2}$ *is given by*

$$((i, b), (j, a)) \in R_{1\times 2} \text{ if and only if } \big((i = j) \Rightarrow (b, a) \in R_i\big)$$

for any $(b,a) \in B_i \times A_j$ *and* $(i,j) \in \{1,2\} \times \{1,2\}$.

If we recall the well-known categorical theorem which states that if a category has a terminal object and binary product, then it has all finite products, we need to prove just the existence of a terminal object (namely, the nullary product) in order to prove the category ChuCors to be Cartesian.

Any formal context of the form $\langle B, A, B \times A\rangle$ where the incidence relation is the full Cartesian product of the sets of objects and attributes is (isomorphic to) the terminal object of ChuCors. Such a formal context has just one formal concept $\langle B, A\rangle$; hence, from any other formal context there is just one Chu correspondence to $\langle B, A, B \times A\rangle$.

The explicit construction of a general product (not necessarily either binary or nullary) is given below:

Definition 6. *Let* $\{\mathcal{C}_i\}_{i\in I}$ *be an indexed family of formal contexts* $\mathcal{C}_i = \langle B_i, A_i, R_i\rangle$*, the product* $\prod_{i\in I}\mathcal{C}_i$ *is the formal context given by*

$$\prod_{i\in I}\mathcal{C}_i = \left\langle \biguplus_{i\in I} B_i, \biguplus_{i\in I} A_i, R_{\times I} \right\rangle$$

where $\big((k,b),(m,a)\big) \in R_{\times I} \Leftrightarrow \big((k=m) \Rightarrow (b,a) \in R_k\big)$.

It is worth to note that the arbitrary product of contexts commutes with both the concept lattice construction and the bonds between contexts. These two results are explicitly stated below.

Lemma 1. *Let* $\mathcal{C}_i = \langle B_i, A_i, R_i\rangle$ *be a formal context for* $i \in I$*. It holds that* $\mathrm{CL}(\prod_{i\in I}\mathcal{C}_i)$ *is isomorphic to* $\prod_{i\in I}\mathrm{CL}(\mathcal{C}_i)$.

Lemma 2. *Let* I *and* J *be two index sets, and consider the two sets of formal contexts* $\{\mathcal{C}_i\}_{i\in I}$ *and* $\{\mathcal{D}_j\}_{j\in J}$*. The following isomorphism holds*

$$\mathrm{Bonds}\Big(\prod_{i\in I}\mathcal{C}_i, \prod_{j\in J}\mathcal{D}_j\Big) \cong \prod_{(i,j)\in I\times J} \mathrm{Bonds}(\mathcal{C}_i, \mathcal{D}_j)\,.$$

Tensor product Another product-like construction can be given in the category ChuCors.

Note that if $\varphi \in \mathrm{Chu}(\mathcal{C}_1, \mathcal{C}_2)$, then we can consider $\varphi^* \in \mathrm{Chu}(C_2^*, C_1^*)$ defined by $\varphi_L^* = \varphi_R$ and $\varphi_R^* = \varphi_L$.

Definition 7. *The tensor product* $\mathcal{C}_1 \boxtimes \mathcal{C}_2$ *of contexts* $\mathcal{C}_i = \langle B_i, A_i, R_i \rangle$ *for* $i \in \{1, 2\}$ *is defined as the context* $\langle B_1 \times B_2, \mathrm{Chu}(\mathcal{C}_1, \mathcal{C}_2^*), R_\boxtimes \rangle$ *where*

$$R_\boxtimes\big((b_1, b_2), \varphi\big) = \downarrow_2 \big(\varphi_L(b_1)\big)(b_2).$$

The properties of the tensor product were shown in [7], together with the result that ChuCors with $\boxtimes$ is symmetric and monoidal. Those results were later extended to the L-fuzzy case in [3]. In both papers, the structure of the formal concepts of a tensor product context was established as an ordered pair formed by a bond and a set of Chu correspondences.

Lemma 3. *Let* (β, X) *be a formal concept of the tensor product* $\mathcal{C}_1 \boxtimes \mathcal{C}_2$*, it holds that* $\beta = \bigwedge_{\psi \in X} \beta_\psi$ *and* $X = \{\psi \in \mathrm{Chu}(\mathcal{C}_1, \mathcal{C}_2^*) \mid \beta \leq \beta_\psi\}$.

Due to the monoidal properties of $\boxtimes$ on ChuCors we can add a notion of n-ary tensor product of n formal contexts $\boxtimes_{i=1}^n \mathcal{C}_i$ of any n formal contexts $\mathcal{C}_i$ for $i \in \{1, \ldots, n\}$. Hence, it is possible to consider a notion of n-ary bond that we can imagine as any extent of n-ary tensor product.

Definition 8. *Let* $\mathcal{C}_i = \langle B_i, A_i, R_i \rangle$ *be formal contexts for* $i \in \{1, \ldots, n\}$*. A* dual n-ary bond *between* $\{\mathcal{C}_i\}_{i=1}^n$ *is an* n*-ary relation* $\beta \subseteq \prod_{i=1}^n B_i$ *such that for all* $i \in \{1, 2, \ldots, n\}$ *and any* $(b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_n) \in \prod_{j=1, j\neq i}^n B_i$ *it holds that*

$$\beta(b_1, \ldots, b_{i-1}, (-), b_{i+1}, \ldots, b_n) \in \mathrm{Ext}(\mathcal{C}_i)\,.$$

Lemma 4. *Let* $\{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ *be a set of* n *formal contexts and* β *be some* n*-ary bond between such contexts. Let* $\mathcal{D}_i^\beta$ *be a new formal context defined as* $\langle B_i, \prod_{j=1, j\neq i}^n B_j, \mathcal{R}_i \rangle$ *where* $\mathcal{R}_i(b_i, (b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_n)) = \beta(b_1, \ldots, b_n)$ *for any* $i \in \{1, 2, \ldots, n\}$*. Then* $\mathrm{Ext}(\mathcal{D}_i^\beta) \subseteq \mathrm{Ext}(\mathcal{C}_i)$.

3 Conjunctive linear logic in FCA

One of the main differences between linear and classical logic is the coexistence of two different conjunctions in linear logic, in both cases the underlying semantics is that two actions are possible, or can be executed, but the difference relies on how these actions are actually performed: on the one hand, we have the multiplicative conjunction $\otimes$ (times) which expresses that both actions will be performed; on the other hand, the additive conjunction & (with) states that, although both actions are possible, actually just one will be performed.

3.1 Additive conjunction

The categorical product $\times$ on ChuCors plays the role of additive conjunction &. Recall that the product $\mathcal{C}_1 \times \mathcal{C}_2$ is defined as $\langle B_1 \uplus B_2, A_1 \uplus A_2, R_{1\times 2}\rangle$ where $R_{1\times 2}(b,a) = \big((i=j) \Rightarrow (b,a) \in R_i\big)$ for all $b \in B_i$ and $a \in A_j$ for all $i, j \in \{1,2\}$.

The semantics of the additive conjunction is that, in order to perform the action of $\mathcal{C}_1 \times \mathcal{C}_2$, we have first to choose which among the two possible actions we want to perform, and then to do the one selected. And this is exactly what happens here. From the previous section about the categorical product of ChuCors, it is known that a concept lattice of a product of formal contexts is equal to a product of concept lattices of the input formal contexts. Hence no interaction or parallel action of input formal contexts occurs in case of the categorical product.

3.2 Multiplicative conjunction and dual bonds

Any dual bond between two formal contexts $\mathcal{C}_1$ and $\mathcal{C}_2$ plays a role of multiplicative conjunction $\otimes$. From the definition of bonds one can see the parallelism of use of input contexts, where every value in the bond, as a relation, is from *both* extents of input contexts.

The existing isomorphism between Chu correspondences and bonds, and monoidal properties of Chu correspondences which follows from the fact that Chu correspondences form a category, bonds satisfies all properties of conjunction.

In the sense of category theory, any dual bond can be seen as a Galois connection between concept lattices of their input contexts, because of the categorical equivalence between category ChuCors and the category of complete lattices and isotone Galois connections.

In [1, 8] one can find that the tensor product is used as multiplicative conjunction, due to its monoidal properties on category of Chu Spaces or on any monoidal category in general. Here, in FCA, the tensor product is a special formal context with nice properties that generates all bonds between the input formal contexts. Hence tensor product shows all possibilities how we can connect or use in parallel two formal contexts.

4 Sequent calculus

The idea here is to develop a logic between contexts and, specifically, a proof theory for this logic.

We will consider the problem of defining a consequence relation $\models$ which allows for developing formally a sequent calculus between contexts.

Definition 9. *Two contexts $\mathcal{C}$ and $\mathcal{D}$ are isomorphic if their concept lattices are isomorphic.*

Definition 10. *Given formal contexts $\mathcal{C}_1, \ldots, \mathcal{C}_n, \mathcal{C}$, we say that $\mathcal{C}$ is a* consequence *of contexts $\mathcal{C}_1, \ldots, \mathcal{C}_n$, denoted as $\mathcal{C}_1, \ldots, \mathcal{C}_n \models \mathcal{C}$, if $\mathcal{C}$ is isomorphic to a bond between all contexts in the left hand side. Specifically, $\mathcal{C}_1, \ldots, \mathcal{C}_n \models \mathcal{C}$ if and only if $\mathcal{C}$ is isomorphic to some n-ary bond between input formal contexts $\mathcal{C}_1, \ldots, \mathcal{C}_n$.*

The relation just defined satisfies the properties of closure operator and, hence, can be considered as a relation of logical consequence between contexts, since any consequence relation can be viewed as a finitary closure operator on a set (of sentences or formulas).

Lemma 5. *The relation $\models$ above is a consequence relation.*

Recall the notion of sequent of Gentzen calculus. Any sequent is of the form $\Gamma \vdash \Delta$ and has the following meaning: *from the conjunction of all hypothesis of Γ follows some formula of Δ*. Hence as a conjunction of all contexts we use some n-ary bond between input contexts.

Without entering into the details, the following sequent rules from conjunctive linear logic can be proved in terms of this definition.

Axiom rule: As a unary bond we use a context itself

$$\frac{}{\mathcal{C} \models \mathcal{C}}$$

Constants rule Due to isomorphism $\mathrm{Bonds}(\mathcal{C}, \top) \cong \mathrm{Ext}(\mathcal{C})$ where $\top = \langle \{\diamond\}, \{\diamond\}, \neq \rangle$ we can write the following rule

$$\frac{\mathcal{C}_1, \ldots, \mathcal{C}_n \models \mathcal{C}}{\top, \mathcal{C}_1, \ldots, \mathcal{C}_n \models \mathcal{C}}$$

$\otimes$-left From the definition of $\models$ and associativity of tensor product, or of associativity inside of n-ary product, we can write

$$\frac{\mathcal{C}_1, \ldots, \mathcal{C}_{n-1}, \mathcal{C}_n, \models \mathcal{C}}{\mathcal{C}_1, \ldots, \mathcal{C}_{n-2}, (\mathcal{C}_{n-1} \otimes \mathcal{C}_n) \models \mathcal{C}}$$

$\otimes$-right Any dual bond between n-ary and m-ary bonds is an $n+m$-ary bond between all input formal contexts

$$\frac{\mathcal{C}_1 \ldots, \mathcal{C}_n \models \mathcal{C} \qquad \mathcal{D}_1 \ldots, \mathcal{D}_m \models \mathcal{D}}{\mathcal{C}_1 \ldots, \mathcal{C}_n, \mathcal{D}_1 \ldots, \mathcal{D}_m \models \mathcal{C} \otimes \mathcal{D}}$$

$\times$**-left** Due to the distributivity of tensor and categorical product on formal contexts, n-ary bond between $\mathcal{C}_1, \ldots, \mathcal{C}_{n-1}, \mathcal{C}_n \times \mathcal{D}$ is equal to product of n-ary bonds between $\mathcal{C}_1, \ldots, \mathcal{C}_{n-1}, \mathcal{C}_n$ and $\mathcal{C}_1, \ldots, \mathcal{C}_{n-1}, \mathcal{D}$. Hence it is easy to use a full relation as the n-ary bond between $\mathcal{C}_1, \ldots, \mathcal{C}_{n-1}, \mathcal{D}$ to obtain the following rule.

$$\frac{\mathcal{C}_1, \ldots, \mathcal{C}_{n-1}, \mathcal{C}_n, \models \mathcal{C}}{\mathcal{C}_1, \ldots, \mathcal{C}_{n-1}, \mathcal{C}_n \times \mathcal{D} \models \mathcal{C}}$$

$\times$**-right** One of the possibilities here is to add to hypothesis a special context, product of two singletons $\top \times \top$.

$$\frac{\mathcal{C}_1 \ldots, \mathcal{C}_n \models \mathcal{D}_1 \qquad \mathcal{C}_1 \ldots, \mathcal{C}_n \models \mathcal{D}_2}{\mathcal{C}_1 \ldots, \mathcal{C}_n \models \mathcal{D}_1 \times \mathcal{D}_2}$$

5 Conclusion

We have obtained a preliminary notion of logical consequence relation between contexts which, together with the interpretation of the multiplicative (resp. additive) conjunction as the cartesian product (resp. bond) of contexts, enable to prove the correctness of the corresponding rules of the sequent calculus of the conjunctive fragment of linear logic.

Of course, we are just scratching the surface of the problem of providing a full calculus since we still need to find the adequate context-related constructions to interpret the rest of connectives. This is future work.

References

1. S. Abramsky and N. Tzevelekos. Introduction to categories and categorical logic. *Lecture Notes in Physics*, 813:3–94, 2011.
2. P. Cellier, S. Ferré, M. Ducassé, and T. Charnois. Partial orders and logical concept analysis to explore patterns extracted by data mining. *Lecture Notes in Artificial Intelligence*, 6828:77–90, 2011.
3. O. Krídlo, S. Krajči, and M. Ojeda-Aciego. The category of L-Chu correspondences and the structure of L-bonds. *Fundamenta Informaticae*, 115(4):297–325, 2012.
4. O. Krídlo and M. Ojeda-Aciego. On L-fuzzy Chu correspondences. *Intl J of Computer Mathematics*, 88(9):1808–1818, 2011.
5. O. Krídlo and M. Ojeda-Aciego. Revising the link between L-Chu correspondences and completely lattice L-ordered sets. *Annals of Mathematics and Artificial Intelligence*, 72:91–113, 2014.
6. M. Krötzsch, P. Hitzler, and G.-Q. Zhang. Morphisms in context. *Lecture Notes in Computer Science*, 3596:223–237, 2005.
7. H. Mori. Chu correspondences. *Hokkaido Mathematical Journal*, 37:147–214, 2008.
8. V. Pratt. Chu spaces as a semantic bridge between linear logic and mathematics. *Theoretical Computer Science*, 294:439–471, 2003.
9. O. Ridoux and S. Ferré. Introduction to logical information systems. *Information Processing and Management*, 40:383–419, 2004.

Morphisms Between Pattern Structures and Their Impact on Concept Lattices

Lars Lumpe[1] and Stefan E. Schmidt[2]

Institut für Algebra,
Technische Universität Dresden
larslumpe@gmail.com[1], midt1@msn.com[2]

Abstract. We provide a general framework for pattern structures by investigating adjunctions between posets and their morphisms. Our special interest is the impact of pattern morphisms on the induced concept lattices. In particular we are interested in conditions which are sufficient for the induced residuated maps to be injective, surjective or bijective. One application is that every representation context of a pattern structure has a formal concept lattice that is induced by a certain pattern morphism.

1 Introduction

Pattern structures within the framework of formal concept analysis have been introduced in [3]. Since then they have turned out to be a useful tool for analysing various real-world applications (cf. [3–7]). Our paper extends the concept of representation contexts and interprets them via morphisms, closely related to o-projections as recently introduced and investigated in [2]. In [8], we disussed the meaning of projections of pattern structures, realizing the importance of residual projections. As a matter of fact, our generalization of representation contexts of pattern structures gives rise to residual projections.

2 Preliminaries

The fundamental order theoretic concepts of our paper are nicely presented in the book on *Residuation Theory* by T.S. Blythe and M.F. Janowitz (cf. [1]).

Definition 1 (Adjunction)**.** *Let $\mathbb{P} = (P, \leqslant)$ and $\mathbb{L} = (L, \leqslant)$ be posets; furthermore let $f : P \to L$ and $g : L \to P$ be maps.*

(1) The pair (f, g) is an **adjunction** *w.r.t. $(\mathbb{P}, \mathbb{L})$ if $fx \leqslant y$ is equivalent to $x \leqslant gy$ for all $x \in P$ and $y \in L$. In this case, we will refer to $(\mathbb{P}, \mathbb{L}, f, g)$ as a* **poset adjunction***.*

(2) f is **residuated** *from $\mathbb{P}$ to $\mathbb{L}$ if the preimage of a principal ideal in $\mathbb{L}$ under f is always a principal ideal in $\mathbb{P}$, that is, for every $y \in L$ there exists $x \in P$ s.t.*

$$f^{-1}\{t \in L \mid t \leqslant y\} = \{s \in P \mid s \leqslant x\}.$$

(3) g is **residual** *from $\mathbb{L}$ to $\mathbb{P}$ if the preimage of a principal filter in $\mathbb{P}$ under g is always a principal filter in $\mathbb{L}$, that is, for every $x \in P$ there exists $y \in L$ s.t.*

$$g^{-1}\{s \in P \mid x \leqslant s\} = \{t \in L \mid y \leqslant t\}.$$

(4) The dual of $\mathbb{L}$ is given by $\mathbb{L}^{\mathrm{op}} = (L, \geqslant)$ with $\geqslant := \{(x, t) \in L \times L \mid t \leqslant x\}$. The pair (f, g) is a **Galois connection** *w.r.t. $(\mathbb{P}, \mathbb{L})$ if (f, g) is an* **adjunction** *w.r.t. $(\mathbb{P}, \mathbb{L}^{\mathrm{op}})$.*

The following well-known facts are straightforward (cf. [1]).

Proposition 1. *Let $\mathbb{P} = (P, \leqslant)$ and $\mathbb{L} = (L, \leqslant)$ be posets.*

(1) A map $f : P \to L$ is residuated from $\mathbb{P}$ to $\mathbb{L}$ iff there exists a map $g : L \to P$ s.t. (f, g) is an adjunction w.r.t. $(\mathbb{P}, \mathbb{L})$.

(2) A map $g : L \to P$ is residual from $\mathbb{L}$ to $\mathbb{P}$ iff there exists a map $f : P \to L$ s.t. (f, g) is an adjunction w.r.t. $(\mathbb{P}, \mathbb{L})$.

(3) If (f, g) and (h, k) are adjunctions w.r.t. $(\mathbb{P}, \mathbb{L})$ with $f = h$ or $g = k$ then $f = h$ and $g = k$.

(4) If f is a residuated map from $\mathbb{P}$ to $\mathbb{L}$, then there exists a unique residual map f^+ from $\mathbb{L}$ to $\mathbb{P}$ s.t. (f, f^+) is an adjunction w.r.t. $(\mathbb{P}, \mathbb{L})$. In this case, f^+ is called the **residual map** *of f.*

(5) If g is a residual map from $\mathbb{L}$ to $\mathbb{P}$, then there exists a unique residuated map g^- from $\mathbb{P}$ to $\mathbb{L}$ s.t. (g^-, g) is an adjunction w.r.t. $(\mathbb{P}, \mathbb{L})$. In this case, g^- is called the **residuated map** *of g.*

(6) A residuated map f from $\mathbb{P}$ to $\mathbb{L}$ is surjective iff $f \circ f^+ = id_L$ iff f^+ is injective.

(7) A residuated map f from $\mathbb{P}$ to $\mathbb{L}$ is injective iff $f \circ f^+ = id_L$ iff f^+ is surjective.

Definition 2. *Let $\mathbb{P} = (P, \leqslant)$ be a poset and $T \subseteq P$. Then*

(1) The **restriction** *of $\mathbb{P}$ onto T is given by $\mathbb{P}|T := (T, \leqslant \cap (T \times T))$, which clearly is a poset too.*

(2) The **canonical embedding** *of $\mathbb{P}|T$ into $\mathbb{P}$ is given by the map $T \to P, t \mapsto t$.*

(3) T is a **kernel system** *in $\mathbb{P}$ if the canonical embedding τ of $\mathbb{P}|T$ into $\mathbb{P}$ is residuated. In this case, the residual map φ of τ will also be called the* **residual map** *of T in $\mathbb{P}$. The composition $\kappa := \tau \circ \varphi$ is referred to as the* **kernel operator** *associated with T in $\mathbb{P}$.*

(4) *Dually, T is a* **closure system** *in $\mathbb{P}$ if the cannonical embedding τ of $\mathbb{P}|T$ into $\mathbb{P}$ is residual. In this case, the residuated map ψ of τ will also be called the* **residuated map** *of T in $\mathbb{P}$. The composition $\gamma := \tau \circ \psi$ is referred to as the* **closure operator** *associated with T in $\mathbb{P}$.*

(5) *A map $\kappa : P \to P$ is a* **kernel operator** *on $\mathbb{P}$ if $s \leqslant x$ is equivalent to $s \leqslant \kappa x$ for all $s \in \kappa P$ and $x \in P$.*
Remark: In this case, κP forms a kernel system in $\mathbb{P}$, the kernel operator of which is κ.

(6) *Dually, a map $\gamma : P \to P$ is a* **closure operator** *on $\mathbb{P}$ if $x \leqslant t$ is equivalent to $\gamma x \leqslant t$ for all $x \in P$ and $t \in \gamma P$.*
Remark: In this case, φP forms a closure system in $\mathbb{P}$, the closure operator of which is γ.

The following known facts will be needed for the sequel (cf. [1]) .

Proposition 2. *Let $\mathbb{P} = (P, \leqslant)$ and $\mathbb{L} = (L, \leqslant)$ be posets.*

(1) *If f is a residuated map from $\mathbb{P}$ to $\mathbb{L}$ then f preserves all existing suprema in $\mathbb{P}$, that is, if $s \in P$ is the supremum (least upper bound) of $X \subseteq P$ in $\mathbb{P}$ then fs is the supremum of fX in $\mathbb{L}$.*
In case $\mathbb{P}$ and $\mathbb{L}$ are complete lattices, the reverse holds too: If a map f from $\mathbb{P}$ to $\mathbb{L}$ preserves all suprema, that is,

$$f(\sup_{\mathbb{P}} X) = \sup_{\mathbb{L}} fX \text{ for all } X \subseteq P,$$

then f is residuated.

(2) *If g is a residual map from $\mathbb{L}$ to $\mathbb{P}$, then g preserves all existing infima in $\mathbb{L}$, that is, if $t \in L$ is the infimum (greatest lower bound) of $Y \subseteq L$ in $\mathbb{L}$ then gt is the infimum of gY in $\mathbb{P}$.*
In case $\mathbb{P}$ and $\mathbb{L}$ are complete lattices, the reverse holds too: If a map g from $\mathbb{L}$ to $\mathbb{P}$ preserves all infima, that is,

$$f(\inf_{\mathbb{P}} Y) = \inf_{\mathbb{L}} gY \text{ for all } Y \subseteq L,$$

then g is residual.

(3) *For an adjunction (f, g) w.r.t. $(\mathbb{P}, \mathbb{L})$ the following hold:*

(a1) f is an isotone map from $\mathbb{P}$ to $\mathbb{L}$.

(a2) $f \circ g \circ f = f$

(a3) fP is a kernel system in $\mathbb{L}$ with $f \circ g$ as associated kernel operator on $\mathbb{L}$. In particular, $L \to P, y \mapsto fgy$ is a residual map from $\mathbb{L}$ to $\mathbb{L}|fP$.

(b1) g is an isotone map from $\mathbb{L}$ to $\mathbb{P}$.

(b2) $g \circ f \circ g = g$

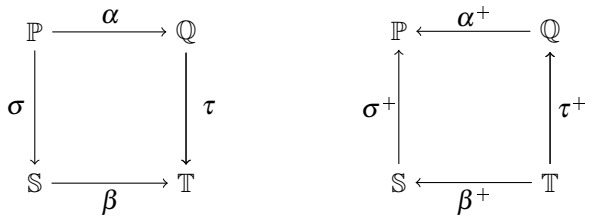
(b3) gL is a closure system in $\mathbb{P}$ with $g \circ f$ as associated closure operator on $\mathbb{P}$. In particular, $P \to gL, x \mapsto gfx$ is a residuated map from $\mathbb{P}$ to $\mathbb{P}|gL$.

3 Adjunctions and Their Concept Posets

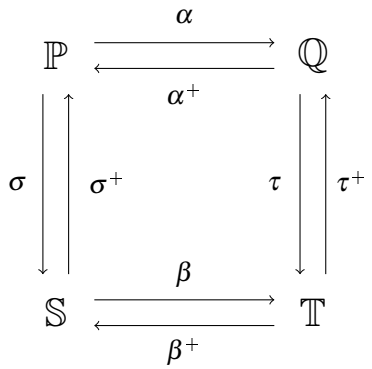
Definition 3. *Let $\mathcal{P} := (\mathbb{P}, \mathbb{S}, \sigma, \sigma^+)$ and $\mathcal{Q} := (\mathbb{Q}, \mathbb{T}, \tau, \tau^+)$ be poset adjunctions. Then a pair (α, β) forms a morphism from $\mathcal{P}$ to $\mathcal{Q}$ if $(\mathbb{P}, \mathbb{Q}, \alpha, \alpha^+)$ and $(\mathbb{S}, \mathbb{T}, \beta, \beta^+)$ are poset adjunctions satisfying*

$$\tau \circ \alpha = \beta \circ \sigma$$

Remark: This implies $\alpha^+ \circ \tau^+ = \sigma^+ \circ \beta^+$, that is, the following diagrams are commutative:



Next we illustrate the involved poset adjunctions:



Definition 4 (Concept Poset)**.** *For a poset adjunction $\mathcal{P} = (\mathbb{P}, \mathbb{S}, \sigma, \sigma^+)$ let*

$$\mathrm{B}\mathcal{P} := \{(p, s) \in P \times S \mid \sigma p = s \wedge \sigma^+ s = p\}$$

denote the set of (**formal**) **concepts** *in $\mathcal{P}$. Then the* **concept poset** *of $\mathcal{P}$ is given by*

$$\mathbb{B}\mathcal{P} := (\mathbb{P} \times \mathbb{S}) \mid \mathrm{B}\mathcal{P},$$

that is, $(p_0, s_0) \leqslant (p_1, s_1)$ holds iff $p_0 \leqslant p_1$ iff $s_0 \leqslant s_1$, for all $(p_0, s_0), (p_1, s_1) \in B\mathcal{P}$. If (p, s) is a formal concept in $\mathcal{P}$ then p is referred to as **extent** *in $\mathcal{P}$ and s as* **intent** *in $\mathcal{P}$.*

From [9] we point out Theorem 1:

Theorem 1. *Let (α, β) be a morphism from a poset adjunction $\mathcal{P} = (\mathbb{P}, \mathbb{S}, \sigma, \sigma^+)$ to a poset adjunction $\mathcal{Q} = (\mathbb{Q}, \mathbb{T}, \tau, \tau^+)$. Then*

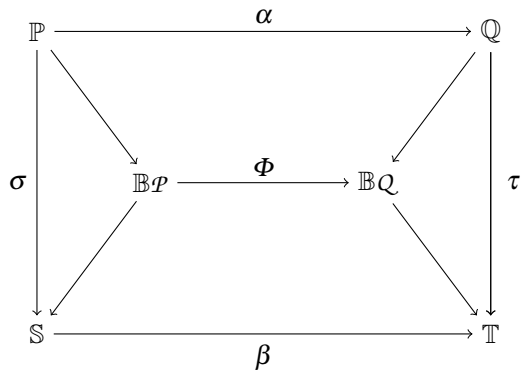
$$(\mathbb{B}\mathcal{P}, \mathbb{B}\mathcal{Q}, \Phi, \Phi^+)$$

is a poset adjunction for

$$\Phi : \mathrm{B}\mathcal{P} \to \mathrm{B}\mathcal{Q}, (p,s) \mapsto (\tau^+\beta s, \beta s)$$

and

$$\Phi^+ : \mathrm{B}\mathcal{Q} \to \mathrm{B}\mathcal{P}, (q,t) \mapsto (\alpha^+ q, \sigma\alpha^+ q).$$



Theorem 2. *Under the conditions of the previous theorem the following hold:*

(1) If α is surjective then Φ is surjective too.
(2) If β is injective then Φ is injective too.
(3) If α is surjective and β is injective then Φ is an isomorphism from $\mathrm{B}\mathcal{P}$ to $\mathrm{B}\mathcal{Q}$..

Proof. (1) Assume that α is surjective, that is, $\alpha \circ \alpha^+ = id_Q$. Then for all $(p,s) \in \mathrm{B}\mathcal{P}$, the second component of $(\Phi \circ \Phi^+)(p,s)$ is $\beta\sigma\alpha^+ q = \tau\alpha\alpha^+ q = \tau q = s$. This yields $\Phi \circ \Phi^+ = id_{\mathrm{B}\mathcal{Q}}$, that is, Φ is surjective.
(2) The first component of $(\Phi^+ \circ \Phi)(p,s)$ is $\alpha^+\tau + \beta s = \tau + \beta^+\beta s = \tau + s = p$. Therefore, $\Phi^+ \circ \Phi = id_{\mathrm{B}\mathcal{P}}$, which yields Φ being injective.
(3) If α is surjective and β is injective, than Φ and Φ^+ are naturally inverse by (1) and (2), that is, Φ is an isomorphism from $\mathrm{B}\mathcal{P}$ to $\mathrm{B}\mathcal{Q}$.

□

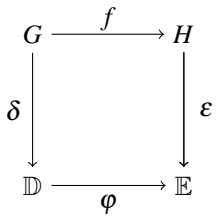
4 The Impact of Pattern Morphism on Concept Lattices

Definition 5. *A triple $\mathcal{G} = (G, \mathbb{D}, \delta)$ is a* **pattern setup** *if G is a set, $\mathbb{D} = (D, \sqsubseteq)$ is a poset, and $\delta : G \to D$ is a map. In case every subset of $\delta G := \{\delta g \mid g \in G\}$ has an infimum in $\mathbb{D}$, we will refer to $\mathcal{G}$ as* **pattern structure***. Then the set*

$$\mathrm{C}_{\mathcal{G}} := \{\inf_{\mathbb{D}} \delta X \mid X \subseteq G\}$$

forms a closure system in $\mathbb{D}$ and furthermore $\mathbb{C}_{\mathcal{G}} := \mathbb{D}|C_{\mathcal{G}}$ forms a complete lattice.

If $\mathcal{G} = (G, \mathbb{D}, \delta)$ and $\mathcal{H} = (H, \mathbb{E}, \varepsilon)$ each is a pattern setup, then a pair (f, φ) forms a **pattern morphism** *from $\mathcal{G}$ to $\mathcal{H}$ if $f : G \to H$ is a map and φ is a residual map from $\mathbb{D}$ to $\mathbb{E}$ satisfying $\varphi \circ \delta = \varepsilon \circ f$, that is, the following diagram is commutative:*



In the sequel we show how our previous considerations apply to pattern structures.

Theorem 3. *Let (f,φ) be a pattern morphism from a pattern structure $\mathcal{G} = (G,\mathbb{D},\delta)$ to a pattern structure $\mathcal{H} = (H,\mathbb{E},\varepsilon)$.*
To apply the previous theorem we give the following construction:
f gives rise to an adjunction (α,α^+) between the power set lattices $\mathbf{2}^G := (2^G,\subseteq)$ and $\mathbf{2}^H := (2^H,\subseteq)$ via

$$\alpha : 2^G \to 2^H, X \mapsto fX$$

and

$$\alpha^+ : 2^H \to 2^G, Y \mapsto f^{-1}Y.$$

Further let φ^- denote the residuated map of φ w.r.t. $(\mathbb{E},\mathbb{D})$, that is, $(\mathbb{E},\mathbb{D},\varphi^-,\varphi)$ is a poset adjunction. Then, obviously, $(\mathbb{D}^{\mathrm{op}},\mathbb{E}^{\mathrm{op}},\varphi,\varphi^-)$ is a poset adjunction too.

For pattern structures the following operators are essential:

$$\begin{aligned}
&{}^{\diamond} : 2^G \to D, X \mapsto \inf_{\mathbb{D}} \delta X\\
&{}^{\blacklozenge} : D \to 2^G, d \mapsto \{g \in G \mid d \sqsubseteq \delta g\}\\
&{}^{\square} : 2^H \to E, Z \mapsto \inf_{\mathbb{E}} \varepsilon Z\\
&{}^{\blacksquare} : E \to 2^H, e \mapsto \{h \in H \mid e \sqsubseteq \varepsilon h\}
\end{aligned}$$

It now follows that (α,φ) forms a morphism from the poset adjunction

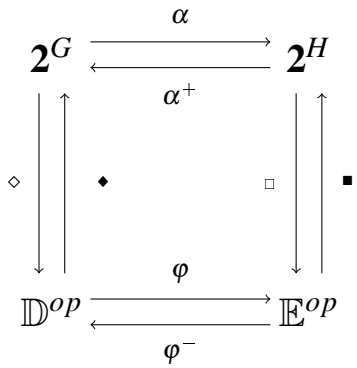
$$\mathcal{P} = (\mathbf{2}^G,\mathbb{D}^{\mathrm{op}},{}^{\diamond},{}^{\blacklozenge})$$

to the poset adjunction

$$\mathcal{Q} = (\mathbf{2}^H,\mathbb{E}^{\mathrm{op}},{}^{\square},{}^{\blacksquare}).$$

In particular, $(fX)^{\square} = \varphi(X^{\diamond})$ holds for all $X \subseteq G$.

We receive the following diagram of adjunctions:

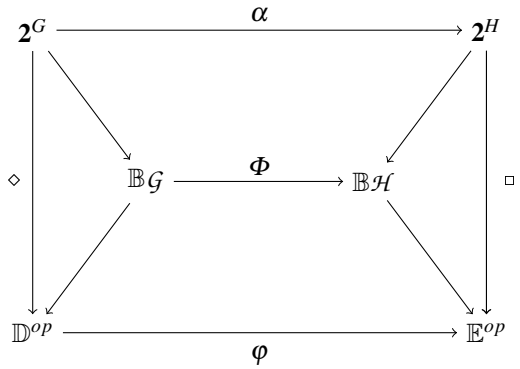


For the following we recall that the concept lattice of $\mathcal{G}$ is given by $\mathbb{B}\mathcal{G} := \mathbb{B}\mathcal{P}$ and the concept lattice of $\mathcal{H}$ is $\mathbb{B}\mathcal{H} := \mathbb{B}\mathcal{Q}$.
Then Theorem 1 yields that the quadruple $(\mathbb{B}\mathcal{G}, \mathbb{B}\mathcal{H}, \Phi, \Phi^+)$ is an adjunction for

$$\Phi : \mathrm{B}\mathcal{G} \to \mathrm{B}\mathcal{H}, (X,d) \mapsto ((\varphi d)^{\blacksquare}, \varphi d)$$

and

$$\Phi^+ : \mathrm{B}\mathcal{H} \to \mathrm{B}\mathcal{G}, (Z,e) \mapsto (f^{-1}Z, (f^{-1}Z)^{\diamond}).$$



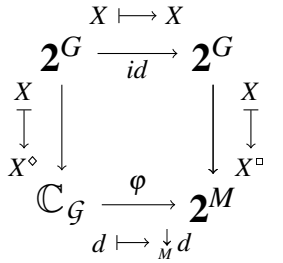
By Theorem 2 the following hold:

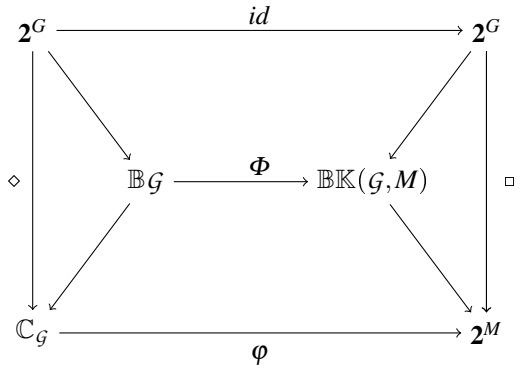
(1) If f is surjective then Φ is surjective too.
(2) If φ is injective then Φ is injective too.
(3) If f is surjective and φ is injective then Φ is an isomorphism from $\mathbb{B}\mathcal{G}$ to $\mathbb{B}\mathcal{H}$.

Theorem 4. *Let $\mathcal{G} = (G, \mathbb{D}, \delta)$ and $\mathcal{H} = (H, \mathbb{E}, \varepsilon)$ be pattern structure. And let $\mathcal{G}^\bullet = (G, \mathbb{C}_\mathcal{G}, \delta^\bullet)$ be the pattern structure induced by $\mathcal{G}$ via $\delta^\bullet : G \to C_\mathcal{G}$, $g \mapsto \delta g$. It follows $\mathbb{B}\mathcal{G}^\bullet = \mathbb{B}\mathcal{G}$. Further let (f, φ) be a pattern morphism from $\mathcal{G}^\bullet$ to $\mathcal{H}$. Then with the notation introduced in the previous theorem, the map Φ from $\mathbb{B}\mathcal{G}$ to $\mathbb{B}\mathcal{H}$ is residuated. If f is surjective then so is Φ, if φ is injective then so is Φ. If f is surjective and φ is injective then Φ is an isomorphism from $\mathbb{B}\mathcal{G}$ to $\mathbb{B}\mathcal{H}$.*

Definition 6. *The **representation context** of a pattern structure $\mathcal{G} = (G, \mathbb{D}, \delta)$ w.r.t. a subset M of D is given by $\mathbb{K}(\mathcal{G}, M) := (G, M, I)$ with $I := \{(g, m) \in G \times M \mid m \sqsubseteq \delta g\}$.*

Theorem 5. *Let $\mathcal{G} = (G, \mathbb{D}, \delta)$ be a pattern structure and let M be a subset of D.*
The pattern structure associated with the representation context $\mathbb{K}(\mathcal{G}, M)$ is given by $\mathcal{H} := (G, \mathbf{2}^M, \varepsilon)$ with $\varepsilon : G \to 2^M, g \mapsto \downarrow_M \delta g$ where $\downarrow_M d := \{m \in M \mid m \sqsubseteq d\}$ for all $d \in D$.
In particular, the concept lattice of $\mathbb{K}(\mathcal{G}, M)$ is given by $\mathbb{BK}(\mathcal{G}, M) = \mathbb{B}\mathcal{H}$.
Using the notation from the previous theorem, (id_G, φ) is a pattern morphism from $\mathcal{G}^\bullet$ to $\mathcal{H}$ for $\varphi : C_\mathcal{G} \to 2^M, x \mapsto \downarrow_M x$. Furthermore, the map Φ from $\mathbb{B}\mathcal{G}$ to $\mathbb{B}\mathcal{H} = \mathbb{BK}(\mathcal{G}, M)$ is a residuated surjection. In case M is join-dense w.r.t. $\mathbb{C}_\mathcal{G}$ (that is, φ is injective), Φ is an isomorphism from $\mathbb{B}\mathcal{G}$ to $\mathbb{BK}(\mathcal{G}, M)$.





Remark: Based on the paradigm of concept morphisms, the previous theorem extends and sheds new light on theorem 1 of [3]. We generalize the definition of representation context introduced in [3] by allowing an abitrary subset M of patterns of the underlying pattern structure $\mathcal{G}$ as attribute set of the representation context $\mathbb{K}(\mathcal{G}, M)$. It then turns out that $\mathbb{K}(\mathcal{G}, M)$ has a formal concept lattice which is induced by a morphism on $\mathcal{G}^\bullet$. More explicitly, there is a morphism on $\mathcal{G}^\bullet$ to the pattern structure of $\mathbb{K}(\mathcal{G}, M)$ which induces a residuated surjection from the concept lattice of $\mathcal{G}$ to the concept lattice of $\mathbb{K}(\mathcal{G}, M)$. In case M is join-dense w.r.t. $\mathcal{G}$, the morphism between the concept lattices is an isomorphism (see also Theorem 1 of [3]). Our extension of the concept of representation context gives rise to various constructions of o-projections (as introduced in [2]) on $\mathcal{G}^\bullet$.

References

1. T.S. Blyth, M.F.Janowitz (1972), Residuation Theory, Pergamon Press, pp. 1-382.
2. A. Buzmakov, S. O. Kuznetsov, A. Napoli (2015) , Revisiting Pattern Structure Projections. Formal Concept Analysis. Lecture Notes in Artificial Intelligence (Springer), Vol. 9113, pp 200-215.
3. B. Ganter, S. O. Kuznetsov (2001), Pattern Structures and Their Projections. Proc. 9th Int. Conf. on Conceptual Structures, ICCS01, G. Stumme and H. Delugach (Eds.). Lecture Notes in Artificial Intelligence (Springer), Vol. 2120, pp. 129-142.
4. T. B. Kaiser, S. E. Schmidt (2011), Some remarks on the relation between annotated ordered sets and pattern structures. Pattern Recognition and Machine Intelligence. Lecture Notes in Computer Science (Springer), Vol. 6744, pp 43-48.
5. M. Kaytoue, S. O. Kuznetsov, A. Napoli, S. Duplessis (2011), Mining gene expression data with pattern structures in formal concept analysis. Information Sciences (Elsevier), Vol.181, pp. 1989-2001.
6. S. O. Kuznetsov (2009), Pattern structures for analyzing complex data. In H. Sakai et al. (Eds.). Proceedings of the 12th international conference on rough sets, fuzzy sets, data mining and granular computing (RSFDGrC09). Lecture Notes in Artificial Intelligence (Springer), Vol. 5908, pp. 33-44.
7. S. O. Kuznetsov (2013), Scalable Knowledge Discovery in Complex Data with Pattern Structures. In: P. Maji, A. Ghosh, M.N. Murty, K. Ghosh, S.K. Pal, (Eds.). Proc. 5th International Conference Pattern Recognition and Machine Intelligence (PReMI2013). Lecture Notes in Computer Science (Springer), Vol. 8251, pp. 30-41.
8. L. Lumpe, S. E. Schmidt (2015), A Note on Pattern Structures and Their Projections. Formal Concept Analysis. Lecture Notes in Artificial Intelligence (Springer), Vol. 9113, pp 145-150.
9. L. Lumpe, S. E. Schmidt (2015), Pattern Structures and Their Morphisms. CLA 2015, pp 171-179.

A Reachability-based Navigation Paradigm for Triadic Concepts

Diana Troancă

Babeş-Bolyai University Cluj Napoca
dianat@cs.ubbcluj.ro

Abstract. Formal Concept Analysis offers a simple formalization for representing knowledge structures extracted from various data. Lately, the triadic case has become increasingly popular, given that data can often be interpreted in a triadic setting for further processing and analysis. However, visualization and navigation in triadic conceptual landscapes is not trivial and, so far, there are no tools implementing navigation in triconcept sets. This paper extends a navigation paradigm based on a reachability relation of triconcepts and on appropriately defined dyadic projections, and it offers a detailed description of different implementation methods. Moreover, we propose a visualization of the reachability clusters that gives an overview of the triconcepts' structure and can assist the local navigation.

1 Introduction

Nowadays, understanding big collections of data can have a great impact on advancing different scientific fields. Formal Concept Analysis (FCA) provides a powerful mathematical tool that addresses knowledge processing and knowledge representation [3]. The main advantage of FCA is the intuitive visualization and navigation methods offered by concept lattices. FCA was extended to the triadic case by Lehman and Wille in 1995 [6]. Since then, different theoretical aspects were studied and extended from the dyadic to the triadic case, and trilattices were proposed as a visualization method. However, this type of representation does not support an intuitive navigation method. Moreover, for slightly larger triadic data sets, the complexity of the representation makes any navigation attempt useless. Therefore, the problem of visualization and navigation in triadic conceptual spaces needs to be further analyzed and new approaches have to be found.

Previously, we have proposed two methods of navigating in triadic data. The first approach is based on narrowing down the space of formal concepts according to constraints added by the user. This membership-constraint-based approach was formally described and the theoretical aspects were studied in detail [7]. Moreover, the navigation paradigm was implemented, tested and evaluated [1, 9]. This approach, however, generates lists of elements, hence the users cannot visualize the underlying structure of the triconcept set.

The second approach has a local character and is based on appropriately defined dyadic projections. For this purpose, we defined the *reachability* relation and studied some of its properties, as well as the properties of other resulting structures, such as reachability *clusters* [8]. In our previous paper, we analyzed the theoretical aspects of the described paradigm and shortly sketched a navigation strategy without going into details about the implementation methods. This paper aims to extend the navigation paradigm and to offer a comprehensive description of strategy behind. Moreover, we show how the structure of the reachability clusters can be used for supporting the local navigation paradigm.

2 Preliminaries

This section introduces the basic notions of triadic formal concept analysis as well as some of the theoretical aspects of the reachability-based navigation. For a deeper understanding of formal concept analysis we refer the reader to the standard literature [3, 6], while a more detailed discussion on the properties of the reachability relation can be found in our previous paper [8].

The fundamental structures of triadic formal concept analysis are those of a triadic formal context and a triconcept.

Definition 1. *A* triadic formal context*, also referred to as a* tricontext*, is a quadruple* $\mathbb{K} = (K_1, K_2, K_3, Y)$ *consisting of three sets* K_1, K_2, K_3 *and a ternary relation* $Y \subseteq K_1 \times K_2 \times K_3$*. The elements of* K_1, K_2, K_3 *are called* (formal) object, attributes *and* conditions*. An element* $(g, m, b) \in Y$ *of the incidence relation is read* object g has attribute m under condition b*.*

Definition 2. *The* triadic concepts*, also called* triconcepts*, of a tricontext* $\mathbb{K} = (K_1, K_2, K_3, Y)$ *are exactly the triples* (A_1, A_2, A_3) *that satisfy* $A_1 \times A_2 \times A_3 \subseteq Y$ *and which are maximal w.r.t. component-wise set inclusion.*

The following definition shows how dyadic projections can be obtained from a triadic context.

Definition 3. *Every triadic context* (K_1, K_2, K_3, Y) *gives rise to the following dyadic contexts:*

$\mathbb{K}^{(1)} := (K_1, K_2 \times K_3, Y^{(1)})$ *with* $gY^{(1)}(m, b) :\Leftrightarrow (g, m, b) \in Y$*,*
$\mathbb{K}^{(2)} := (K_2, K_1 \times K_3, Y^{(2)})$ *with* $mY^{(2)}(g, b) :\Leftrightarrow (g, m, b) \in Y$*, and*
$\mathbb{K}^{(3)} := (K_3, K_1 \times K_2, Y^{(3)})$ *with* $bY^{(3)}(g, m) :\Leftrightarrow (g, m, b) \in Y$*.*

For $\{i, j, k\} = \{1, 2, 3\}$ *and* $A_k \subseteq K_k$*, we define* $\mathbb{K}^{(ij)}_{A_k} := (K_i, K_j, Y^{(ij)}_{A_k})$*, where* $(a_i, a_j) \in Y^{(ij)}_{A_k}$ *if and only if* $(a_i, a_j, a_k) \in Y$ *for all* $a_k \in A_k$*.*

Intuitively, the contexts $\mathbb{K}^{(i)}$ represent "flattened" versions of the triadic context, obtained by putting the "slices" of (K_1, K_2, K_3, Y) side by side. Moreover, $\mathbb{K}^{(ij)}_{A_k}$ corresponds to the intersection of all those slices that correspond to elements of A_k.

Next we introduce the notion of *reachability* relation and *reachability cluster*.

Definition 4. *For* (A_1, A_2, A_3) *and* (B_1, B_2, B_3) *triadic concepts, we say that* (B_1, B_2, B_3) *is* directly reachable *from* (A_1, A_2, A_3) using perspective (1) *and we write* $(A_1, A_2, A_3) \prec_1 (B_1, B_2, B_3)$ *if and only if* $(B_2, B_3) \in \mathfrak{B}(\mathbb{K}^{(23)}_{A_1})$. *Analogously, we can define direct reachability using perspectives* (2) *and* (3).

We say that (B_1, B_2, B_3) *is* directly reachable *from* (A_1, A_2, A_3) *if it is directly reachable using at least one of the three perspectives, that is, formally* $(A_1, A_2, A_3) \prec (B_1, B_2, B_3) :\Leftrightarrow [(A_1, A_2, A_3) \prec_1 (B_1, B_2, B_3)] \vee [(A_1, A_2, A_3) \prec_2 (B_1, B_2, B_3)] \vee [(A_1, A_2, A_3) \prec_3 (B_1, B_2, B_3)]$.

Definition 5. *We define the* reachability relation *between two triconcepts as being the transitive closure of the direct reachability relation* $\prec$. *We denote this relation by* $\lhd$.

For checking whether triconcept (B_1, B_2, B_3) is directly reachable from triconcept (A_1, A_2, A_3), we have proposed the following algorithm.

Algorithm 1: Procedure directlyReachable((A_1, A_2, A_3), (B_1, B_2, B_3))

If $A_1 = B_1$ **or** $A_2 = B_2$ **or** $A_3 = B_3$ **then**
    Return true

If $A_1 \subset B_1$ **then**
    $P_e = \mathbb{K}^{(23)}_{A_1}$
    If $(B_2)'_{P_e} = B_3$ **and** $(B_3)'_{P_e} = B_2$ **then**
        Return true

If $A_2 \subset B_2$ **then**
    $P_i = \mathbb{K}^{(13)}_{A_2}$
    If $(B_1)'_{P_e} = B_3$ **and** $(B_3)'_{P_e} = B_1$ **then**
        Return true

If $A_3 \subset B_3$ **then**
    $P_m = \mathbb{K}^{(12)}_{A_3}$
    If $(B_1)'_{P_e} = B_2$ **and** $(B_2)'_{P_e} = B_1$ **then**
        Return true

Return false

The derivations used in the description of the algorithm are the simple dyadic derivations and the index was added just to highlight that each dyadic derivation corresponds to a different dyadic context. For example, $(B_2)'_{P_e} = B_3$ uses the dyadic derivation operator of the context P_e.

When studying the properties of the reachability relation, the notion of reachability cluster arises. Intuitively, a reachability cluster is a maximal set of mutually reachable triconcepts. Formally, a reachability cluster is defined as follows.

Definition 6. *The equivalence class of a triconcept* (A_1, A_2, A_3) *with respect to the preorder* $\lhd$ *on* $\mathfrak{T}(\mathbb{K})$ *will be called a* reachability cluster *and denoted by* $[(A_1, A_2, A_3)]$.

Next, we consider the dyadic context of reachable triconcepts $\mathbb{K}_{\lhd}$.

Definition 7.
Let $\mathbb{K} = (K_1, K_2, K_3, Y)$ *be a triadic context. Then we denote with* $\mathbb{K}_{\lhd} = (\mathfrak{T}(\mathbb{K}), \mathfrak{T}(\mathbb{K}), \lhd)$ *the formal context of triconcepts with the reachability relation.*

While analyzing the correlation between reachability clusters and the dyadic concepts $(M, N) \in \mathfrak{B}(\mathbb{K}_{\lhd})$ of the reachability context, we have shown that there is a one-to-one relation between the clusters and the non-empty intersections of the extent and intent from the dyadic concepts $M \cap N$. These results give rise to a display method of all reachability clusters that can support the navigation among triconcepts, as detailed in the next sections.

3 Navigation Strategy

Considering the theoretical aspects highlighted in Section 2 and described in more detail in our previous paper ([8]), we propose a strategy for navigating among triconcepts inside a reachability cluster as well as between clusters. In addition, as guidance during the navigation, we suggest the use of the cluster structure that shows an overview of the reachable triconcepts. Intuitively, a step of the navigation paradigm consists of moving from one triconcept to another directly reachable triconcept. Hence, by following a navigation path of several steps one can explore the triadic conceptual knowledge landscape.

One problem that has not been solved yet in an efficient manner is how to choose a starting point. Currently, for this purpose, we use a preprocessing step that computes all triconcepts, for example using `Trias` [4]. Once we have the triconcept set, one can choose a triconcept as a starting point and navigate by choosing one perspective, i.e. one of the three dimensions. However, we highlight the fact that this preprocessing step is not necessary for the rest of the navigation and, if a starting point is chosen using a different method, this time consuming step can be eliminated. The local navigation paradigm is described by the following steps:

- choose a triconcept $T = (A_1, A_2, A_3)$ and a perspective (i) with $i \in \{1, 2, 3\}$
- compute the derived context $\mathbb{K}_{A_i}^{(jk)}$ of the triadic relation with $j, k \in \{1, 2, 3\} \setminus \{i\}$ s.t. $j < k$
- generate the concept lattice of $\mathbb{K}_{A_i}^{(jk)}$
- attach as labels to the dyadic concepts in the lattice the corresponding triconcepts by adding the third component
- choose one of the triconcepts that are represented by the nodes in the dyadic lattice as a next step

For computing the derived context $\mathbb{K}_{A_i}^{(jk)}$, one must select from the triadic relation the pairs of elements $(a_j, a_k) \in A_j \times A_k$ which are in relation with all elements from A_i, i.e. $(a_i, a_j, a_k) \in Y, \forall a_i \in A_i$, assuming, without loss of generality, that $i < j < k$. Afterwards, the third step consists of generating the concept lattice of the derived context. This can be done using one of the existing FCA tools [2]. In the next step, for a dyadic concept (B_j, B_k) of the derived context $\mathbb{K}_{A_i}^{(jk)}$ we must identify the corresponding triconcept $(B_i, B_j, B_k) \in \mathfrak{T}(\mathbb{K})$. Theoretically, this can be done by using the corresponding derivation operator to compute the third component of a triconcept. However, considering that we have already computed all triconcepts, it is more efficient to select the triconcept having the two components B_j and B_k (which will be unique given the maximality condition) from the triconcept set.

The previously described navigation and visualization paradigm has a local character which is an advantage when considering large contexts. However, one disadvantage of the navigation is that not every triconcept can be reached from every other triconcept, although this seems to be the case in most practical scenarios [8]. Therefore, we believe that it is useful to have an overall view of the navigation clusters' structure in order to understand whereto one can navigate. With that purpose, we obtain the lattice structure of the reachability context $\mathbb{K}_{\lhd} = (\mathfrak{T}(\mathbb{K}), \mathfrak{T}(\mathbb{K}), \lhd)$ as follows:

- compute the direct reachability relation between triconcepts
- compute the transitive closure of the direct reachability relation
- represent the concept lattice of clusters

For implementing the first step, we can use Algorithm 1 that outputs whether a triconcept is directly reachable from another triconcept. For the second step, the transitive closure of the direct reachability relation $\prec$ must be computed in order to obtain the reachability relation $\lhd$. This can be done by using one of the existing algorithms that compute the transitive closure of a relation, such as `Warshall` algorithm ([11]), `Warren` algorithm ([10]), etc.

After obtaining that, the reachability context $\mathbb{K}_{\lhd}$ can be formed and we can compute the clusters and the partial order on the cluster set, using the concept lattice of $\mathbb{K}_{\lhd}$. We have shown that each reachability cluster is uniquely identified by the intersection of extent and intent of exactly one dyadic concept of $\mathbb{K}_{\lhd}$. Hence, we can compute the concept lattice of $\mathbb{K}_{\lhd}$ and label each node with the intersection of extent and intent, i.e. the corresponding cluster, and no label when the intersection is empty. In the obtained lattice one can visualize the partial order between the clusters. An example illustrating the described procedure for obtaining the cluster structure is presented in Section 4.

Alternatively, we can make use of the directed graph having the triconcepts as vertices and the edges given by the direct reachability relation. Here, we can identify the reachability clusters, as well as deduce the transitive closure of the direct reachability relation as described in Proposition 1.

Proposition 1. *Let $\mathbb{K}$ be a tricontext and G the graph with $\mathfrak{T}(\mathbb{K})$ as vertices and the edges given by the direct reachability relation. Then, the reachability clusters*

of $\mathbb{K}$ are identified by the strongly connected components of G. Furthermore, for triconcepts $T_1, T_2 \in \mathfrak{T}(\mathbb{K})$, we have that $T_1 \triangleleft T_2$ if, in graph G, there is a directed path from T_1 to T_2.

Furthermore, the same graph G can be used to deduce the partial order relation between the clusters as described in Proposition 2.

Proposition 2. *Let $\mathbb{K}$ be a tricontext and G the graph with $\mathfrak{T}(\mathbb{K})$ as vertices and the edges given by the direct reachability relation. If $T_1 \in C_1$ and $T_2 \in C_2$ are two triconcepts from different clusters s.t. in G there is a path from T_1 to T_2, then we have that $C_1 \leq C_2$. Observe that, considering T_1 and T_2 belong to different clusters, in the case that there is a path from T_1 to T_2, we cannot have a path from T_2 to T_1.*

However, a disadvantage of the graph-based approach is that it does not output the lattice representation of the clusters, hence an FCA tool has to be used if we want to obtain a visualization of the cluster structure.

An important aspect to keep in mind during the navigation is that not every triconcept is reachable from any other triconcept. In fact, when navigating from a triconcept to another belonging to a different cluster, one cannot navigate back by using the standard navigation method described. For this purpose, we propose the use of a navigation history, allowing the user to go back to a certain triconcept in the previous navigation path.

In conclusion, to support exploration through the dataset, we suggest using both the local visualization method for triconcepts and the visualization of the cluster structure as follows. We compute the lattice showing the cluster structure at the beginning of the navigation and make it available to the user throughout the whole exploration process. Then, at each step of the navigation, when visualizing the concept lattice of the possible next steps, i.e. the directly reachable triconcepts, we highlight all the triconcepts belonging to the same cluster as the current triconcept. In that way, the user can easily choose to navigate within the same cluster or to a different one. Furthermore, looking at the cluster structure in parallel, the user can navigate more easily towards a potential goal of the navigation.

4 Example

In practice, experiments that we ran on real datasets showed that tricontexts had one single reachability cluster comprising all triconcepts. This leads us to believe, that, in general, real datasets have a high correlation in the data and therefore, all triconcepts are contained in the same reachability cluster. However, theoretically it is possible that a tricontext comprises several reachability cluster. In this section, we present a small example of an artificial context containing more than two reachability clusters and exemplify how the structure of the reachability clusters can be obtained.

Let us consider the following triadic context $\mathbb{K} = (G, M, B, Y)$, with the object set $G = \{g_1, g_2, g_3\}$, the attribute set $M = \{m_1, m_2, m_3\}$ and the condition set $B = \{b_1, b_2, b_3\}$.

b1	m1	m2	m3
g1	×		
g2			
g3			

b2	m1	m2	m3
g1			
g2			
g3		×	

b3	m1	m2	m3
g1	×	×	
g2			
g3			

The triconcepts of this context are the following:

$T_1 = (\{g_3\}, \{m_2\}, \{b_2\})$
$T_2 = (\{g_1\}, \{m_1\}, \{b_1.b_3\})$
$T_3 = (\{g_1\}, \{m_1.m_2\}, \{b_3\})$
$T_4 = (\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\}, \emptyset)$
$T_5 = (\{g_1, g_2, g_3\}, \emptyset, \{b_1, b_2, b_3\})$
$T_6 = (\emptyset, \{m_1, m_2, m_3\}, \{b_1, b_2, b_3\})$

The triconcepts are partitioned in clusters the following way:

$C_1 = \{(\{g_3\}, \{m_2\}, \{b_2\})\}$
$C_2 = \{(\{g_1\}, \{m_1\}, \{b_1.b_3\}), (\{g_1\}, \{m_1.m_2\}, \{b_3\})\}$
$C_3 = \{(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\}, \emptyset), (\{g_1, g_2, g_3\}, \emptyset, \{b_1, b_2, b_3\}),$
$(\emptyset, \{m_1, m_2, m_3\}, \{b_1, b_2, b_3\})\}$

Then, we obtain the dyadic context of reachability $\mathbb{K}_\triangleleft$ as depicted in Figure 1.

$\mathbb{K}_\triangleleft$	T_1	T_2	T_3	T_4	T_5	T_6
T_1	×	×	×	×	×	×
T_2		×	×	×	×	×
T_3		×	×	×	×	×
T_4				×	×	×
T_5				×	×	×
T_6				×	×	×

Fig. 1: Dyadic context of reachability $\mathbb{K}_\triangleleft$

Fig. 2: Concept lattice of $\mathbb{K}_\triangleleft$

Moreover, when computing the concept lattice of $\mathbb{K}_\triangleleft$, we obtain a representation of the reachability clusters as depicted in Figure 2. In this lattice, the bottom node corresponds to cluster C_1, the node in the middle corresponds to cluster C_2 and the upper node corresponds to cluster C_3. The partial order between the clusters can be read from the lattice the following way: if one can navigate from cluster C_1 to cluster C_2 going upwards, then $C_1 \leq C_2$, so we have that $C_1 \leq C_2 \leq C_3$.

During the navigation, taking this cluster structure into consideration, one can deduce, for example, that starting from triconcept T_4 you can never reach any of the triconcepts T_1, T_2 or T_3. Hence, the structure of the clusters can be of use also when choosing a starting point for the navigation.

5 Conclusions and Future Work

In this paper, we have extended a navigation paradigm for triadic datasets that helps the user to get an overview of the data and to understand its underlying structure. To this end we described the steps of the navigation among triconcepts based on local dyadic projections and we have shown how the structure of the reachability clusters can be obtained using different methods. Furthermore, we have highlighted the fact that the local navigation paradigm can benefit from the lattice representation of the clusters' structure by offering the user an overview of the underlying data structure. The described navigation paradigm was implemented in a tool suite called `FCA Tools Bundle` that is described in more detail in an additional paper [5].

In our future work, we plan to combine the reachability-based navigation with the membership-constraint-based navigation into a new improved paradigm in order to solve the problem of choosing a starting point. The reachability-based navigation can offer the visualization support, while the constraints added by the user can be taken into consideration by highlighting the concepts in the local visualization that satisfy the constraints.

References

1. Dragoş, S., Haliţă, D., Troancă, D.: Investigating trend-setters in e-learning systems using polyadic formal concept analysis and answer set programming. In: Proc. of AI4KM 2016, co-located with IJCAI. pp. 42–48 (2016)
2. Ganter, B., Stumme, G., Wille, R. (eds.): Formal Concept Analysis, Foundations and Applications, LNCS, vol. 3626. Springer (2005)
3. Ganter, B., Wille, R.: Formal Concept Analysis - Mathematical Foundations. Springer (1999)
4. Jäschke, R., Hotho, A., Schmitz, C., Ganter, B., Stumme, G.: TRIAS - An Algorithm for Mining Iceberg Tri-Lattices. In: Clifton, C.W., Zhong, N., Liu, J., Wah, B.W., Wu, X. (eds.) Proc. of ICDM 2006. pp. 907–911. IEEE (2006)
5. Kis, L.L., Săcărea, C., Troancă, D.: FCA Tools Bundle - a Tool that Enables Dyadic and Triadic Conceptual Navigation. In: Proc. of FCA4AI 2015, co-located with IJCAI 2015 (2015)
6. Lehmann, F., Wille, R.: A Triadic Approach to Formal Concept Analysis. In: Ellis, G., Levinson, R., Rich, W., Sowa, J.F. (eds.) Proc. of ICCS 1995. Lecture Notes in Computer Science, vol. 954, pp. 32–43. Springer (1995)
7. Rudolph, S., Săcărea, C., Troancă, D.: Membership constraints in formal concept analysis. In: Yang, Q., Wooldridge, M. (eds.) Proc. of IJCAI 2015. pp. 3186–3192. AAAI Press (2015)
8. Rudolph, S., Săcărea, C., Troancă, D.: Towards a navigation paradigm for triadic concepts. In: Baixeries, J., Săcărea, C., Ojeda-Aciego, M. (eds.) Proc. of ICFCA 2015. LNCS, vol. 9113, pp. 232–248. Springer (2015)
9. Rudolph, S., Săcărea, C., Troancă, D.: Conceptual navigation for polyadic formal concept analysis. In: Proc. of AI4KM 2016, co-located with IJCAI. pp. 35–41 (2016)
10. Warren, H.S.: A modification of Warshall's algorithm for the transitive closure of binary relations. Commun. ACM 18(4), 218–220 (1975)
11. Warshall, S.: A Theorem on Boolean Matrices. J. ACM 9(1), 11–12 (1962)

FCA Tools Bundle - a Tool that Enables Dyadic and Triadic Conceptual Navigation

Levente Lorand Kis, Christian Săcărea, and Diana Troancă

Babeş-Bolyai University Cluj-Napoca
kis_lori@yahoo.com, csacarea@cs.ubbcluj.ro, dianat@cs.ubbcluj.ro

Abstract. Formal Concept Analysis is a prominent field of applied mathematics handling collections of knowledge - formal concepts - which are derived from some basic data types, called formal contexts by using concept forming operators. One of the strengths of FCA is the elegant, intuitive and powerful graphical representation of landscapes of knowledge as concept lattices. Nevertheless, in case of triadic FCA (3FCA) for more than 20 years there was no automatic tool for graphical representation of triconcept sets. Moreover, the triangular representation of trilattices, used so far in 3FCA has several disadvantages. Besides the lack of clarity in representation, one major disadvantage is that not every trilattice has a triangular diagram representation. In this paper we focus on the problem of locally navigating in triconcept sets and propose a tool which implements this navigation paradigm. To the best of our knowledge this is the first tool which makes navigation in larger triconcepts sets possible, by flipping through a certain collection of concept lattices.

1 Introduction

The FCA community agrees upon the necessity of having powerful software tools for formal context handling and lattice representation, as well as for other FCA features which are close to the heart of everyone from this community. Moreover, these tools need to be also accessible also to users outside the FCA community and there is plenty of work which has been done in this area. Without being comprehensive, an overview of FCA software was compiled by Priss and can be found on her webpage[1]. We will not cite any of these tools, since citing all of them would expand the Bibliography section out of the scope of a workshop paper, and making a citation selection would mean to emphasize some tools and to disregard some others, which again is not the scope of this paper.

Nevertheless, we need to point out that all tools are handling the 'classical' dyadic case or some extensions like *patterns structures* or *relational FCA*, while the triadic case is neglected. We do not know any tool able to draw a trilattice diagram or to represent somehow triconcept sets. In our opinion, triadic FCA (3FCA) has a real potential for real life applications once the problem of knowledge representation has been solved in a satisfactory manner. Based on some

[1] http://www.upriss.org.uk/fca/fcasoftware.html

previous attempts of representing triconcept sets as graphs [9] or using a circular visualization tool [3], we focused on a local navigation paradigm for triconcept sets [8]. This paradigm is meant for exploring triconcept sets by choosing one triconcept (A_1, A_2, A_3) and locking one of its components (either the extent, or the intent, or the modus) and then using the dyadic projection $\mathbb{K}^{ij}_{Ak}$ of related *reachable* concepts. Brief details on this paradigm are included in Section 2.

In this paper, we present the current state of `FCA Tools Bundle`, a collection of tools for dyadic and triadic Formal Concept Analysis. The triadic part consists of the implementation of the above-mentioned local navigation paradigm, while the dyadic part contains lattice building and visualization tools. In Section 2 we briefly introduce some preliminaries regarding triadic contexts and the suggested navigation paradigm, while in Section 3 we motivate the implementation of the tool suite. Section 4 describes the features implemented by the tool, while Section 5 presents an example of navigating through a triadic dataset. In Section 6 we describe the architecture of the tool and the technologies as well as the external tools used in the implementation. Finally, Section 7 describes some of the future directions for the development of the tool.

2 Preliminaries

We focus in this section only on the local navigation paradigm, as it has been introduced in our previous paper [8]. For more, we refer to the standard literature [4, 7].

Let $\mathbb{K} = (K_1, K_2, K_3, Y)$ be a tricontext. There are two types of dyadic projections which can be obtained from $\mathbb{K}$. First, we can 'flatten' $\mathbb{K}$ by slicing it and putting all slices side by side: $\mathbb{K}^{(i)} := (K_i, K_j \times K_k, Y^{(i)})$, with $(a_i, (a_j, a_k)) \in Y^{(i)}$ if and only if $(a_i, a_j, a_k) \in Y$, where $a_i \in K_i, a_j \in K_j, a_k \in K_k$ and $\{i, j, k\} = \{1, 2, 3\}$ are pairwise different.

Another projection is given by 'locking' a subset $A_k \subseteq K_k$ and intersecting all those slices that correspond to elements of A_k. More specifically, for $\{i, j, k\} = \{1, 2, 3\}$ and $A_k \subseteq K_k$, we define $\mathbb{K}^{(ij)}_{A_k} := (K_i, K_j, Y^{(ij)}_{A_k})$, where $(a_i, a_j) \in Y^{(ij)}_{A_k}$ if and only if $(a_i, a_j, a_k) \in Y$ for all $a_k \in A_k$.

In our previous paper, we have studied some theoretical issues regarding triconcepts and these dyadic projections [8] . If (A_1, A_2, A_3) is a triconcept, then the dimensions $i \in \{1, 2, 3\}$ are called *i-perspectives*. Performing a dyadic projection after one of these three perspectives, one can observe that there is a one-to-one correspondence between the corresponding dyadic concepts and some triconcepts of the original tricontext $\mathbb{K}$. These triconcepts are called *directly reachable* using perspective i. The reachability relation is defined as the transitive closure of the direct reachability relation. This enables *local navigation* in triconcept sets, by iteratively choosing a perspective, then browsing the direct reachable triconcepts set by exploring the subsequent projected dyadic concept lattice and so on. This perspective-locking-and-unlocking procedure gives rise to interesting theoretical questions, partly solved in our previous paper [8]. Of course, the *local character*

of the navigation comes with the cost of loosing overview. Nevertheless, navigating along the reachability relation is synonym to flipping through several concept lattices and investigate connections of triconcept sets which are not visible from the 'classical' trilattice diagram representation. Section 4 describes the practical implementation of this local navigation paradigm in `FCA Tools Bundle`.

3 Motivation

The main motivation of developing this tool comes from practical applications of FCA. Some data have an inherent triadic structure, folksonomies being one of the prominent examples, and for these data one can generate either a complete list of triconcepts or one can restrict only to the most frequent [6]. But even data without inherent triadic structure might be investigated from a triadic point of view, which increases the popularity of 3FCA and makes it suitable for more applications. These data can usually be interpreted as many-valued contexts which are then scaled using the `ToscanaJ Suite` [2]. Using a locally developed plugin called `Toscana2Trias`, one can build a triadic view on the data directly from `ToscanaJ`. Triconcepts can be easily computed but without a visualization tool there is no navigation support and hence the applicability of 3FCA remains uncertain.

Making FCA more popular outside of its natural community is another goal which motivates this development. The paradigm introduced in our previous paper seems to be a good starting point, since it enables navigation in triconcept sets by using the natural elegance and expressiveness of concept lattices [8].

Once the paradigm has been set up, several problems had to be overcome. For instance, the well-known node overlapping, which requires manual rearrangement of nodes. For this purpose, we implement a concept lattice generation method that integrates a collision detection functionality. This feature will enable to automatically check when nodes in the diagram are overlapping and rearrange the diagram in such a way that there are no more overlappings, i.e. collisions among the nodes.

Usability and accessibility is another goal of this tool development. Hence it has been developed as an open source project. Furthermore, it is made public as a web site which requires registration, but no other installation process which makes it easy to use.

Last but not least, revival of software development for FCA is in our opinion, a major objective for our community, to which we strongly adhere.

4 `FCA Tools Bundle` - Description and Features

The `FCA Tools Bundle`[2] currently implements features for dyadic and triadic formal concept analysis. The main purpose of the tool is to enable the user to

[2] `https://fca-tools-bundle.com`

visualize formal contexts of different dimensions in several formats (concept list, concept lattice) and to interact with them.

The tool offers some public contexts for users who want to test the functionalities (see Figure 1) and, in addition, there is an import functionality that allows users to add other dyadic or triadic contexts. The allowed formats for importing a formal context are `cxt` and `csv`. The `cxt` format is the standard format for dyadic contexts, while for triadic context it is a straightforward extension of the format. When using the second format, the `csv` file must contain only the tuples of the relation from which the context can be reconstructed.

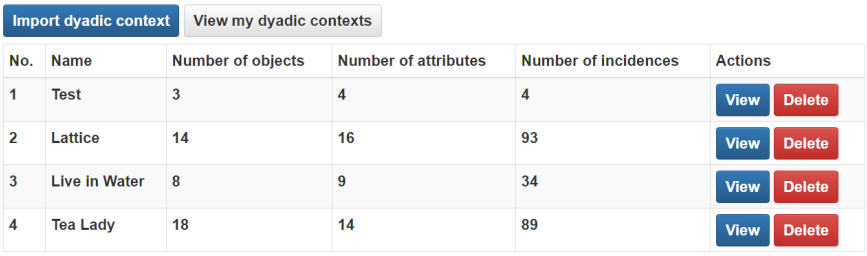


Fig. 1: List of dyadic contexts

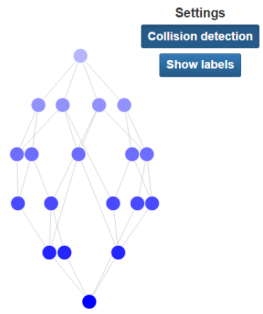


Fig. 2: Concept Lattice of a Dyadic Context

The key features offered by `FCA Tools Bundle` for a dyadic context are to visualize the details of the context, i.e. object set, attribute set, incidences and concept set and to compute and visualize the concept lattice of the context represented in Figure 2.

As mentioned previously, another useful feature offered by the tool is the possibility to avoid overlappings in the concept lattice using a collision detection functionality. After computing the concept lattice, the y coordinate of the nodes is locked, but they are allowed to move freely in their respective layer. Therefore, the user can move the nodes on the x-axis and rearrange them as they find suitable.

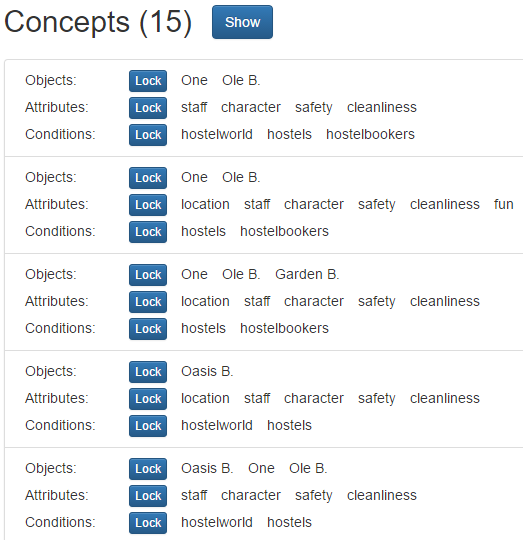


Fig. 3: Triconcepts of the hostels context

For a triadic context, one can first visualize the concept list. Figure 3 shows a list of triconcepts from the hostel example given by Glodeanu [5]. In addition, an important functionality, which was not implemented by any previous tool, is a local visualization of parts of the triadic context which enables the navigation paradigm in the triconcept set. One can choose a set of elements (of the same type, i.e. objects, attributes, or conditions) to project on and visualize the concept lattice of the dyadic projection. This functionality can be used for the navigation paradigm based on dyadic projections ([8]) and a proof of concept will be presented in Section 5.

5 Example

In a previous paper we defined a local navigation paradigm for triadic contexts based on appropriately defined dyadic projections. In this section we show how `FCA Tools Bundle` can be used to navigate in a triadic context using this paradigm. For this purpose we consider the hostels context, which was previously defined by Glodeanu [5]. Figure 4 depicts the layered cross-table representation of the hostel context.

As mentioned previously, some of the triconcepts of the hostel context are represented in Figure 3. We choose the triconcept

$T_1 = (\{One, OleB., GardenB.\}, \{location, staff, character, safety, cleanliness\}, \{hostels, hostelbookers\})$

as a starting point and lock the third perspective, i.e. the set of conditions $\{hostels, hostelbookers\}$. The corresponding dyadic projection is depicted in Fig-

hostel-world	character	safety	location	staff	fun	cleanliness
Nuevo S.			×			
Samay		×	×	×		×
Oasis B.	×	×	×	×		×
One	×	×		×		×
Ole B.	×	×		×		×
Garden B.			×	×		×

hostels	character	safety	location	staff	fun	cleanliness
Nuevo S.			×	×		
Samay		×	×	×		×
Oasis B.	×	×	×	×	×	×
One	×	×	×	×	×	×
Ole B.	×	×	×	×	×	×
Garden B.	×	×	×	×	×	×

hostel-bookers	character	safety	location	staff	fun	cleanliness
Nuevo S.			×	×		
Samay	×	×	×	×		×
Oasis B.		×	×	×	×	×
One	×	×	×	×	×	×
Ole B.	×	×	×	×	×	×
Garden B.	×	×	×	×		×

Fig. 4: Hostels tricontext

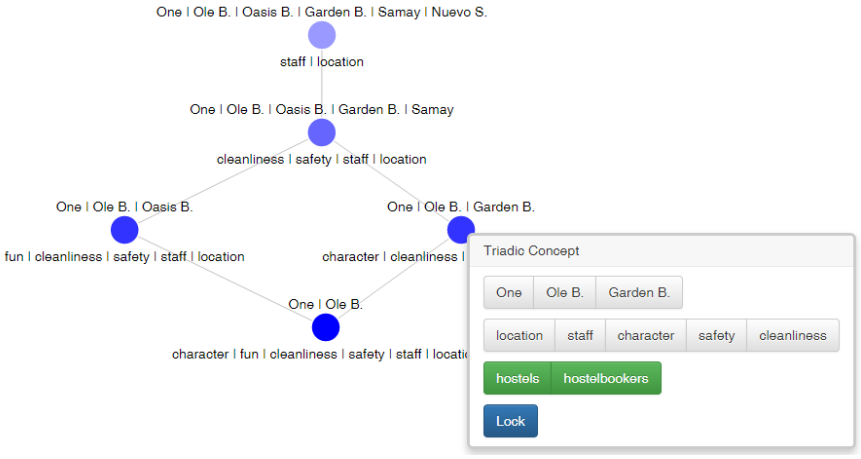


Fig. 5: Dyadic projection on the condition set $\{hostels, hostelbookers\}$

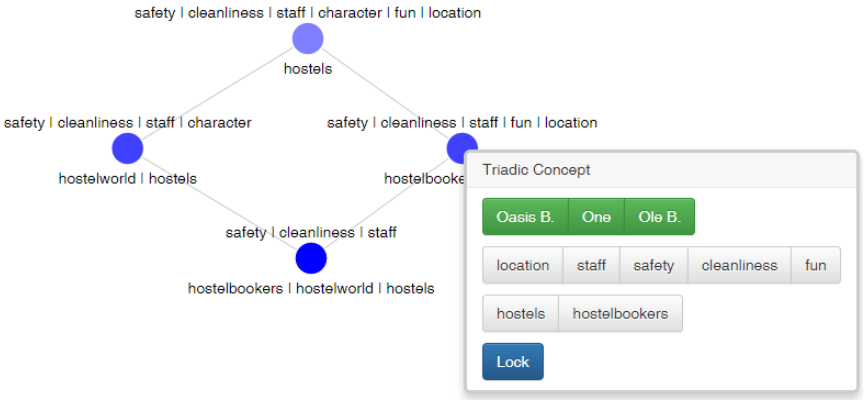


Fig. 6: Dyadic projection on the object set $\{OasisB., One, OleB.\}$

ure 5. By right clicking a dyadic concept in the projection the user can see the associated triadic concept. Here, the triconcept T_1 corresponds to the right-most dyadic concept. Herefrom, one can choose a different triconcept, for example the left-most concept

$T_2 = (\{One, OleB., OasisB.\}, \{fun, cleanliness, safety, staff, location\}$
$\{hostels, hostelbookers\})$,

and project on the object set. The corresponding concept lattice, after this second lock step, is depicted in Figure 6.

6 Architecture of `FCA Tools Bundle`

FCA Tools Bundle is a platform that intends to implement FCA features for the dyadic and triadic case. It has a log-in system which can offer a personalized experience to the user. Therefore, a formal context can be public, i.e. visible to every other user, or private, i.e. visible only to the user who defined it.

There are multiple technologies used in implementing FCA Tools Bundle as well as some external FCA tools that are integrated for using some of the previously implemented FCA algorithms. For computing dyadic concepts a slightly modified version of `InClose2` ([1]) is used. In the triadic case, the `Trias` algorithm is used for computing the triconcepts [6].

Considering that one of our motivations was to improve visualization methods for concept sets, special attention was given to the concept lattice representation. For this purpose, we used a force-directed approach that tries to position the nodes at somewhat equal distance and with as few intersections among the edges as possible. First, a layer is assigned to each node, which determines its final position on the y-axis of the lattice diagram, i.e. it cannot be changed by dragging the node up or down. For this purpose, we use a concept lattice drawer algorithm proposed by Roland Puntaier[3]. Afterwards the forces rearrange the nodes in the concept lattice as follows. Each node has a repulsion force that pushes other nodes away, while simultaneously each line between the nodes acts as a spring-like force and attracts pairs of nodes towards each other. This approach seems to produce one of the best outputs for the concept lattice, however, it can still be the case that there are overlappings of nodes. For that reason, we implement a custom collision detection algorithm adapted to the x-axis of the lattice representation, since rearranging the nodes on this axis will be sufficient for avoiding overlappings of the nodes. This algorithm follows the ideas described by Mike Bostock[4] and uses the quadtree structure implemented in the D3JS library.

7 Conclusions and Future Work

In this paper, we presented `FCA Tools Bundle`, a platform that offers, for now, features of visualization and navigation for dyadic and triadic FCA. We have improved concept lattices generation using a detection collision algorithm, in order

[3] `https://gist.github.com/rpuntaie/37a380a84f9843b5dd17`
[4] `https://bl.ocks.org/mbostock/3231298`

to avoid manually arranging the concept lattice for concept visibility. Moreover, we have shown how concept lattices can be used for a triadic navigation paradigm based on appropriately defined dyadic projections.

The development of the tool is ongoing and there are multiple functionalities that we plan to implement in the future, some of which are shortly described in this section. We intend to implement exploration procedures for dyadic as well as triadic FCA, based on implication computation. Moreover, temporal FCA is another branch that lacks user-friendly tools, hence implementing lifetracks and temporal views is another work which needs to be done [10]. Another important case that we plan to consider, since so far it was only implemented in the `ToscanaJ Suite`, are many-valued contexts. Furthermore, we will relate FCA to pattern structures and implement corresponding algorithms.

In conclusion, we believe that the presented version of `FCA Tools Bundle` brings an important contribution to the collection of FCA tools, by implementing functionalities of visualization and navigation in triadic concept sets, which, to the best of our knowledge, are not present in any other tool.

References

1. Andrews, S.: In-Close2, a High Performance Formal Concept Miner. In: Andrews, S., Polovina, S., Hill, R., Akhgar, B. (eds.) Proc. of ICCS 2011. LNCS, vol. 6828, pp. 50–62. Springer (2011)
2. Becker, P., Correia, J.H.: The ToscanaJ Suite for Implementing Conceptual Information Systems. In: Ganter, B., Stumme, G., Wille, R. (eds.) Formal Concept Analysis, Foundations and Applications. LNCS, vol. 3626, pp. 324–348. Springer (2005)
3. Dragoş, S., Haliţă, D., Săcărea, C., Troancă, D.: Applying Triadic FCA in Studying Web Usage Behaviors. In: Proc. of KSEM 2014, pp. 73–80. Springer (2014)
4. Ganter, B., Wille, R.: Formal Concept Analysis - Mathematical Foundations. Springer (1999)
5. Glodeanu, C.V.: Tri-ordinal Factor Analysis. In: Cellier, P., Distel, F., Ganter, B. (eds.) Proc. of ICFCA 2013. LNCS, vol. 7880, pp. 125–140. Springer (2013)
6. Jäschke, R., Hotho, A., Schmitz, C., Ganter, B., Stumme, G.: TRIAS - An Algorithm for Mining Iceberg Tri-Lattices. In: Clifton, C.W., Zhong, N., Liu, J., Wah, B.W., Wu, X. (eds.) Proc. of ICDM 2006. pp. 907–911. IEEE (2006)
7. Lehmann, F., Wille, R.: A Triadic Approach to Formal Concept Analysis. In: Ellis, G., Levinson, R., Rich, W., Sowa, J.F. (eds.) Proc. of ICCS 1995. LNCS, vol. 954, pp. 32–43. Springer (1995)
8. Rudolph, S., Săcărea, C., Troancă, D.: Towards a Navigation Paradigm for Triadic Concepts. In: Baixeries, J., Săcărea, C., Ojeda-Aciego, M. (eds.) Proc. of ICFCA 2015. LNCS, vol. 9113, pp. 232–248. Springer (2015)
9. Sacarea, C.: Investigating Oncological Databases Using Conceptual Landscapes. In: Hernandez, N., Jäschke, R., Croitoru, M. (eds.) Proc. of ICCS 2014. LNCS, vol. 8577, pp. 299–304. Springer (2014)
10. Wolff, K.E.: Temporal Concept Analysis. In: Proc. of ICCS 2001 - International Workshop on Concept Lattices-based KDD. pp. 91–107 (2001)

Steps Towards Interactive Formal Concept Analysis with LatViz

Mehwish Alam[1], Thi Nhu Nguyen Le[2], and Amedeo Napoli[2]

1. Laboratoire d'Informatique de Paris-Nord, Université Paris 13, Paris, France
2. LORIA (CNRS – Inria Nancy Grand Est – Université de Lorraine)
BP 239, Vandoeuvre-lès-Nancy, F-54506, France
{alam@lipn.univ-paris13.fr}{thi-nhu-nguyen.le,amedeo.napoli@loria.fr}

Abstract. With the increase in Web of Data (WOD) many new challenges regarding exploration, interaction, analysis and discovery have surfaced. One of the basic building blocks of data analysis is classification. Many studies have been conducted concerning Formal Concept Analysis (FCA) and its variants over WOD. But one fundamental question is, after these concept lattices are obtained on top of WOD, how the user can interactively explore and analyze this data through concept lattices. To achieve this goal, we introduce a new tool called `LatViz`, which implements several algorithms for constructing concept lattices and allows further navigation over lattice structure. `LatViz` proposes some remarkable improvements over existing tools and introduces various new functionalities such as interaction with expert, visualization of Pattern Structures, AOC posets, concept annotations, filtering concept lattice based on several criteria and finally, an intuitive visualization of implications. This way the user can effectively perform an interactive exploration over a concept lattice which is a basis for a strong user interaction with WOD for data analysis.

Keywords: Lattice Visualization, Interactive Exploration, Web of Data, Formal Concept Analysis.

1 Introduction

In the last decade, there has been a huge shift from the Web of Documents to the Web of Data (WOD). Web of Documents represents data in the form of HTML pages which are linked with other HTML pages through hyperlinks. This web of documents has evolved into WOD where all the information contained is represented in the form of entities and relations allowing the semantics to be embedded in the representation of these data. These data are in the form of a (node-arc) labeled graph belonging to several domains such as newspapers, publications, biomedical data etc. The growth in the publication of data sources in WOD has made it an important source of data, which has led towards many challenges pertaining to effective utilization of this data. WOD mainly represents data in the form of Resource Description Framework (RDF)[1]. There are several

[1] `http://www.w3.org/RDF/`

ways such as data dumps and SPARQL queries to access these data, which can be utilized for several applications, one of which is visualization and interactive exploration for data analysis purposes. Several visualization tools have been developed for this purpose, one of which is LODLive[2] [6], where the user can choose data-sets such as DBpedia and Freebase and specify an entity as a starting point for browsing the node-arc labeled graph. Another tool based on graphical display is RelFinder [15], where given several entities the tool automatically finds the paths connecting these entities. The major drawback of LODLive is that after two hops the number of nodes increase and it is hard to visualize the data. Moreover, these tools are good for getting an insight into what RDF graph contains but they are not built for the purpose of knowledge discovery.

In order to provide the user with the ability to perform data analysis and knowledge discovery over such kind of data, there is a need to perform classification, where the obtained classes are further made available to the user for exploration and subjective interpretation. In the current study we use Formal Concept Analysis as the basis for classification. Several studies have already been conducted using FCA and its variants over RDF graphs or its generalization to knowledge graphs. Out of these studies so far RV-Xplorer [3] is the only tool that actually allows interactive exploration of clustered RDF data [2]. The purpose of this paper is to enhance the functionalities discussed in the previous studies. In this study we introduce a new tool `LatViz` which increases the interpretability of a concept lattice by remarkably improving the user interaction with the concept lattice as compared to existing tools. Various new functionalities have been introduced such as the visualization of Pattern Structures and AOC-posets, concept annotation, filtering concept lattice and pattern concept lattice based on several criteria and finally, an intuitive visualization of implications. This way the user can effectively perform an interactive exploration over a concept lattice which in turn gives a basis for a strong user interaction with WOD for knowledge discovery purposes. In this paper, we detail the important interaction operations implemented in `LatViz`. In the rest of this paper we refer to "user" as an "expert" as (s)he having basic knowledge about the lattice structure.

The paper is structured as follows: Section 2 introduces a motivating example. Section 3 introduces some of the important functionalities of `LatViz`, while in Section 4, we discuss some of the related tools already developed and finally Section 5 details the future perspectives of the current work.

2 Motivating Example

Let us consider that an expert is searching for papers published by a particular team in conferences or journals related to his/her field of research. In order to locate the papers of his/her interest (s)he has to search for specific keywords or authors in the local portal. For getting the view of which kind of papers are contained (s)he has to run a broad query and then narrow down his/her

[2] `http://en.lodlive.it/`

query to obtain papers on specific keywords or authors or group of keywords or authors. The expert will end up running several queries to get what (s)he wants. Moreover, if the expert wants to know the collaborations of the team with other members of the research community outside the team, as well as the diversity and the specialization of the team members, this cannot be directly obtained by simple querying. To obtain such kind of knowledge there is a need to introduce a support for data analysis. Based on this scenario, we show how the expert can be guided thanks to an adapted visualization tool to obtain such information of interest with the help of concept lattices.

3 `LatViz` for Interactive Exploration of Concept Lattices

3.1 User Interface

The display of `LatViz` resembles Conexp[3], which provides basic functionalities for building a concept lattice. `LatViz` implements two algorithms for building a concept lattice from a binary context, one of which is introduced in [14]. Another, efficient algorithm for building a concept lattice is AddIntent [20]. A demo of LatViz is available on-line through `http://latviz.loria.fr/latviz/`.

The concept lattice for the scenario in section 2 was created by mapping the RDF data to a formal context $\mathcal{K} = (G, M, I)$. The subjects of the triples were considered as the set of objects G, the objects in the RDF triples i.e., keywords and authors were considered as the set of attributes M. In this example, the RDF triples were created from the publications of the Knowledge Discovery (KDD) team in LORIA. The number of objects in the context are 343 and attributes are 1516. Very often huge concept lattices are obtained based on the context size. `LatViz` provides several interactive operations allowing for reduction of exploration space of the expert. To-date this is the most interactive tool having many unique functionalities such as handling numeric data with the help of interval pattern structures, AOC-posets, filtering concept lattice and implications which provides support for data analysis. Other functionalities such as annotating the lattice, level-wise display of a concept lattice etc. are discussed in many contexts but are not yet directly implemented in the commonly used tools. In the following we detail each of these functionalities for data analysis.

3.2 AOC-Posets

AOC-poset is a partially ordered set of the attribute and object concepts, first introduced in [18, 19]. The object and attribute concept are referred to as *introducers* in [5]. Once an attribute is introduced in a concept it is inherited from top to bottom while, dually, an introduced object is "inherited" from bottom to top. During this study, we implement the Hermes Algorithm introduced in [5] for building AOC-Poset from binary context. Figure 1 show the highlighted AOC-Posets of the concept lattice built for the running scenario.

[3] `http://conexp.sourceforge.net/`

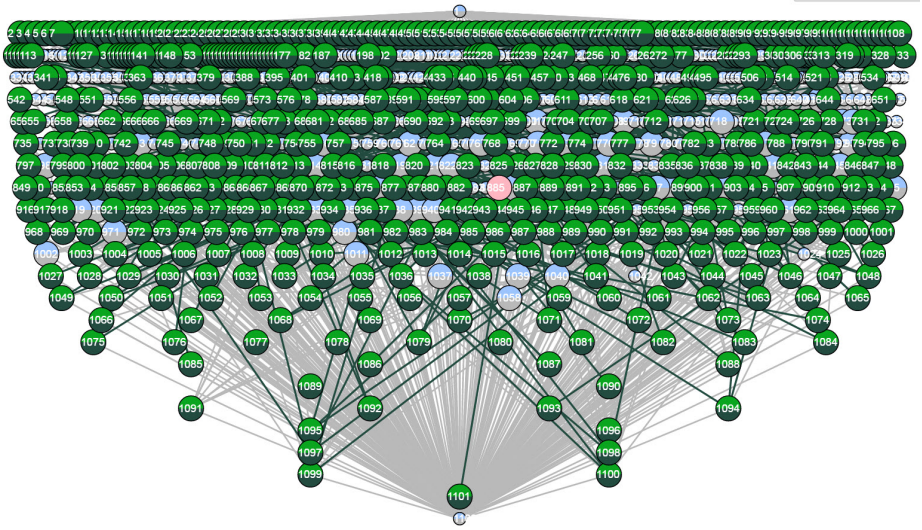


Fig. 1: AOC-Posets.

3.3 Displaying Concept Lattice Level-wise

AOC-Posets actually reduce exploration space but still a huge number of concepts remain to be observed. `LatViz` allows the creation of concept lattice level-wise by interaction. When an expert clicks on the top concept, `LatViz` computes and displays the first level. After that the expert can select the concept for continuing the exploration, then `LatViz` computes the next level for that concept. In Figure 2, the top image shows the first level of the concept lattice built by selecting the top concept. Then the expert can view the contents of each concept on mouseover. In the running example, expert locates the concept with all the papers of Amedeo Napoli (i.e., K#2), which shows that the total number of documents written by Amedeo Napoli are 152. On selecting this concept, the next level of the lattice originating from the selected concept is computed (shown in the bottom image in Figure 2).

3.4 Display Sub/Super-Concepts of a Concept

In case of huge concept lattices sometimes it is hard to keep track of the ordering relations between the concepts. `LatViz` allows the expert to only highlight sub/super-concepts of a concept. For example, if the expert wants to display all the publications along with the collaborations of the author Amedeo Napoli, (s)he can highlight the associated sub-lattice of the attribute concept of "Amedeo Napoli". Figure 3 shows the highlighted sub-lattice in brown. An expert can highlight the super-concepts connected to a concept. If the expert is looking for all the papers having some keywords common with the paper *Knowledge Organization and Information Retrieval Using Galois Lattices* having one or more of the

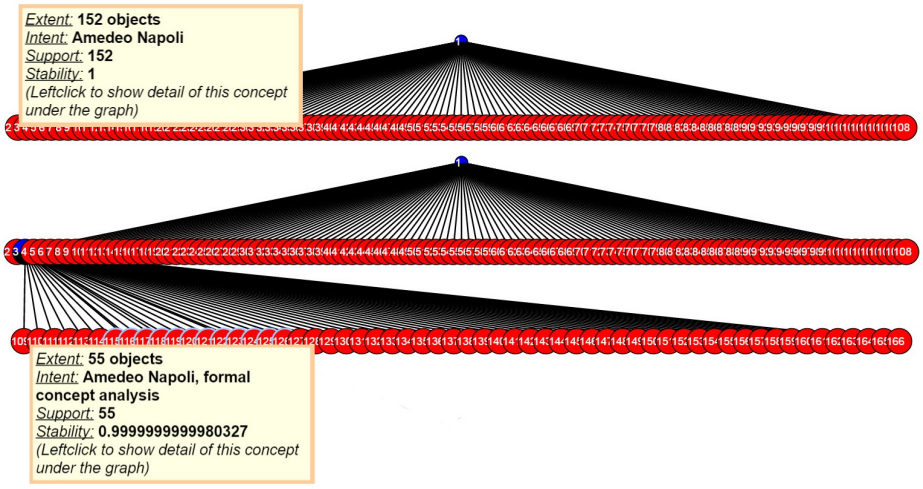


Fig. 2: Top image shows the first level of the concept lattice, the bottom image shows the second level built by interaction.

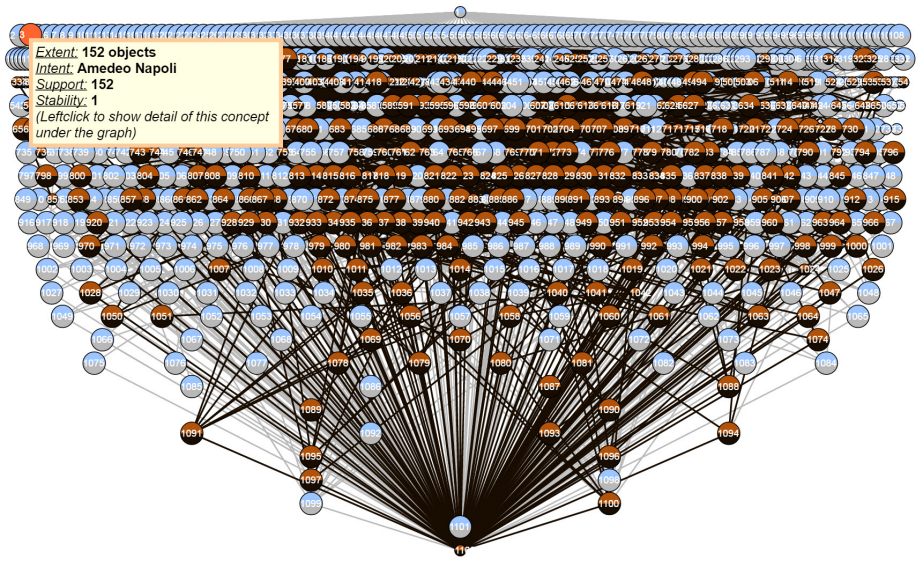


Fig. 3: The sub-lattice highlighted for the author "Amedeo Napoli".

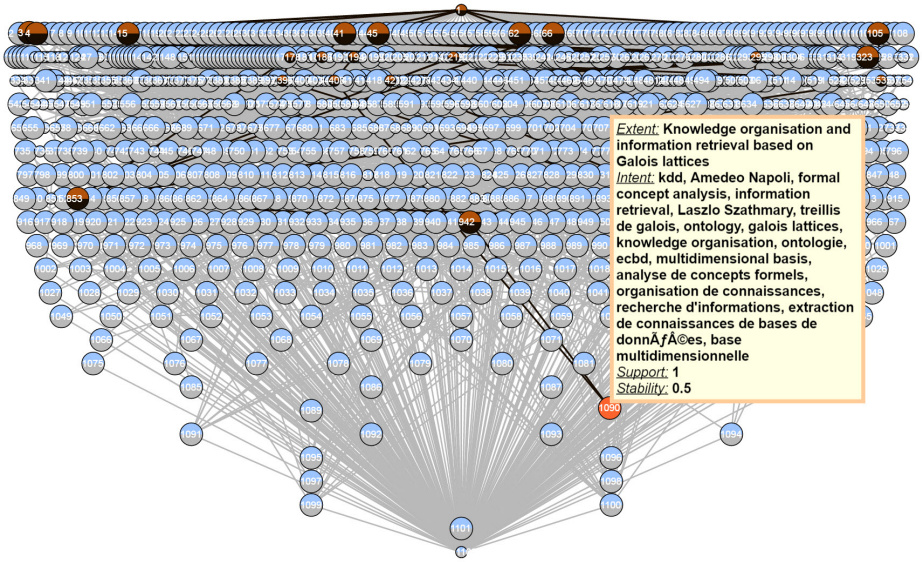


Fig. 4: Highlighting super-concepts of a concept.

keywords in the intent of the concept then (s)he can highlight the sub-lattice of super-concepts associated to it (see Figure 4).

3.5 Display/Hide the Sub-lattice

This functionality was partially implemented in RV-Xplorer [3] to reduce the interaction space of the expert. Here the expert can only show the part of the concept lattice in which (s)he is interested. The expert can locate the interesting concept by navigation, containing some intent or extent. If an intent is interesting and the expert marks the concept as interesting then only the sub-concepts of this concept are shown to the expert as the intents are inherited from top to bottom. Dually, if an extent is interesting for the expert then all the super-concepts are shown to the expert as the extent is inherited bottom-top. Previously, the expert highlighted the sub-lattice of the concept containing all the papers of Amedeo Napoli, now if the expert is interested in only the papers of Amedeo Napoli on Knowledge Representation then (s)he can navigate downwards and only see this part of concept lattice by marking it interesting (see Figure 5). Similarly, we previously highlighted all the super-concepts of the concept having the paper entitled *Knowledge Organization and Information Retrieval Using Galois Lattices* in the extent, Figure 6 only shows the associated sub-lattice to have a clearer view (see Figure 6).

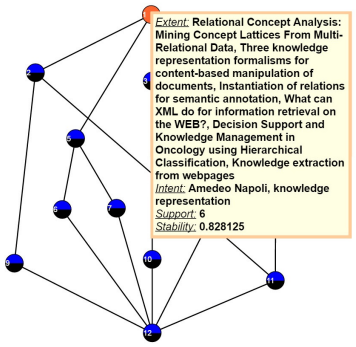


Fig. 5: Showing only sub-lattice of the interesting concept.

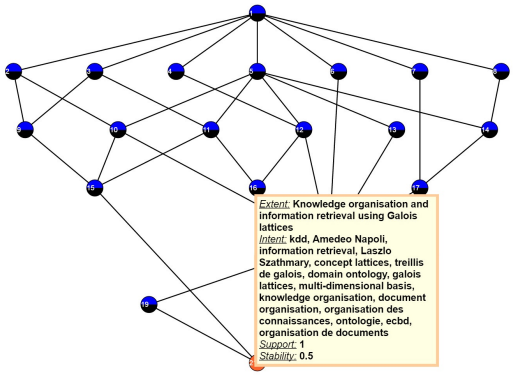


Fig. 6: Showing only super-concepts of the interesting concept.

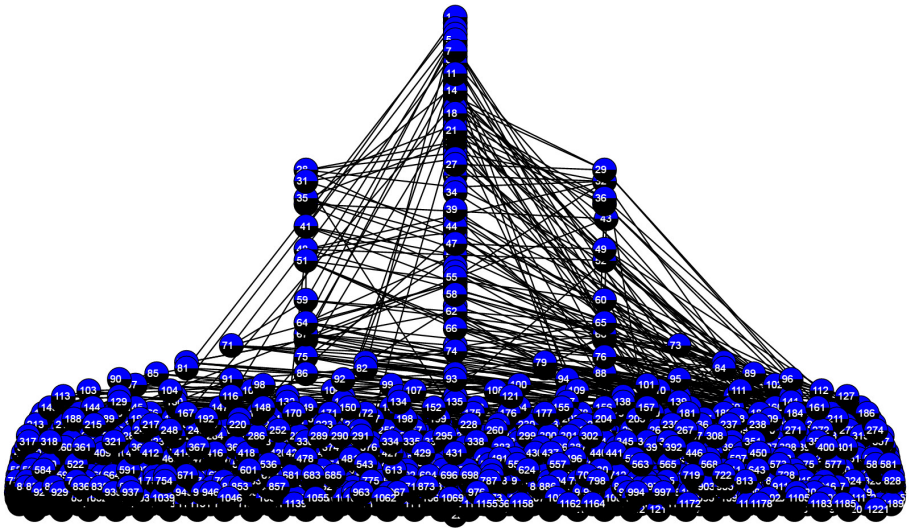


Fig. 7: Interval pattern concept lattice for publications.

3.6 Interval Pattern Structures

Interval Pattern Structures were first introduced in [16] for dealing with numerical data. Consider two descriptions $\delta(g_1) = \langle [l_i^1, r_i^1] \rangle$ and $\delta(g_2) = \langle [l_i^2, r_i^2] \rangle$, with $i \in [1..n]$ where n is the number of intervals used for the description of entities. The similarity operation $\sqcap$ and the associated subsumption relation $\sqsubseteq$ between descriptions are defined as the convex hull of two descriptions as follows: $\delta(g_1) \sqcap \delta(g_2) = \langle [min(l_i^1, l_i^2), max(r_i^1, r_i^2)] \rangle$. Based on this similarity measure interval pattern concept lattice can be built. In the running scenario, three numerical attributes for the papers were used i.e., year of publications, rank of the conference and the number of pages. The ranks of the conferences were con-

sidered based on COmputing Research and Education (CORE) rankings[4]. The ranks were A*, A, B, C and other which were coded as 1, 2, 3, 4 and 5 respectively. The final concept lattice generated for the last five years of publications of Knowledge Discovery Team is shown in Figure 7.

3.7 Lattice Filtering Criteria

There are two categories of filtering provided by `LatViz`; one is for the concept lattice created with the binary data and the other one is provided for the pattern concept lattice built with the help of interval pattern structures.

Filtering Concept Lattice. After a concept lattice is built by applying FCA, expert is allowed to set several filtering criteria such as stability, lift, extent size, intent size and finally specific object or attribute names. Let us consider that in the running example, the expert is looking for the papers published by Amedeo Napoli on the topic of pattern structures and FCA. A filter on the number of attributes in the intent is set to 3. The filtered concept lattice obtained over the complete lattice is shown in Figure 8. It further shows the authors with who Amedeo Napoli has worked i.e., Sergei O. Kuznetsov and Mehdi Kaytoue. This part of concept lattice shows the community of authors working with Amedeo Napoli on the topic of pattern structures.

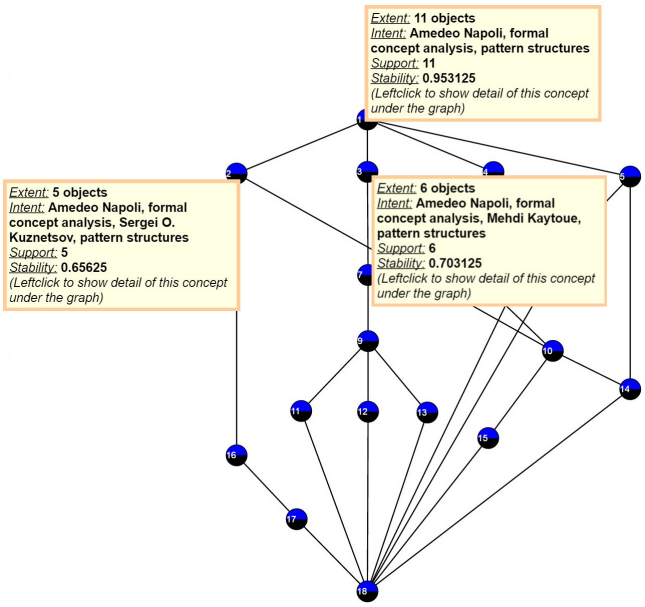


Fig. 8: Filtered concept lattice obtained from binary context.

[4] http://portal.core.edu.au/conf-ranks/

Filtering Pattern Concept Lattice. Interval Pattern Concept Lattices can also be filtered by specifying the number of attributes to be considered, the upper and the lower limits for the intervals in the intent of each attribute along with stability, lift and extent size. Let us consider the pattern concept lattice in Figure 7 which is hard to interpret. To make it more readable based on what an expert wants, (s)he is allowed to specify filters. For example, if the expert is looking for a paper published in a conference of a rank 1-4 in the year 2012 - 2015 and has the number of pages not less than 2 and no more than 42 then the respective filters can be set for the values of all three attributes. The filtered pattern concept lattice will then only contain the part of lattice needed by the user. Figure 7 shows the concept containing group of papers published from 2014-2015 in conferences with rank 2 having number of pages 2-42.

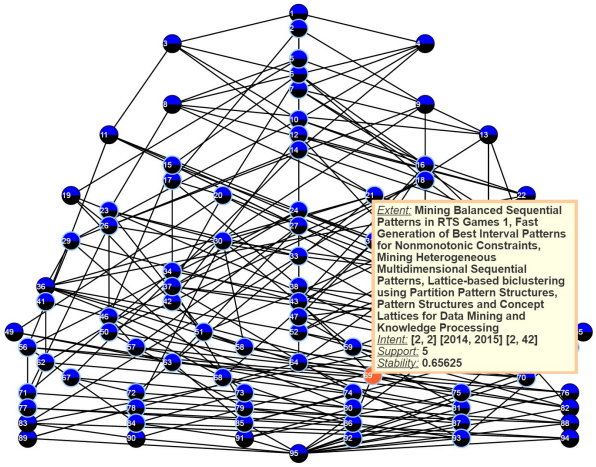


Fig. 9: Filtered Pattern Concept Lattice.

3.8 Attribute Implications

One of the many proposed visualization techniques for implications includes *table-based views*. The columns in the table represent rule ID, LHS and RHS of the rule, support and confidence measures. These views were used because of the simplicity of storage. However, as the number of rules can be too many it is not very evident for the expert to focus on interesting rules at a simple glance. Another way of visualizing association rules are *Matrix Views*, where rows represent the LHS and columns represent the RHS of the rules. Support and confidence are displayed by different colors in the intersection of the LHS and RHS. In case of a formal context, the number of objects/attributes can be very big leading to problems in displaying the matrix. By carefully taking into account the above drawbacks, we finally settle on visualizing implications with the help of scatter plots, where the x-axis shows the increasing support

and the y-axis shows the increasing lift (as we are considering implications the confidence of the rule is always 100%). Such kind of display helps the expert to single-out the rules (s)he wants to visualize based on the values of support and lift. Figure 10 shows implications of the running example, x-axis keeps the support in percentage and y-axis keeps lift. The number on top of the circle shows the number of rules existing in the same point in the plot. On mouse over, expert can view the implications.

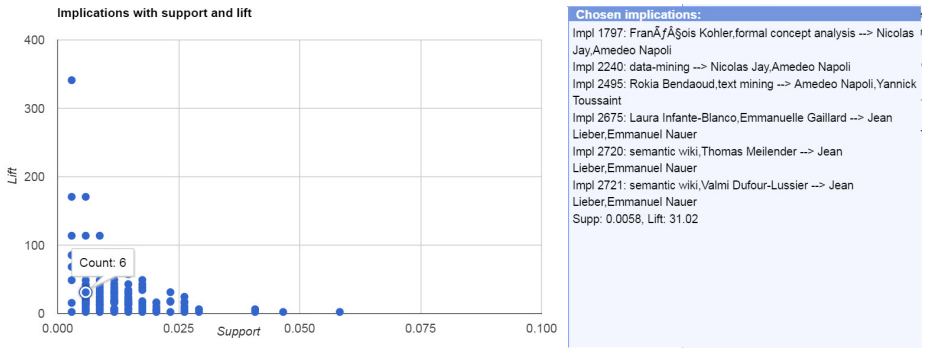


Fig. 10: Attribute implications for the running example.

4 Related Tools

In [2], the authors focus mainly on interactive data exploration over RDF data for interactive knowledge discovery. It clusters RDF triples based on RDF Schema and then allows interactive exploration with the help of RV-Xplorer (Rdf View eXplorer) [3]. It is a tool for visualizing views over RDF graphs mainly for identifying interesting parts of data and allow data analysis. It has also been extended for clustering SPARQL query answers. To-date there have been many other tools developed for reducing the effort of expert in observing and interpreting a concept lattice. Many of the tools have been developed for more specific purposes. CREDO [8] and FooCA [17] are the Web Clustering Engines [7] which take the answers from queries posed against search engines and create a concept lattice which is then displayed to the expert for interaction. CREDO allows only limited interaction, however, FooCA allows the expert to edit the context and iteratively build the concept lattice. CEM [10] is an email manager which allows quick search through the e-mails and usually deals with smaller concept lattices. Camelis [11] is a system based on FCA for the organization of documents allowing several navigation operations. Another set of tools such as Sewelis [13] and Sparklis [12] allows navigation/interaction over knowledge graphs. Many other tools such as Galicia[5], ConExp and ToscanaJ[6] are developed for academic purposes. `LatViz` takes the basic functionalities of ConExp and takes it to the another level by

[5] `https://sourceforge.net/projects/galicia/`
[6] `http://toscanaj.sourceforge.net/`

providing visualization for many algorithms introduced over time to increase the readability. Moreover, it re-uses the source-code for building concept lattice with the help of the algorithm in [14] from ToscanaJ [4]. It can not only be applied to WOD but it has been extended for interpreting any kind of data.

5 Discussion and Future Improvements

`LatViz` is a tool built for allowing expert interaction for data analysis purposes. It provides many new functionalities for reducing the exploration space of the expert and enable him to interpret the results. As a future perspective, we also want to implement other variations of pattern structures such as Pattern Structures introduced for structured set of attributes discussed in [1] and Heterogeneous Pattern Structures [9]. We also want to extend the implementation of implications to association rules. Finally, we also want to take into account matrix factorization.

References

1. Mehwish Alam, Aleksey Buzmakov, Amedeo Napoli, and Alibek Sailanbayev. Revisiting pattern structures for structured attribute sets. In *Proceedings of the Twelfth International Conference on Concept Lattices and Their Applications.*, pages 241–252, 2015.
2. Mehwish Alam and Amedeo Napoli. Interactive exploration over RDF data using formal concept analysis. In *2015 IEEE International Conference on Data Science and Advanced Analytics, DSAA.*, pages 1–10, 2015.
3. Mehwish Alam, Matthieu Osmuk, and Amedeo Napoli. RV-Xplorer: A way to navigate lattice-based views over RDF graphs. In *Proceedings of the Twelfth International Conference on Concept Lattices and Their Applications.*, pages 23–34, 2015.
4. Peter Becker and Joachim Hereth Correia. The ToscanaJ suite for implementing conceptual information systems. In *Formal Concept Analysis, Foundations and Applications*, pages 324–348, 2005.
5. Anne Berry, Alain Gutierrez, Marianne Huchard, Amedeo Napoli, and Alain Sigayret. Hermes: a simple and efficient algorithm for building the aoc-poset of a binary relation. *Ann. Math. Artif. Intell.*, 72(1-2):45–71, 2014.
6. Diego Valerio Camarda, Silvia Mazzini, and Alessandro Antonuccio. Lodlive, exploring the web of data. In *I-SEMANTICS 2012 - 8th International Conference on Semantic Systems, I-SEMANTICS '12, Graz, Austria, September 5-7, 2012*, 2012.
7. Claudio Carpineto, Stanislaw Osiński, Giovanni Romano, and Dawid Weiss. A survey of web clustering engines. *ACM Comput. Surv.*, 41(3):17:1–17:38, 2009.
8. Claudio Carpineto and Giovanni Romano. Exploiting the potential of concept lattices for information retrieval with CREDO. *J. UCS*, 10(8):985–1013, 2004.
9. Víctor Codocedo and Amedeo Napoli. A proposition for combining pattern structures and relational concept analysis. In *Formal Concept Analysis - 12th International Conference, ICFCA 2014, Proceedings*, pages 96–111, 2014.
10. Richard Cole and Gerd Stumme. CEM - A conceptual email manager. In *8th International Conference on Conceptual Structures, ICCS, Proceedings*, 2000.

11. Sébastien Ferré. Camelis: a logical information system to organise and browse a collection of documents. *Int. J. General Systems*, 38(4):379–403, 2009.
12. Sébastien Ferré. SPARKLIS: a SPARQL endpoint explorer for expressive question answering. In *Proceedings of the Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC.*, 2014.
13. Sébastien Ferré and Alice Hermann. Reconciling faceted search and query languages for the semantic web. *IJMSO*, 7(1):37–54, 2012.
14. Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin/Heidelberg, 1999.
15. Philipp Heim, Steffen Lohmann, and Timo Stegemann. Interactive relationship discovery via the semantic web. In *The Semantic Web: Research and Applications, 7th Extended Semantic Web Conference, ESWC 2010, Heraklion, Crete, Greece, May 30 - June 3, 2010, Proceedings, Part I*, pages 303–317, 2010.
16. Mehdi Kaytoue, Sergei O. Kuznetsov, and Amedeo Napoli. Revisiting numerical pattern mining with formal concept analysis. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence.*, pages 1342–1347, 2011.
17. Bjoern Koester. Conceptual knowledge retrieval with FooCA: Improving web search engine results with contexts and concept hierarchies. In *Advances in Data Mining, Applications in Medicine, Web Mining, Marketing, Image and Signal Mining, 6th Industrial Conference on Data Mining, ICDM Proceedings*, 2006.
18. R. Osswald and W. Petersen. A logical approach to data-driven classification. In *KI*, volume 2821 of *Lecture Notes in Computer Science*. Springer, 2003.
19. Wiebke Petersen. A set-theoretical approach for the induction of inheritance hierarchies. *Electr. Notes Theor. Comput. Sci.*, 53:296–308, 2001.
20. Dean van der Merwe, Sergei A. Obiedkov, and Derrick G. Kourie. Addintent: A new incremental algorithm for constructing concept lattices. In *ICFCA*, pages 372–385, 2004.

Linked Data Querying through FCA-based Schema Indexing

Dominik Brosius and Steffen Staab

Institute for Web Science and Technologies, University of Koblenz-Landau,
{dbrosius, staab}@uni-koblenz.de

Abstract. *The efficiency of SPARQL query evaluation against Linked Open Data may benefit from schema-based indexing. However, many data items come with incomplete schema information or lack schema descriptions entirely. In this position paper, we outline an approach to an indexing of linked data graphs based on schemata induced through Formal Concept Analysis. We show how to map queries onto RDF graphs based on such derived schema information. We sketch next steps for realizing and optimizing the suggested approach.*

Keywords: Linked Data, Formal Concept Analysis, Schema Indexing

1 Introduction

The ease of consuming Linked Open Data (LOD) depends on the availability of schema information. This holds for tasks such as browsing, where a user may not know the structure of the data found in a dataset, or querying, where schema information could be used for query optimization. In the LOD setting a query can be evaluated against the original LOD datasets through query federation or by evaluating it centrally against a LOD crawl. In both scenarios, the efficiency of query evaluation may be improved by identifying those datasets schematically matching the query – maybe partially – while leaving out others.

In the LOD cloud schema information is sparse, as many data items come with incomplete schema information or lack schema descriptions entirely. Methods for schema induction and ontology learning have been studied to remedy this problem ([4], cf. section 2). A method, that has successfully been used in this context, is Formal Concept Analysis (FCA) (cf. [2], [3], [12]).

To our knowledge, FCA has not been used for constructing indices, that allow a schema-oriented query-to-graph mapping. Particularly, we are not aware of previous work on mapping queries to formal contexts. We expect that FCA will benefit this kind of indexing through the formally sound construction of schemata that are free of external heuristics and concise at the same time. In this paper, we outline an approach for such a schema index and point to future work on an implementation.

2 Related Work

For our approach to schema indexing, we propose a property-based schema induction using FCA. Property-based type clustering is discussed in literature as a remedy for the schema sparsity in LOD datasets. The feasibility of such approaches has been analyzed in [5]. The authors show that the properties of resources within LOD datasets can be used for deducing resource types. [10] argue for a property-based data access for programming with Linked Data in order to deal with a lack of type descriptions. [6] further corroborate this position, as they argue for property-based concept definitions. In order to ensure quality of such definitions, they propose a system for incorporating interactive user feedback.

Schema induction, the learning of schema information from data, is a way to handle schema sparsity. [11] describes a statistical approach to schema induction through association rule mining on transaction tables. A recent approach is presented in [7]. It combines the mining of property-based entity descriptions and type clustering over these using DBSCAN. An overview over the field of schema/ontology learning is given in [4].

FCA has been applied to problems related to ontology learning. In [3] it has been used for learning taxonomies from natural language text. The authors of [2] and [12] use attribute exploration from FCA for semi-automatic approaches to ontology completion. Lately, [1] used FCA-based association rule mining for completing type definitions given in DBPedia data through further implied information.

We combine the induction of schema with building and providing an index for query-to-graph mapping. An index for subgraph querying is presented in [14]. There the authors make use of a precomputed lattice of subgraphs in order to narrow down the candidates for subgraph queries. An approach similar to ours is demonstrated in [8]. The authors present a schema-based index that is consisting of three layers, each supporting different types of queries. The index is constructed through clustering of RDF type information in a stream-based fashion. The approaches presented in [7], [8] and [12] either introduce heuristics or human oversight or require case specific parameter tuning. We sketch a FCA-based schema index that is free of such external factors and general enough to map the diverse data found in the LOD cloud.

3 Preliminaries

In the following we give a short introduction to Formal Concept Analysis (FCA), a method for conceptual knowledge discovery and representation, as well as the basics of RDF and SPARQL.

We lend the following definitions 1, 2, 3 from [13]:

Definition 1 (Formal Context). *A formal context* $\mathbb{K} \coloneqq (G, M, I)$ *is a triple comprised of a set of objects* G*, a set of attributes* M *and an indicidence relation* $I \subseteq G \times M$ *encoding that "*g *has attribute* m*" iff* gIm*.*

Definition 2 (Formal Concept). *For $A \subseteq G$ and $B \subseteq M$ [13] define $A' := \{m \in M | \forall g \in A : gIm\}$, $B' := \{g \in G | \forall m \in B : gIm\}$.*

A formal concept is a pair (A, B) with $A' = B$ and $B' = A$. A is the extent, B is the intent of the concept. As in [13], we abbreviate the set of all formal concepts for a formal context as $\mathcal{L}(G, M, I)$.

Formal concepts fall naturally into a hierarchy, called a *concept lattice*, by a subconcept-superconcept-relation defined via the subset relations over G and M, respectively.

Definition 3 ($\leq$). *For two concepts (A_1, B_1), (A_2, B_2) [13] define $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2$ (equivalently, $\Leftrightarrow B_1 \supseteq B_2$). The pair $(\mathcal{L}(G, M, I), \leq)$ is called concept lattice.*

RDF Datasources of the LOD cloud contain RDF[1] data. For now, we abstract from the possibility of blank nodes in RDF data.

Definition 4 (RDF Graph). *Given the set of RDF resource identifiers U and the set of literals L and having the set of RDF terms $T := U \cup L$: An RDF graph is a set of RDF triples $D \subseteq \{<s, p, o> | s \in U, p \in U, o \in T\}$. We further define the set of all known RDF graphs $\mathcal{D} := \{D | D \text{ is a known graph}\}$.*

SPARQL A language for formulating graph oriented queries against RDF datasets is SPARQL[2]. For this paper, we will focus on Basic Graph Patterns that are used for formalizing graph pattern matching and, thus, are fundamental to SPARQL(cf. [9]). Here, we further assume that queries are only evaluated against the default graph of a datasource.

Definition 5 (Basic Graph Pattern). *Given a set V of variable names, a Triple Pattern tp is a triple $(s, p, o) \in (U \cup V) \times (U \cup V) \times (T \cup V)$. A Basic Graph Pattern (BGP) of a query q is a set containing a number of Triple Patterns. We abbreviate the set of all queries only containing a single BGP as BGP.*

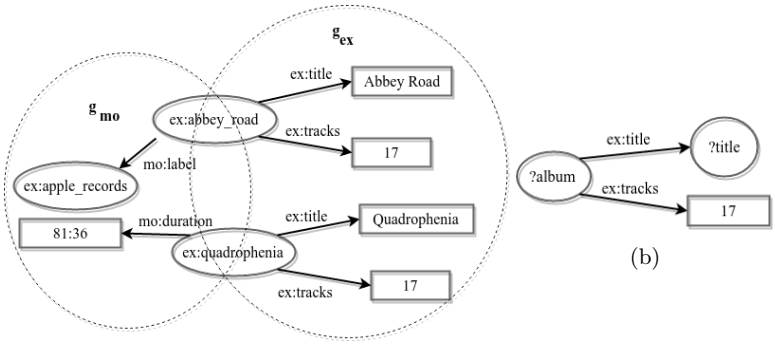
With this restriction to single-BGP queries, we define the solution to a query q through matching its BGP against the RDF graph D as $q(D)$ (cf. [9]). We further define a solution to q against a set of graphs $\mathcal{D}$ as $q(\mathcal{D}) := q(\bigcup_{D \in \mathcal{D}} D)$.

An example for a BGP and a possible solution is given in fig. 1 b), c).

4 Schema Index

In a query federation system a central query processor accepts and processes queries by dispatching (parts of) the queries to datasets (i.e., computing nodes serving them). In the case of LOD these datasets are graphs. The schema index maps queries onto minimal sets of graphs required for evaluating the given

[1] Resource Description Framework; `http://www.w3.org/TR/rdf-concepts/`
[2] `http://www.w3.org/TR/sparql11-overview/`



album	title
ex:abbey:road	"Abbey Road"
ex:quadrophenia	"Quadrophenia"
...	...

(c)

Fig. 1: Examples of (a) RDF graphs, (b) a BGP and (c) a solution to the BGP

queries. The optimal schema index minimizes the subset of all known RDF graphs to be used for returning the complete solution of a query q:

$$\mathcal{D}_i(q) = \text{argmin}_{\mathcal{D}_j \subseteq \mathcal{D} \wedge q(\mathcal{D}_j) = q(\mathcal{D})} |\mathcal{D}_j|. \tag{1}$$

The improvement of efficiency of query evaluation then depends on the number of graphs left out of $\mathcal{D}_i(q)$. Judging a graph to be relevant for query evaluation may be based on a schematic match of a query and the graph.

In our approach the matching will be informed by an underlying lattice of property type clusters induced from the graph data via FCA. With our approach, the schema index constructs and maintains a schema lattice covering every graph encountered. We build this lattice by applying FCA on a formal context extracted from an RDF graph.

Definition 6 (Resource-Feature Map). *We define a feature domain F as the set of all known predicates p encountered in $\mathcal{D}$. A resource-feature map $f : \mathcal{D} \times U \to 2^F$ assigns an RDF resource r the set of describing features found in a graph: $f(D, r) = \{p \in F | \exists o \in T :< r, p, o >\in D\}$.*

Example 1. $f(\texttt{g}_{\texttt{ex}}, \texttt{ex:quadrophenia}) = \{\texttt{ex:title}, \texttt{ex:tracks}\}$

There are other possible manifestations of such a resource-feature map. For example, one could (also) map the types assigned to resources through `rdf:type`.

Definition 7 (Schema Lattice). *We define a graph-covering formal context $\mathbb{K} := (U, F, I_f)$ with the set of resources U, the feature domain F and $I_f :=$*

$\{(r,p)|\exists D \in \mathcal{D}, \exists r \in U : p \in f(D,r)\}$. *The schema lattice* $\mathcal{L}_S = (\mathcal{L}(U,F,I_f),\leq)$ *is the lattice constructed from* $\mathbb{K}$ *via FCA.*

While extracting features given in the graphs, the schema index will also store the set of graphs contributing individual features to the subsequent lattice construction.

Definition 8 (Feature-contributing Graph). *For an RDF resource* r *and a feature* $p \in F$*, we define a set of feature-contributing graphs as* $\gamma(r,p) = \{D|p \in f(D,r)\}$*. For the features of an intent* B*, we define the set of contributing graphs as* $\gamma'(B) = \{D|\exists p \in B, \exists r \in U : D \in \gamma(r,p)\}$.

We conceive the schema index as a pair $(\mathcal{L}_S, \Gamma)$ of the schema lattice $\mathcal{L}_S$ and a function Γ mapping queries to indexed graphs. In order to look up graphs that a query should be evaluated against, the schema index then derives the queries type information through the *Query-Type Map* function:

Definition 9 (Query-Type Map). *The Query-Type Map* Θ *is a function* Θ: $BGP \rightarrow 2^F$ *that maps a query* q *to a set of intents* $\Theta(q)$.

For a query q the schema index returns the set of schematically appropriate graphs as follows:

$$\Gamma(q) = \{D|D \in \gamma'(B) \text{ for some } B \in \Theta(q)\}.$$

For our running example (cf. fig. 1), the graph g_{ex} would be returned for the BGP, as here {`ex:title`, `ex:tracks`} constitutes an intent reflected in the query.

5 Outlook and Conclusion

In this position paper, we sketch the idea of a schema index for a query-to-graph mapping. For constructing the index we aim at an automatic schema induction process through FCA. We hypothesize that FCA is a good method for this use. One, it is independent of external factors, such as expert involvement as in [12] or a task specific parameter tuning as done in [7]. Two, by design it is a method targeting property-based entity descriptions as argued for in [6], [5].

Our proposition is that the schema index returns the graphs necessary for a complete query evaluation:

$$q(\Gamma(q)) = q(\mathcal{D}) \quad \text{(for every query } q\text{)} \tag{2}$$

However, $\Gamma(q)$ may return graphs that might be disregarded for evaluating BGP. Hence, the proposed schema index is not necessarily optimal. For a future implementation we plan to approximate the optimal case (cf. (1)). This must only be done to a degree that is sensible, since potential savings in execution time are being countered by costs for building and looking up the index.

Towards a realization of the proposed schema index, we expect to face questions such as for the size of formal contexts as given through the feature domains of a graph. Next steps for our research will also involve empirical analysis of runtime behaviour and scalability of FCA in the LOD use scenario. We have indications that in spite of exponential worst-case complexity, the property-based clustering of data items may lead to empirically acceptable runtime behaviour. A further question in this context is how to efficiently update existing lattices, when, e.g., encountering further features of an object while still ingesting a graph in a stream-based fashion. For such a scenario, we will also look into what lattice construction algorithm to choose. Finally, we plan to address the problems of reducing the size of formal contexts and lattices through appropriate data preparation and cleansing lattices of "noisy" concepts.

References

1. M. Alam, A. Buzmakov, V. Codocedo, and A. Napoli. Mining definitions from RDF annotations using formal concept analysis. *Proc. of IJCAI*, pages 823–829, 2015.
2. F. Baader, B. Ganter, and U. Sattler. Completing description logic knowledge bases using formal concept analysis. *Proc. of IJCAI*, pages 230–235, 2007.
3. P. Cimiano, A. Hotho, and S. Staab. Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis. *Journal of Artificial Intelligence Research*, 24:305–339, 2011.
4. C. D'Amato, N. Fanizzi, and F. Esposito. Inductive learning for the Semantic Web: What does it buy? *Semantic Web*, 1(1-2):53–59, 2010.
5. T. Gottron, M. Knauf, S. Scheglmann, and A. Scherp. A Systematic Investigation of Explicit and Implicit Schema Information on the Linked Open Data Cloud. *Proc. of ESWC*, 7882:228–242, 2013.
6. S. Homoceanu, P. Wille, and W. Balke. ProSWIP: Property-based data access for semantic web interactive programming. In *Proc. of ISWC*, pages 184–199, 2013.
7. K. Kellou-Menouer and Z. Kedad. Schema discovery in RDF data sources. In *Proc. of ER*, pages 481–495, 2015.
8. M. Konrath, T. Gottron, S. Staab, and A. Scherp. Schemex - efficient construction of a data catalogue by stream-based indexing of linked data. *Web Semantics*, 16:52–58, 2012.
9. E. Prudhommeaux, A. Seaborne, et al. Sparql query language for rdf. *W3C recommendation, https://www.w3.org/TR/rdf-sparql-query/*, 15, 2008.
10. S. Scheglmann, G. Groener, S. Staab, and R. Lämmel. Incompleteness-aware programming with RDF data. *Proc. of the 2013 workshop on Data driven functional programming, DDFP 2013*, pages 11–14, 2013.
11. J. Völker and M. Niepert. Statistical schema induction. In *Proc. of ESWC*, pages 124–138, 2011.
12. J. Völker and S. Rudolph. Fostering web intelligence by semi-automatic OWL ontology refinement. *Proc. of Web Intelligence, WI 2008*, pages 454–460, 2008.
13. R. Wille. *Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts*, pages 445–470. 1982.
14. D. Yuan and P. Mitra. Lindex: A lattice-based index for graph databases. *VLDB Journal*, 22(2):229–252, 2013.

Contribution to the Classification of Web of Data based on Formal Concept Analysis

Justine Reynaud[1,2], Yannick Toussaint[1,2], and Amedeo Napoli[1,2]

[1] INRIA Nancy-Grand Est, 54600 Villers-les-Nancy, F-54600, France
[2] Université de Lorraine, LORIA, UMR 7503, Vandoeuvre-les-Nancy, F-54506, France

Abstract. During the last decade, the web has taken a huge importance in everyday life, and has become what is commonly called a web of data. The available resources can be used by human agents but also by software agents, as it is the case for very large ontologies such as YAGO or resources such as DBpedia. These particular datasets can be linked together for constituting the Linked Open Data (LOD) cloud, where basic data are expressed as (subject, predicate, object) triples. One issue of main interest is knowledge discovery within LOD, which can help information retrieval and knowledge engineering. Formal concept analysis (FCA), which is a mathematical theory allowing classification and data analysis, was already used to classify LOD elements. In this research work, we are interested in analyzing the different approaches (extensions) based on FCA for knowledge discovery in the web of data. One objective is to study the efficiency and the applicability of the existing approaches and to propose some improvements.

Keywords: Formal Concept Analysis, Pattern Structures, Triadic Concept Analysis, Linked Open Data

1 Introduction

In this paper, we would like to extend and complete a previous work on the classification of Linked Open Data (LOD) [2]. The basic unit of LOD is the RDF triple which is composed of three elements, namely a subject, a predicate and an object, i.e. *(subject, predicate, object)*. It can be noticed that subjects, predicates and objects can be organized into a partial ordering depending on a specific schema (i.e. RDFS) or an ontology (e.g. YAGO or DBpedia Ontology).

The classification of LOD should take into account the RDF triple as a basic unit and this can be done in several ways. We distinguish ways that relies on a "triples" view of LOD and ways that relies on a "graph" view. Following the lines of [2], we consider an approach which considers triples and which is based on pattern strucures [6]. We complete this approach by integrating the organization of predicates in the classification of RDF triples.

In [2], it was shown how the classification of RDF triples amounts to classify pairs of objects and attributes –as in a binary context– where attributes are partially ordered. Actually, given a subject which corresponds to an object

in a binary context, predicates are considered "one by one" and attributes are viewed as "ranges" of the predicates. Attributes are organized within a partial ordering and there are two main ways of dealing with this order. The first one is to consider a "scaling" as in [3,4] where the description of an attribute correspond to the set of its ancestors in the attribute hierarchy, and the similarity between two attributes is given by the minimal elements in the intersection of their descriptions In the second way, it can be shown that pattern structures for structured sets of attributes are very well adapted to solve the problem of classifying RDF triples for analyzing the content of LOD. As it was precised in [6], it can be considered that the description of an attribute is an antichain and the similarity is given by the intersection of antichains (which is actually an alternative way of handling the scaling introduced just above).

Both approaches are discussed in [2], where in addition a specific procedure based on RMQ is used for efficiently computing the intersection of antichains. In the present work we add the "predicate dimension" within RDF triples and we propose first elements for extending our preceding approach by considering the predicate classification. The basics of the approach are detailed but for the moment no experiments are proposed, which is planned in a future work. However, we show that this new proposal is sound and formally consistent.

The paper is organized as follows. First, we present some preliminaries about the web of data and LOD. Then, we recall and discuss the previous approaches for classifying RDF triples or RDF graphs. Finally, we detail and illustrate our new proposal, and prove a main proposition on the similarity of object descriptions.

2 Web of data

Basically, the web of data consists of resources and relations between those resources. It can be represented as a graph where nodes are resources and edges are relations.

The core of the web of data is the `RDF` (Resource Description Framework) language, based on graph model where basic units are (`subject`, `predicate`, `object`) triples. These triples, also called `RDF` statements, describe facts [1].

Definition 1 (RDF Triple). *Given a set of URIs U, blank nodes B and literals L, an RDF triple is represented as* $t = (s,p,o) \in (U \cup B) \times U \times (U \cup B \cup L)$*, where:*

- `s` *is called subject,* `p` *predicate and* `o` *object;*
- *U is the set of all resources identified by a* `URI` *(Universal Resource Identifier);*
- *B is the set of resources that are unidentified (called* blank node*);*
- *L is the set of literals, which are values like strings, dates or integers.*

As its name suggests, `RDF` allows one to describe *resources*. Resources can refer to any object or thing. For convenience, the terms borrowed from description logics can be used to distinguish different types of resources:

Properties: express binary relationships between any two entities;

Classes: represent sets of entities;

Instances: entities that belong to a class;

Variables: unidentified resources (i.e. blank nodes).

`RDF` has some specific vocabulary. For example, the property `rdf:type` enables to declare an instance as belonging to a class. This expression has two parts, separated by a colon. The second part identifies the specific resource of this vocabulary, here "type". The first part, also called *prefix*, is an abbreviation for "`http://www.w3.org/1999/02/22-rdf-syntax-ns#`", the *namespace* which refers to the `RDF` vocabulary. Each vocabulary has its namespace.

In order to give `RDF` some structure, schemas are used. Schemas describe constraints on facts. They correspond to the TBox in description logics terms. Each vocabulary may have its own schema, that is a structure between its entities, called its reference schema. The structure of `RDF` triples is based on `RDFS` (`RDF` Schema), bringing additionnal properties such as `rdfs:subPropertyOf` and `rdfs:subClassOf`. These two additionnal properties enable to build a hierarchy of relations one the one hand and classes on the other hand. Afterwards, these properties will be denoted `subP` and `subC`, respectively. Instances can be linked to the hierarchy of classes, but they are not part of it, since they are related to a class with `rdf:type`. Thus, in the following, a hierarchy will refer to an orderred set of classes, and instances will be attached to their class (for simplicity, a unique class per instance is considered here).

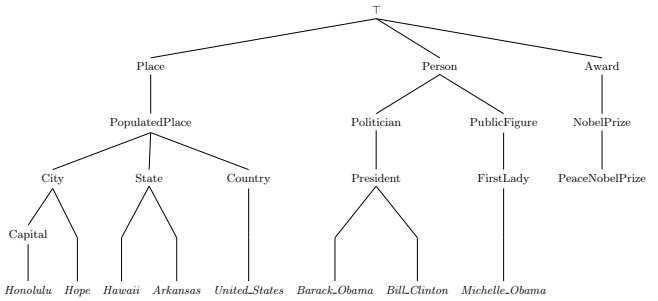
The language `SPARQL` offers to run queries on the web of data. A query is composed of `RDF` triples containing variables. For example, the query `SELECT ?x WHERE {?x rdf:type C}` returns all the instances of C.

A toy knowledge base, freely inspired by the example of S. Ferré in [5], is presented in Figure 1. The Figures 1b and 1d correspond to a set of `RDF` statements and the associated graph respectively. The Figures 1c and 1a represent the background knowledge. Instances of the knowledge base are considered and linked to the class hierarchy.

Reference schemas. Resulting from the linked open data, each data set is connected to others. Thus, hierarchies are not guarantee to be a tree – instances belonging to classes of their schema and to classes of another schemata for example.

In this work, we consider that all the classes on one hand and all the properties on the other hand belong to the same reference schema; that is, have the same namespace. We will also assume that, for any reference schema, there are neither `subP` nor `subC` cycle.

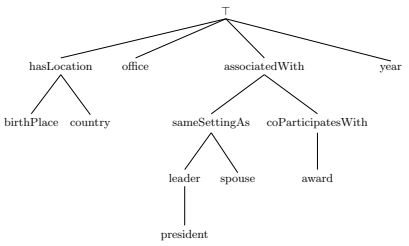
Another concern is that, the tree structure is not guaranteed, even if there is no cycle. If C_1 and C_2 are two incomparable classes, both subclasses of C_3 and C_4 that are also incomparable, then, we lost the tree structure and we have a conflict. Here, we suppose that we know how to linearize the hierarchy of each class as the linearization of a product of two partial orderings.



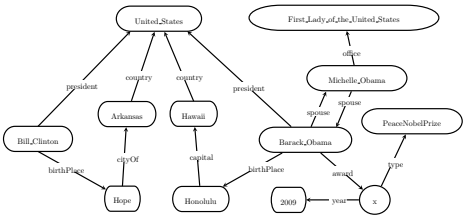
(a) `subC` relation

```
Barack_Obama president United_States .
Bill_Clinton president United_States .
Barack_Obama birthPlace Honolulu .
Bill_Clinton birthPlace Hope .
Honolulu capital Hawaii .
Hope city Honolulu .
Arkansas country United_States .
Hawaii country United_States .
Barack_Obama spouse Michelle_Obama .
Michelle_Obama office First_Lady_of_the_United_States .
Michelle_Obama spouse Barack_Obama .
Barack_Obama award x .
x type PeaceNobelPrize .
x year 2009 .
```

(b) Knowledge base as `RDF` triples



(c) `subP` relation



(d) Knowledge base as graph

Fig. 1: **Toy knowledge base.** The subfigures 1d and 1b illustrate a set of facts (this corresponds to the ABox in description logics terms). Subfigure 1c illustrates the hierarchy of properties w.r.t. the `subP` relation. The subfigure 1a shows the hierarchy of classes with their instances.

3 Formal Concept Analysis

Formal Concept Analysis (FCA) [7] is a mathematical framework used for classification and knowledge discovery. As a learning process, FCA allows one to build an ordered set of concepts where objects are classified w.r.t. the attributes that they share.

A lot of extensions have been proposed, and some of them can be usefull to deal with WOD. Two approaches are possible: the first consists in considering the `RDF` graph itself whereas the second consists in considering the `RDF` statements.

Classifying the web of data enables to find inconsistencies resulting from the merging of different data sets. It is also a way to discover relationships or implications that are not explicit in any single data set. Moreover, the visual support given by the lattice allows users to easily navigate through hierarchy for exploratory research.

3.1 WOD as a set of statements

The first approach consider WOD as a set of RDF triples. This approach is related to the works of [4], [8], [9] and [2].

Classification w.r.t. background knowledge The WOD can be classified through RDF triples. However, in order to be interesting enough, this approach has to take into account background knowledge such as the classes and properties hierarchies. This problem has been developed in [4]. The authors consider a set of documents as objects and a set of terms as attributes. Terms belong to a thesaurus and they are organized in a tree structure. In order to build the lattice, they consider M^* the set of terms in the thesaurus, and define an order $\leqslant_{M^*}$, meaning that "*any attribute implies any of its more general attributes.*" For example, if the term *indexing* is more specific than *information-analysis* in the thesaurus and if it is an attribute of a document d, then *information-analysis* is also an attribute of d. Then, the authors define the intersection $\cap^*$ of two intents m_1 and m_2 as "*the most specific attributes in M^* that are more general than m_1 and m_2.*" With this new operator, they can build concept lattices taking into account background knowledge. In the following, we will present an extension of this approach.

Triadic Concept Analysis The triadic concept analysis (TCA) [12] considers an additionnal dimension (the modus). This implies a ternary relation $Y \subseteq G \times M \times B$ which can be interpreted as *"the object g takes the value b under the condition m."* The corresponding Galois connections are more complex: they are three and each one corresponds to a dimension expressed in terms of the two others.

Mining triconcepts. TCA is used to describe folksonomies, i.e. communities of users U who can annotate resources R with keywords (tags) T. An example of taxonomy is Bibsonomy, a tool allowing users to manage publications and to tag them. Here a concept would be a group of users tagging a set of resources with identical keywords The algorithm Trias [8] has been proposed to mine tri-concepts. It is based on a projection of the triadic context (U, T, R, Y) onto a dyadic context $(U, (T \times R), I)$. For each concept (A, J) found in $(U, (T \times R), I)$ context, a new dyadic context (T, R, J) is built. For each concept (B, C) found, (A, B, C) is a concept of (U, T, R, Y).

Finding biclusters. In [9], the authors aim to find biclusters in a numerical dataset. The context is similar to a standard formal context, but instead of a binary context, a context with numerical values is considered. In order to have a triadic context, objects and attributes remain the same and the numerical values are transformed into a third dimension by means of an interordinal scaling. A threshold θ is provided and the new dimension is made up of intervals whose range sizes are less or equal to θ.

Limitations. Given that TCA can handle three dimensions, it could be interesting for WOD. However, a simple approach does not take into account

the background knowledge. Thus, taking into account the hierarchies implies a scaling. This implies to add all classes in the dimension of subjects and objects, and all properties in the dimension of predicates. This approach works well in the case of biclustering, when a threshold is given, limiting the number of intervals to consider. By contrast, with WOD, such a threshold can hardly be considered. Moreover, having this constraint does not guarantee the scalability.

Pattern structures Pattern structures (PS) [6] are proposed in order to consider data which are not binary data. Attributes then become descriptions which are partially ordered thanks to a similarity operation. Formally, a pattern structure is defined as follows:

Definition 2 (Pattern structure). *Let G be a set of objects, $(D, \sqcap)$ a semi-lattice and $\delta : G \to D$ a mapping. Then $(G, (D, \sqcap), \delta)$ is called a pattern structure. The new Galois connections are the following:*

- $A^{\square} = \bigsqcap_{g \in A} \delta(g)$ *for* $A \subseteq G$
- $d^{\square} = \{g \in G \mid d \sqsubseteq \delta(g)\}$ *for* $d \in D$

Pattern structures are used on triples in [2]. In this work, the authors aim to provide a navigation space over RDF resources. The underlying idea is that, when a resource is a subject in a triple, we can consider that the pair (p,o) is *describing* this resource. Thus, two resources can be compared regarding how they are described by the triples in which they are subjects. Moreover, there is background knowledge related to objects provided by the properties `rdf:type` and `subC`. Considering a set of triples $\mathcal{B} = (s_i, p_j, o_k)$ and a hierarchy of classes, this idea can be materialized in term of formal concepts. To that purpose, a pattern structure $(G, (D, \sqcap), \delta)$ is constructed as described below.

Entities[3] and their descriptions.

The set $\mathcal{B}$ of `RDF` statements is built with a `SPARQL` query on a specific namespace. The associated reference schema is used to construct the hierarchy of classes from this namespace.

First, resources that are subjects of at least one triple in $\mathcal{B}$ are considered as entities of the pattern structure: $G = \{s_1, \ldots s_n\}$. They will be compared regarding the objects they share for each predicates. Objects can be instances or classes, but here only classes are considered. Indeed, the hierarchy between the objects is based on the property `rdfs:subClassOf`, but instances are linked to classes with the relation `rdf:type`. In order to maintain the consistency, objects that are instances are replaced by the class they belong to.

Each pair (p, o) such that $(s, p, o) \in \mathcal{B}$ is mapped to a description $d \in D$. A description $d \in D$ is a pair (p_i, O_i) where p_i is a predicate and O_i is a set of objects in the range of p_i. Given an entity $s \in G$, its description is defined as follow: $\delta(s) = \{d_1^s, \ldots, d_n^s\}$ where $d_j^s = (p_j, O_j^s)$ with $j = \{1, \ldots, n\}$. Each

[3] The term *object* is ambiguous: it denotes both the first part of a pattern structure and the last element of an `RDF` triple.The term *entity* will be used to denote objects of the pattern structure. The term *object* remains for the triples.

elementary description (p_i, o_j) is replaced with $(p_i, C(o_j))$ where $C(o_j)$ is the class of o_j in the reference schema.

Given two descriptions $\delta(x) = \{d_1^x, \ldots, d_n^x\}$ and $\delta(y) = \{d_1^y, \ldots, d_m^y\}$, we have $\delta(x) \sqsubseteq \delta(y)$ if $\forall d_i^x \in \delta(x), \exists d_j^y \in \delta(y)$ s.t. $i = j$ and $\forall o_i^x \in O_i^x, \exists o_j^y \in O_j^y$ s.t. $C(o_i^x)$ `subC` $C(o_j^y)$.

Similarity between descriptions. The hierarchy of classes from the reference schema is considered. Thus, the similarity between two objects is defined as follows:

$$\begin{aligned}\delta(x) \sqcap \delta(y) =& \{d_1^x, \ldots, d_n^x\} \sqcap \{d_1^y, \ldots, d_m^y\} \\ =& min \bigcup_{\substack{i \in \{1,\ldots,n\} \\ j \in \{1,\ldots,m\}}} \{d_i^x\} \sqcap \{d_j^y\}\end{aligned}$$

where min returns the minimal elements

$$\{d_i^x\} \sqcap \{d_j^y\} = \begin{cases} lcs(C(o_i^x), C(o_j^y)) & \text{if i = j} \\ \emptyset & \text{else} \end{cases}$$

where lcs returns the most specific superclass

This definition is close to the $\cap^*$ used in [4]. The $\sqcap$ operation between two descriptions corresponds to finding the most specific attribute in M^*. Considering only the minimal of the union corresponds to "*retaining only the most specific elements of the set generated this way*".

Limitations. This work introduces a method to mine triples with pattern structures as in [2]. The main limitation is that, the hierarchy of predicates is not considered.

3.2 WOD as a graph

Instead of considering `RDF` triples, it is possible to consider the associated graph. This is what is done in some approaches like [11], [10] and [5].

Pattern structures for graphs In [11], pattern structures are used to classify graphs. Each graph is considered as an object and the set of all its subgraphs is considered as the description. Thus, the similarity between two graphs is the set af all the subgraphs they have in common.

As this approach is expensive, graphs can be simplified by the mean of projections. That is, instead of considering all the subgraphs of a graph, the description is something simpler like the set of chains of a certain size that compose the graph.

Concept lattices of conceptual graphs In [5], an extension to FCA for conceptual graphs, called G-FCA, is proposed. Compared to RDF graphs, conceptual graphs (CG) are oriented bipartite graphs. The two kinds of nodes are

classes in one hand and relations in the other hand. Contrary to RDF graphs which consider only binary relations, CGs handle n-ary relations. *Projected graph patterns* are introduced as concepts. It is similar to a SPARQL query where the graph query is the intent and the candidate solutions are the extent.

Example 1. Given the Figure 1, we have the concept *({(Hope,Arkansas), (Honolulu,Hawaii)}, {(?p, birthPlace, ?x), (?x, city, ?y)})*. Note that, the syntax is different from [5], since the example is adapted to `RDF`. This concept is associated to the query `SELECT ?x ?y WHERE { ?p birthPlace ?x . ?x city ?y.}`

Concept lattices of `RDF` graphs In [10], in addition to the `RDF` graph, a formal context corresponding to the background knowledge is considered. This context has a set of resources that are objects and attributes at the same time. Considering one resource as object, its attributes are the set of resources that are "more general" regarding `subC` and `subP` properties.

Example 2. Given the knowledge base Figure 1, a part of the formal context could be the following:

	City	Capital	president	sameSettingAs	*Honolulu*	*Hawaii*
City	×					
Capital	×	×				
president			×	×		
sameSettingAs				×		
Honolulu	×	×			×	
Hawaii						×

The extent of the pattern is a set of resources whereas the intent is a triple graph. A triple graph is basically a subgraph and some background knowledge. A morphism between two triples graphs is defined and corresponds to an order on intents. Moreover, a product between triple graphs is defined such that the join of two concepts (i.e. triple graphs) corresponds to their product.

Example 3. Given the knowledge context and the graph pattern corresponding to the triple (Honolulu,capital,Hawaii), the graph corresponding to the triple (Honolulu,city,Hawaii) is more general.

4 Taking into account the three parts of the triple

In this section, we propose a generalization of [2] : instead of considering subjects described by (predicate,object) pairs, we consider the entire triple as a description.

Entities and descriptions We suppose that each triple has a unique identifier, like a *transaction id*. These identifiers are the entities of the pattern structure : $G = \{t_1, \ldots, t_n\}$. The description of an entity is a mapping to the triple itself, where instances are replaced by the class they belong. As to not complefy the notation, $C(s)$ and $C(o)$ will be written s and o.

Order on descriptions Given the descriptions $\delta(t_i) = (s_i, p_i, o_i)$ and $\delta(t_j) = (s_j, p_j, o_j)$, the partial order on descriptions is defined as follow:

$$\delta(s_i, p_i, o_i) \sqsubseteq \delta(s_j, p_j, o_j) \Leftrightarrow s_j \texttt{ subC } s_i, p_j \texttt{ subP } p_i, o_j \texttt{ subC } o_i$$

Similarity between descriptions The similarity between two triples t_i and t_j is defined as follow:

$$\delta(t_i) \sqcap \delta(t_j) = (lcs_c(s_i, s_j), lcs_p(p_i, p_j), lcs_c(o_i, o_j))$$

Proposition 1. $\delta(t_i) \sqsubseteq \delta(t_j) \Leftrightarrow \delta(t_i) \sqcap \delta(t_j) = \delta(t_i)$.

Proof.

$$\begin{aligned}
\delta(t_i) \sqsubseteq \delta(t_j) &\Leftrightarrow s_j \texttt{ subC } s_i, p_j \texttt{ subP } p_i, o_j \texttt{ subC } o_i \\
&\Leftrightarrow lcs_c(s_i, s_j) = s_i, lcs_p(p_i, p_j) = p_i, lcs_c(o_i, o_j) = o_i \\
&\Leftrightarrow (lcs_c(s_i, s_j), lcs_p(p_i, p_j), lcs_c(o_i, o_j)) = (s_i, p_i, o_i) \\
&\Leftrightarrow (s_i, p_i, o_i) \sqcap (s_j, p_j, o_j) = (s_i, p_i, o_i) \\
&\Leftrightarrow \delta(t_i) \sqcap \delta(t_j) = \delta(t_i)
\end{aligned}$$

Similarity between set of triples The similarity of two triples can be generalized to set of triples. The description of a set of triple T is the set of descriptions of each of its triples : $\Delta(T) = \{\delta(t) \mid t \in T\}$.

Given two sets of triples T_1 and T_2, the similarity $T_1 \sqcap T_2$ is the set of minimal triples $\delta(t_i) \sqcap \delta(t_j)$ for all t_i in T_1 and for all t_j in T_2 given the order on descriptions.

5 Conclusion

In this paper we discussed some approaches for dealing with the classification of web of data, and more precisely of sets of RDF triples, i.e. *(subject, predicate, object)*. Actually, this kind of classification process is based on the classification of pairs *(subject, attribute)* which simulate the RDF triples and where attributes correspond to object triples and are structured within a hierarchy. Here, we proposed an extension to a previous work which takes into account the classification of predicates which was not the case before. We gave a formal presentation of this proposal and for future work we are planning to make a series of experiments which should (hopefully) validate the current approach.

Acknowledgments

Justine Reynaud is preparing her PhD Thesis with the support of "Région Lorraine" and "Délégation Générale de l'Armement".

References

1. Serge Abiteboul, Ioana Gabriela Manolescu Goujot, and Philippe Rigaux. *Web data management*. Cambridge University Press, New York, 2012.
2. Mehwish Alam, Aleksey Buzmakov, Amedeo Napoli, and Alibek Sailanbayev. Revisiting Pattern Structures for Structured Attribute Sets. In Sadok Ben Yahia and Jan Konecny, editors, *The Twelth International Conference on Concept Lattices and their Applications – CLA 2015, Clermont-Ferrand, France*, pages 241–252. LIMOS Clermont-Ferrand, 2015. CEUR Workshop Proceedings 1466 `http://ceur-ws.org/Vol-1466`.
3. Claudio Carpineto and Giovanni Romano. A lattice conceptual clustering system and its application to browsing retrieval. *Machine Learning*, 24(2):95–122, 1996.
4. Claudio Carpineto and Giovanni Romano. *Concept Data Analysis: Theory and Applications*. John Wiley & Sons, Chichester, UK, 2004.
5. Sébastien Ferré. A Proposal for Extending Formal Concept Analysis to Knowledge Graphs. In *Formal Concept Analysis*, volume LNCS 9113, pages 271–286, Nerja, Spain, June 2015.
6. Bernhard Ganter and Sergei O. Kuznetsov. Pattern structures and their projections. In *ICCS*, LNCS 2120, pages 129–142. Springer, 2001.
7. Bernhard Ganter and Rudolf Wille. *Formal concept analysis - mathematical foundations*. Springer, 1999.
8. Robert Jäschke, Andreas Hotho, Christoph Schmitz, Bernhard Ganter, and Gerd Stumme. TRIAS - an algorithm for mining iceberg tri-lattices. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), 18-22 December 2006, Hong Kong, China*, pages 907–911, 2006.
9. Mehdi Kaytoue, Sergei O. Kuznetsov, Juraj Macko, and Amedeo Napoli. Biclustering meets triadic concept analysis. *Annals of Mathematics and Artificial Intelligence*, 70(1-2):55–79, feb 2014.
10. Jens Kötters. Concept lattices of rdf graphs. *Formal Concept Analysis and Applications FCA&A 2015*, page 81, 2015.
11. Sergei O. Kuznetsov. *Computing Graph-Based Lattices from Smallest Projections*, pages 35–47. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
12. Fritz Lehmann and Rudolf Wille. *A triadic approach to formal concept analysis*, pages 32–43. Springer Berlin Heidelberg, Berlin, Heidelberg, 1995.

From a Possibility Theory View of Formal Concept Analysis to the Possibilistic Handling of Incomplete and Uncertain Contexts

Zina Ait-Yakoub[1], Yassine Djouadi[2], Didier Dubois[2], and Henri Prade[2]

[1] Department of Computer Science
University of Tizi-Ouzou, BP 17, RP, Tizi-Ouzou, Algeria
ait.yakoub@hotmail.fr
[2] IRIT, 118 Route de Narbonne, 31062 Toulouse, Cedex 9, France
djouadi@irit.fr, Didier.Dubois@irit.fr, Henri.Prade@irit.fr

Abstract. The formal similarity between possibility theory and formal concept analysis, made ten years ago, has suggested the introduction in the latter setting of the counterpart of possibilistic operators, which were ignored before. These new operators can be related to the basic operator of formal concept analysis by a triple use of negations on the contexts, on the set-valued arguments and on the obtained results, and lead to consider new compositions worth of interest. They enable us to complete the Guigues-Duquenne basis with rules having disjunctive conclusions. Besides, the approach can be naturally generalized to incomplete contexts and then to uncertain context where uncertainty is graded.

1 Introduction

Formal Concept Analysis (FCA) considers the classical Galois derivation operator (i.e. the sufficiency operator) for extracting formal concepts organized within a hierarchy (i.e. partial ordering) called the concept lattice. The concept lattice has proved highly useful for knowledge discovery. The knowledge is expressed as attribute implications, that are formulas in the form $\{a_1, ..., a_n\} \rightarrow \{b_1, ..., b_m\}$ where $a_1, ..., a_n, b_1, ...$ and b_m are attributes. It is considered that the underlying semantics is a conjunctive one. Indeed, by $\{a_1, ..., a_n\} \rightarrow \{b_1, ..., b_m\}$, the interpretation "$a_1$" and ... and "$a_n$" $\rightarrow$ "b_1" and ... and "b_m" is implicitly agreed.

Recently, Dubois and Prade [6] [9] have given a possibility-theoretic reading of formal concept analysis. Beyond the sufficiency operator currently used in FCA, the possibilistic interpretation proposed by these authors allows to consider three other (powerset) operators namely possibility, necessity and dual sufficiency [5] [3]. In this spirit, the aim of this paper is to enlarge the knowledge representation capability of FCA to so-called "disjunctive attribute implications" instead of the conjunctive attribute implications considered by current approaches [1] (introduced in [11]). It will be shown that the proposed approach considers "open-closed" pairs obtained by means of the asymmetric composition ($\mathrm{N} \circ \Pi$) of necessity and possibility operators, and then we propose a method for inducing disjunctive attribute implications.

The remainder of the paper is organized as follows. Section 2 gives a background on FCA. The possibility-theoretic view of FCA is discussed in section 3, whereas the

next section presents our contribution which highlights the interest of using possibility theory operators in order to induce disjunctive attribute implications from formal contexts. Section 5 presents the same analysis for incomplete formal contexts and finally, Section 6 deals with necessity degrees in uncertain formal contexts.

2 Formal concept analysis: basic notions

Formal concept analysis [1] is a lattice-based setting for data analysis and knowledge representation. It relies essentially on a binary relation between a set of objects and a set of attributes. This relation is called a formal context. More formally, a formal context is a triple $\mathcal{K} = (\mathcal{O}, \mathcal{P}, \mathcal{R})$ where $\mathcal{O}$ is a set of objects, $\mathcal{P}$ a set of attributes and $\mathcal{R}$ a binary relation s.t. $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{P}$. $x\mathcal{R}a$ means that the object x satisfies the attribute a.

Example 1. We consider an example of formal context $\mathcal{K}_S = (\mathcal{O}, \mathcal{P}, \mathcal{R})$ given in Table 1 where $\mathcal{O} = \{John, Maria, Peter, Clara\}$ and $\mathcal{P} = \{Man, Woman, Father, Mother, Parent\}$. The cross mark $\times$ indicates that the related object satisfies the corresponding attribute. Whereas the empty mark indicates the contrary.

The paradigm of formal concept analysis [14] is classically based of an adjoint pair of operators $(.)^\Delta : 2^\mathcal{O} \to 2^\mathcal{P}$ and $(.)^\Delta : 2^\mathcal{P} \to 2^\mathcal{O}$ (called Galois derivation operator in the literature) defined for two sets $X \in 2^\mathcal{O}$ and $A \in 2^\mathcal{P}$ as follows :

$$A^\Delta = \{x \in \mathcal{O} \mid \forall a \in \mathcal{P}\ (a \in A \Rightarrow x\mathcal{R}a)\}$$
$$X^\Delta = \{a \in \mathcal{P} \mid \forall x \in \mathcal{O}\ (x \in X \Rightarrow x\mathcal{R}a)\}$$

That is, A^Δ corresponds to the set of objects that satisfy all attributes in A. Similarly, X^Δ corresponds to the set of of attributes that are satisfied by all objects in X.

A formal concept of $\mathcal{K}$ is a pair of closed sets (X, A) with $X \subseteq \mathcal{O}, A \subseteq \mathcal{P}$ such that $X^\Delta = A$ and $A^\Delta = X$. X is called the extent and A the intent of the formal concept (X, A). For instance, ({Clara}, {Woman, Parent, Mother}) is a formal concept of K_S. The set of all formal concepts (denoted by $\mathcal{B}(\mathcal{O}, \mathcal{P}, \mathcal{R})$) equipped with a partial order $\preceq$ defined as: $(X_1, A_1) \preceq (X_2, A_2)$ if $X_1 \subseteq X_2$ (or equivalently, $A_2 \subseteq A_1$) forms a complete lattice (denoted by $\mathfrak{L}(\mathcal{O}, \mathcal{P}, \mathcal{R})$).

Formal concepts lattices can be characterized in terms of attribute implications [10]. An attribute implication is an expression $A \to B$ where A and B are subsets of attributes ($A, B \in 2^\mathcal{P}$) and it holds in a formal context if $A^\Delta \subseteq B^\Delta$ (equivalently $B \subseteq A^{\Delta\Delta}$). The semantics of the attribute implication is that, for every object $x \in \mathcal{O}$, if every attribute from A applies to the object x, then every attribute from B also applies to x. It is important to remark that the underlying semantics is a conjunctive one. Thus, our objective in the following is to consider additional knowledge in the form of so-called disjunctive attribute implications.

Table 1. Formal context K_S.

$\mathcal{R}$	Man	Woman	Father	Mother	Parent
John	×				
Maria		×			
Peter	×		×		×
Clara		×		×	×

3 Asymmetric Composition of possibilistic operators

The Galois derivation operator which is at the basis of FCA theory is the operator of sufficiency $(.)^{\Delta}$. Some time ago, Dubois and Prade [6,9] have highlighted, in the setting of possibility theory, three other powerset derivation operators, namely the possibility operator (denoted $(.)^{\Pi}$), the necessity operator (denoted $(.)^{\mathrm{N}}$) and the dual sufficiency operator (denoted $(.)^{\nabla}$). The two former operators are given in the following:

— $(A)^{\Pi}$ corresponds to the set of objects that are associated with at least one attribute in A. Formally, we have:

$$(A)^{\Pi} = \{x \in O \mid \exists a \in A, x\mathcal{R}a\}$$

— $(A)^{\mathrm{N}}$ corresponds to the set of objects such that any attribute that satisfies one of them is necessarily in A.

$$(A)^{\mathrm{N}} = \{x \in O \mid \forall a \in \mathcal{P}\ (x\mathcal{R}a \Rightarrow a \in A)\}$$

$(X)^{\Pi}$ and $(X)^{\mathrm{N}}$ are dually obtained.

Let $x\overline{\mathcal{R}}a$ indicates that object x does not satisfy attribute a. In the particular case where the derivation operators $(.)^{\Pi}$, $(.)^{\mathrm{N}}$, $(.)^{\Delta}$ are applied to the complementary context $\overline{\mathcal{K}}(O, \mathcal{P}, \overline{\mathcal{R}})$ (where $\overline{\mathcal{R}} = \{(x, a) \in O \times \mathcal{P} \mid x\overline{\mathcal{R}}a\}$), we will exceptionally use the explicit notation $(.)^{\Pi}_{\overline{\mathcal{K}}}$, $(.)^{\mathrm{N}}_{\overline{\mathcal{K}}}$, $(.)^{\Delta}_{\overline{\mathcal{K}}}$. Given $X \subseteq O$ and $\overline{X}$ its complementary set (i.e. $O \setminus X$), the following recalls some useful properties [5] needed in the rest of the paper.

$$\begin{aligned}
&P_1 : X^{\Delta}_{\overline{\mathcal{K}}} = \overline{(X^{\Pi})}\\
&P_2 : X^{\Delta}_{\overline{\mathcal{K}}} = (\overline{X})^{\mathrm{N}}\\
&P_3 : X_1 \subseteq X_2 \Rightarrow (X_1)^{\Pi} \subseteq (X_2)^{\Pi}\\
&P_4 : X \subseteq ((X)^{\Pi})^{\mathrm{N}}\\
&P_5 : (X_1)^{\Pi} \cup (X_2)^{\Pi} = (X_1 \cup X_2)^{\Pi}\\
&P_6 : X_1 \subseteq X_2 \Rightarrow (X_1)^{\mathrm{N}} \subseteq (X_2)^{\mathrm{N}}\\
&P_7 : (X)^{\Pi} = (((X)^{\Pi})^{\mathrm{N}})^{\Pi}
\end{aligned}$$

These properties are dually satisfied for $A \subseteq \mathcal{P}$.

Let us also denote by NΠ-pair, a formal pair (X, A) s.t. $X = A^{\Pi}$ and $A = X^{\mathrm{N}}$, where X (resp. A) will be called NΠ-extent (resp. NΠ-intent). It may be remarked that both elements X and A present dual topological properties. Indeed, X is an open element, whereas A is a closed one, achieving then an "open-closed" pair. The set of all NΠ-pairs is denoted by $\mathfrak{B}_{\mathrm{N}\Pi}$, whereas the set $\mathfrak{B}_{\mathrm{N}\Pi}(Ext)$ (resp. $\mathfrak{B}_{\mathrm{N}\Pi}(Int)$) corresponds to the set of all NΠ-extents (resp. NΠ-intents). Proposition 1 establishes first a characterization of NΠ-pairs, whereas the proposition 2 gives the algebraic structure of the set $\mathfrak{B}_{\mathrm{N}\Pi}$.

Proposition 1. *Let $X \in 2^{O}$ and $A \in 2^{\mathcal{P}}$, (X, A) is an* NΠ*-pair if and only if $(\overline{X}, A)$ is a formal concept in $\overline{\mathcal{K}}(O, \mathcal{P}, \overline{\mathcal{R}})$.*

Proof. It is proved using properties P_1 and P_2 given in section 3.

It has been already established that the set $\mathfrak{B}_{\mathrm{N}\Pi}$ with a partial order (denoted $\preceq$) defined as $(X_1, A_1) \preceq (X_2, A_2)$ if $X_1 \subseteq X_2$ (or, equivalently, $A_1 \subseteq A_2$) forms a complete lattice, called the NΠ-lattice and denoted by $\mathfrak{L}_{\mathrm{N}\Pi}$. The following proposition gives the infima (greatest lower bound) and the suprema (least upper bound) for a given subset of $\mathfrak{L}_{\mathrm{N}\Pi}$.

Proposition 2. *The infima and suprema of a subset* (X_j, A_j) *(j an index set) of* $\mathfrak{L}_{\mathrm{N}\Pi}$ *are given by:*

$$\bigwedge_{j\in J}(X_j, A_j) = (\bigcup_{j\in J} X_j, ((\bigcup_{j\in J} A_j)^{\Pi})^{\mathrm{N}}); \qquad \bigvee_{j\in J}(X_j, A_j) = (\bigcap_{j\in J} X_j, \bigcap_{j\in J} A_j).$$

Proof. This result can be established using Proposition 1, and the fact that $(\overline{X}, A)$ is a formal concept of $\overline{\mathcal{K}}(\mathcal{O}, \mathcal{P}, \overline{\mathcal{R}})$.

Example 2. Figure 1 illustrates the $\mathfrak{L}_{\mathrm{N}\Pi}$ lattice corresponding to the formal context given in Table 1.

Let us now introduce the mapping μ which associates to each set of attributes $A \in 2^{\mathcal{P}}$, its NΠ-pair such as:

$$\begin{aligned} \mu : 2^{\mathcal{P}} &\rightarrow \mathcal{B}_{\mathrm{N}\Pi} \\ \mathrm{A} &\rightarrow \mu(A) = (A^{\Pi}, (A^{\Pi})^{\mathrm{N}}) \end{aligned}$$

The following proposition establishes the mapping μ for a set A of attributes.

Proposition 3. *Let* $A \subseteq \mathcal{P}$*, then* $\mu(A) = \bigwedge_{a\in A} \mu(\{a\})$

Proof. $A^{\Pi} = \bigcup_{a\in A} a^{\Pi}$ is obtained directly by the definition of possibility operator, we have $\mu(A) = (A^{\Pi}, (A^{\Pi})^{\mathrm{N}}) \Leftrightarrow \mu(A) = (\bigcup_{a\in A} a^{\Pi}, (\bigcup_{a\in A} a^{\Pi})^{\mathrm{N}}) = \bigwedge_{a\in A} \mu(\{a\})$

4 Disjunctive attribute implications

We propose now to introduce disjunctive attribute implications of the form $a_1 \vee ... \vee a_n \longmapsto b_1 \vee ... \vee b_m$ (equivalently denoted by $\bigvee A \longmapsto \bigvee B$ with $A = \{a_1, ..., a_n\}$,

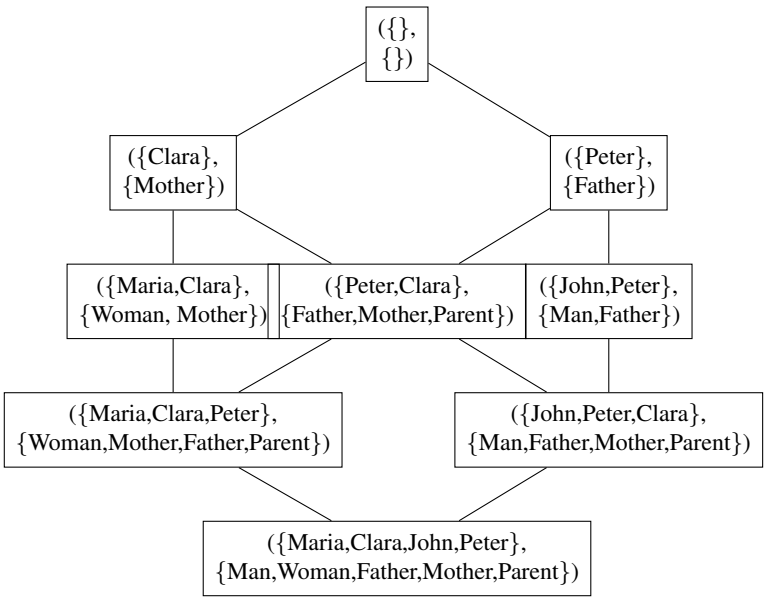


Fig. 1. Lattice $\mathfrak{L}_{\mathrm{N}\Pi}$

and $B = \{b_1, ..., b_m\}$). Being understood that the satisfaction of such an implication is related to the set of all objects in $\mathcal{O}$, we agree that a formal context $\mathcal{K}(\mathcal{O}, \mathcal{P}, \mathcal{R})$ satisfies a disjunctive attribute implication $\bigvee A \longmapsto \bigvee B$ if and only if every object that is never satisfied by each attribute from B is also never satisfied by each attribute from A. Formally, $\mathcal{K} \models \bigvee A \longmapsto \bigvee B$, iff $\forall x \in \mathcal{O}, if\ b_1 \not\subseteq \{x\}^{\Pi} \wedge ... \wedge b_m \not\subseteq \{x\}^{\Pi}$ *then* $a_1 \not\subseteq \{x\}^{\Pi} \wedge ... \wedge a_n \not\subseteq \{x\}^{\Pi}$

For example, the formal context $K_{\mathcal{S}}$ given in Table 1 satisfies the disjunctive attribute implication Parent $\longmapsto$ Father $\vee$ Mother ($K_{\mathcal{S}} \models$ Parent $\longmapsto$ Father $\vee$ Mother).

The following important result can be easily obtained.

Proposition 4. *The disjunctive attribute implication $\bigvee A \longmapsto \bigvee B$ is valid in formal context $\mathcal{K}(\mathcal{O}, \mathcal{P}, \mathcal{R})$ iff the attribute implication $B \longmapsto A$ is valid in formal context $\overline{\mathcal{K}}(\mathcal{O}, \mathcal{P}, \overline{\mathcal{R}})$ iff $A \subseteq ((B)^{\Pi}_{\mathcal{K}})^{\mathrm{N}}_{\mathcal{K}}$.*

Proof. Suppose $B \longmapsto A$ is valid in $\overline{\mathcal{K}}$. In logical terms, it means $\wedge_{b \in B} \neg b \rightarrow \wedge_{a \in A} \neg a$, which is logically equivalent to $\vee_{a \in A} a \rightarrow \vee_{b \in B} b$. Now, $B \longmapsto A$ is valid in $\overline{\mathcal{K}}$ means $A \subseteq B^{\Delta\Delta}_{\overline{\mathcal{K}}}$, that is, $A \subseteq (\overline{B^{\Pi}_{\mathcal{K}}})^{\Delta}_{\overline{\mathcal{K}}}$ iff $A \subseteq ((B)^{\Pi}_{\mathcal{K}})^{\mathrm{N}}_{\mathcal{K}}$. □

A simpler way to assert the satisfaction of a disjunctive attribute implication based on the possibility operator $(.)^{\Pi}$ is given hereafter.

Proposition 5. *Given a formal context $\mathcal{K}(\mathcal{O}, \mathcal{P}, \mathcal{R})$ and $A, B \subseteq \mathcal{P}$, $\mathcal{K} \models \bigvee A \longmapsto \bigvee B$ iff for each $x \in \mathcal{O}$, $B \not\subseteq \overline{\{x\}^{\Pi}}$ or $A \subseteq \overline{\{x\}^{\Pi}}$.*

The disjunctive attribute implications that hold in a formal context $\mathcal{K}(\mathcal{O}, \mathcal{P}, \mathcal{R})$ can be obtained from concept lattice $\mathfrak{L}_{\mathrm{N}\Pi}$. The following proposition illustrates this.

Proposition 6. *Given a formal context $\mathcal{K}(\mathcal{O}, \mathcal{P}, \mathcal{R})$, $\mathcal{K} \models a \rightarrow \bigvee B$ iff $(a^{\Pi}, (a^{\Pi})^{\mathrm{N}}) \leq (B^{\Pi}, (B^{\Pi})^{\mathrm{N}})$*

This means that we have to check in the concept lattice $\mathfrak{L}_{\mathrm{N}\Pi}$ whether the NΠ-pairs associated to a are located above the infima of all NΠ-pairs associated to b from B.

Example 3. In the following we give the set of disjunctive attribute implications that matches to the formal context given in Table 1 by applying the proposition:
{Father→Man, Mother→Woman, Father∨Mother→Parent, Parent→Father∨Mother}

5 Possible and certain implications in incomplete contexts

The case of incomplete context has been only considered by Obiedkov [13] and by Burmeister and Holzer [2] until now. They have proposed to introduce a third value, denoted "?", in a formal context, which leads to the concept of an incomplete context, sometimes also called three values context. More formally, incomplete context $\mathcal{K}_i(\mathcal{O}, \mathcal{P}, \{+, -, ?\}, \mathcal{R}_i)$ where $\mathcal{O}$ is the set of objects, $\mathcal{P}$ the set of attributes, "+", "-", "?" are the three possible entries of the incomplete context, and $\mathcal{R}$ is a ternary relation $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{P} \times \{+, -, ?\}$. The interpretation of the relation $\mathcal{R}$ is as follows. Let $x \in \mathcal{O}$ and $a \in \mathcal{P}$:

— $(x, a, +) \in \mathcal{R}$: it is known that the object x has the attribute a
— $(x, a, -) \in \mathcal{R}$: it is known that the object x does not have the attribute a
— $(x, a, ?) \in \mathcal{R}$: it is unknown, whether the object x has the attribute a or not

An incomplete formal context may be viewed as a weighted family of all standard formal contexts obtained by changing unknown entries $(x, a, ?)$ into known ones ($(x, a, +)$ or $(x, a, -)$). The two extreme cases where all such unknown entries $(x, a, ?)$ are changed into $(x, a, -)$ and the case where all such unknown entries $(x, a, ?)$ are changed into $(x, a, +)$ give birth to lower and upper completions, respectively [8] [4].

In this way, two classical (Boolean) formal contexts, denoted $K_*(\mathcal{O}, \mathcal{P}, \mathcal{R}_*)$ and $K^*(\mathcal{O}, \mathcal{P}, \mathcal{R}^*)$ are obtained as respective results of the two replacements. More formally:

— $K_*(\mathcal{O}, \mathcal{P}, \mathcal{R}_*)$ is a Boolean formal context such that $\mathcal{R}_* = \{(x, a) | (x, a, +) \in \mathcal{R}_i\}$ where $A^{\Delta}_{K_*} = \{x | A \subseteq x\mathcal{R}_*\}$ is the set of objects certainly having all attributes in A
— $K^*(\mathcal{O}, \mathcal{P}, \mathcal{R}^*)$ is a Boolean formal context such that $\mathcal{R}^* = \{(x, a) | (x, a, +) \in \mathcal{R}_i$ or $(x, a, ?) \in \mathcal{R}_i\}$ where $A^{\Delta}_{K^*} = \{x | A \subseteq x\mathcal{R}^*\}$ is the set of objects possibly having all attributes in A.

There exists other intermediate formal contexts by replacing each "?" by "+" or "-" and we obtain exactly 2^n possible formal contexts (n is the number of "?" in the initial formal context). All attribute implications that are obtained from these formal contexts are either possible attribute implications or certain attribute implications. An implication is certain if it is valid in each formal context $\mathcal{K}_j$; this condition may seem hard to verify at first glance. The following theorem solves the problem.

Theorem 1. $A \longmapsto B$ is a certain attribute implication in $\mathcal{K}_i$ iff $A^{\Delta}_{K^*} \subseteq B^{\Delta}_{K_*}$

Proof. Assume that $A \longmapsto B$ is not a certain attribute implication in $\mathcal{K}_i$ and $A^{\Delta}_{K^*} \subseteq B^{\Delta}_{K_*}$. But $A \longmapsto B$ is not certain implication $\Longrightarrow \exists$ a formal context $\mathcal{K}_j | x \in A^{\Delta}_{K_j}$ and $x \notin B^{\Delta}_{K_j}$ $\Longrightarrow \exists$ an object x possibly having all attributes in A and not having the certain attributes in $B \Longrightarrow \exists x \in O \,| x \in A^{\Delta}_{K^*}$ and $x \notin B^{\Delta}_{K_*} \Longrightarrow A^{\Delta}_{K^*} \nsubseteq B^{\Delta}_{K_*}$. □

Another problem is to determine a possible attribute implication that are holds in at least ont formal context $\mathcal{K}_j$, the following theorem facilitates this determination. Proofs are omitted due to space limitations.

Theorem 2. $A \longmapsto B$ is a possible attribute implication in $\mathcal{K}_i$ iff $A^{\Delta}_{K_*} \subseteq B^{\Delta}_{K^*}$

This section also considers disjunctive attribute implications, presented in section 4, in incomplete formal context $\mathcal{K}_i$. As in the case of conjunctive attribute implications we distinguish certain disjunctive attribute implications and possible disjunctive attribute implications. Note that $(A)^{\Pi}_{K_*}$ is the set of objects certainly having at least one attribute in A and $(A)^{\Pi}_{K^*}$ is the set of objects possibly having at least one attribute in A. And $\overline{(A)^{\Pi}_{K_*}}$ is the set of objects that certainly never have any attribute in A and $\overline{(A)^{\Pi}_{K^*}}$ is the set of objects that can never have any attribute in A. We get two major results of this paper.

Theorem 3. $\bigvee A \longmapsto \bigvee B$ is a certain disjunctive attribute implication in $\mathcal{K}_i$ iff $A^{\Pi}_{K^*} \subseteq B^{\Pi}_{K_*}$

Theorem 4. $\bigvee A \longmapsto \bigvee B$ is a possible disjunctive attribute implication in $\mathcal{K}_i$ iff $A^{\Pi}_{K_*} \subseteq B^{\Pi}_{K^*}$

6 Implications from gradually uncertain contexts

In an uncertain formal context the boxes are filled with a pair (α, β) of degree of necessity. That is to say that (α) is the necessity that the object has the attribute, and (β) is the necessity that the object does not have the attribute. Moreover, we should respect the property $\min(\alpha, \beta) = 0$ [7]. Pairs (1,0) and (0,1) correspond to completely informed situations where it is known that object has the attribute (ie. +), respectively the object does not have the attribute (ie. -). The pair (0,0) reflects total ignorance (ie. ?), whereas pairs (α, β) s.t. $1 > \max(\alpha, \beta) > 0$ correspond to partial ignorance.

Consider a pair of thresholds (u, v) with $u > 0$ and $v > 0$. $\mathcal{K}_{(u,v)}$ is an incomplete formal context obtained by replacing:

all entries of the form $(\alpha, 0)$ such that $\alpha \geq u$ by (+)
all entries of the form $(\alpha, 0)$ such that $\alpha < u$ by (?)
all entries of the form $(0, \beta)$ such that $\beta \geq v$ by (-)
all entries of the form $(0, \beta)$ such that $\beta < v$ by (?)

The classical formal context $(\mathcal{K}_{(u,v)})_*$ is obtained by replacing with (+) the pairs $(\alpha, 0)$ such that $\alpha \geq u$ and all the rest with (-). The classical formal context $(\mathcal{K}_{(u,v)})^*$ is obtained by replacing with (-) the pairs $(0, \beta)$ such that $\beta \geq v$ and all the rest with (+).

Observe that $(\mathcal{K}_{(u,v)})_*$ does not depend on v, and increases when u decreases. $(\mathcal{K}_{(u,v)})^*$ does not depend on u, and increases when v increases. Recall that $A^{\Delta}_{\mathcal{K}}$ increases as $\mathcal{K}$ increases (in the sense of inclusion). Therefore, $A^{\Delta}_{(\mathcal{K}_{(u,v)})^*}$ increases when v increases. $B^{\Delta}_{(\mathcal{K}_{(u,v)})_*}$ decreases when u increases.

An attribute implication $A \longmapsto B$ is more certain with u great and v great such that $A^{\Delta}_{(\mathcal{K}_{(u,v)})^*} \subseteq B^{\Delta}_{(\mathcal{K}_{(u,v)})_*}$. Therefore, the degree of certainty $\text{cert}(A \longmapsto B)$ of the attribute implication is equal to the maximum value w such that $A^{\Delta}_{(\mathcal{K}_{(w,w)})^*} \subseteq B^{\Delta}_{(\mathcal{K}_{(w,w)})_*}$. In particular, $\text{cert}(A \longmapsto B) = 1$ iff $A^{\Delta}_{(\mathcal{K}_{(1,1)})^*} \subseteq B^{\Delta}_{(\mathcal{K}_{(1,1)})_*}$ that is to say that the certain attribute implications are calculated with the most certain part of the data. Also a possibility degree is attached to the attribute implication such that $A^{\Delta}_{(\mathcal{K}_{(u,v)})_*} \subseteq B^{\Delta}_{(\mathcal{K}_{(u,v)})^*}$ which is all the greater as u and v are greater.

We also consider the disjunctive attribute implications in the uncertain formal context. Observe that $\overline{(\mathcal{K}_{(u,v)})_*}$ does not depend on v, and increases when u increases, and $\overline{(\mathcal{K}_{(u,v)})^*}$ does not depend on u, and increases when v decreases. Recall that the disjunctive attribute implication $\bigvee A \longmapsto \bigvee B$ is valid in a formal context $\mathcal{K}$ if and only if the attribute implication $B \longmapsto A$ is valid in $\overline{\mathcal{K}}$. Therefore, the degree of certainty $\text{cert}(B \longmapsto A)$ is equal to the maximum value w such that $B^{\Delta}_{(\overline{\mathcal{K}}_{(u,v)})_*} \subseteq A^{\Delta}_{(\overline{\mathcal{K}}_{(u,v)})^*}$, equivalent to $\overline{B^{\Pi}_{(\mathcal{K}_{(u,v)})_*}} \subseteq \overline{A^{\Pi}_{(\mathcal{K}_{(u,v)})^*}}$, which is equivalently written: $A^{\Pi}_{(\mathcal{K}_{(u,v)})^*} \subseteq B^{\Pi}_{(\mathcal{K}_{(u,v)})_*}$. Also a possibility degree is attached to attribute implication such that $A^{\Pi}_{(\mathcal{K}_{(u,v)})^*} \subseteq B^{\Pi}_{(\mathcal{K}_{(u,v)})_*}$ which is all the greater as u and v are greater.

7 Conclusion

All existing works and approaches pertaining to FCA rely on the use of the classical Galois derivation operator (i.e. sufficiency operator). Thus, these works are based on

the complete lattice of all formal concepts obtained using the composition of sufficiency operators. Consequently, induced implications are limited to their conjunctive form. In this paper we propose an approach that enlarges knowledge representation ability to disjunctive attribute implications. Possible links with [12] are to be investigated. The proposed approach considers "open-closed" pairs obtained by means of the asymmetric composition ($N \circ \Pi$) of necessity and possibility operators. We have only focused on composition $(.)^{N\Pi}$. Further researches should concern the study of other possible compositions of possibilistic composite operators such that $(.)^{\Pi\Delta}$, $(.)^{\nabla\Delta}$, etc.

References

1. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, 1999.
2. P. Burmeister and R. Holzer. Treating incomplete knowledge in formal concept analysis. In B. Ganter, G. Stumme, and R. Wille, editors, *Formal Concept Analysis*, pages 114–126. Springer Verlag, Berlin, 2005.
3. Y. Djouadi, D. Dubois, and H. Prade. Possibility theory and formal concept analysis: Context decomposition and uncertainty handling. In *Proc. IPMU'10 13th Int. Conf. on Inform. Process.& Mgmt. of Uncert., Dortmund*, pages 260–269. LNAI 6178, Springer, 2010.
4. Y. Djouadi, D. Dubois, and H. Prade. Graduality, uncertainty and typicality in formal concept analysis. In *35 Years of Fuzzy Set Theory - Celebrat. Vol. Dedicated to the Retirement of E. E. Kerre*, pages 127–147. Springer, 2011.
5. Y. Djouadi and H. Prade. Possibility-theoretic extension of derivation operators in formal concept analysis over fuzzy lattices. *Fuzzy Optimization and Decision Making*, 10:287–309, 2011.
6. D. Dubois, F. Dupin de Saint-Cyr, and H. Prade. A possibility-theoretic view of formal concept analysis. *Fundamenta Informaticae*, 75(1-4):195–213, 2007.
7. D. Dubois and H. Prade. *Possibility Theory*. Plenum Press, 1988.
8. D. Dubois and H. Prade. Formal concept analysis from the standpoint of possibility theory. In *Formal Concept Analysis - Proc. 13th Int. Conf., ICFCA'15, Nerja*, pages 21–38, 2015.
9. D. Dubois and H. Prade.Possibility theory and formal concept analysis in information systems. In *Proc. IFSA'09, Int. Fuzzy Syst. Assoc. World Congr.*, Lisbon,1021-1026, 2009.
10. V. Duquenne. Contextual implications between attributes and some representation properties forfinite lattices. In B. Ganter, R. Wille, and K. Wolff, editors, *Beitrage zur Begriffsanalyse*, pages 213–239. BI Wissenschaftsverlag, 1987.
11. J. L. Guigues and V. Duquenne. Familles minimales d'implications informatives résultant d'un tableau de données binaires. *Mathématiques et Sciences Humaines*, 95:5–18, 1986.
12. R. Missaoui, L. Nourine, and Y. Renaud. Computing implications with negation from a formal context. *Fundamenta Informaticae*, 115(4):357–375, 2012.
13. S. A. Obiedkov. Modal logic for evaluating formulas in incomplete contexts. In U. Priss, D. Corbett, and G. Angelova, editors, *Proc. 10th Int. Conf. on Conceptual Structures (ICCS'02), Borovets, Bulgaria, July 15-19*, volume 2393 of *LNCS*, pages 314–325. Springer, 2002.
14. R. Wille. Restructuring lattice theory: an approach based on hierarchies of concepts. In Ivan Rival, editor, *Ordered sets*, pages 445–470, Dordrecht, Boston, 1982. Reidel.

How Fuzzy FCA and Pattern Structures are connected?

Aleksey Buzmakov[1] and Amedeo Napoli[2]

[1] National Research University Higher School of Economics, Perm, Russia
[2] LORIA (CNRS – Inria – University of Lorraine), Vandœuvre-lès-Nancy, France
avbuzmakov@hse.ru, amedeo.napoli@loria.fr

Abstract. FCA is a mathematical formalism having many applications in data mining and knowledge discovery. Originally it deals with binary data tables. However, there is a number of extensions that enrich standard FCA. In this paper we consider two important extensions: fuzzy FCA and pattern structures, and discuss the relation between them. In particular we introduce a scaling procedure that enables representing a fuzzy context as a pattern structure.

Keywords: fuzzy FCA, pattern structures, scaling

1 Introduction

In this paper we deal with Formal Concept Analysis (FCA) and its extensions. FCA is a mathematical formalism having many applications in data mining and knowledge discovery. It starts from a binary table, a so-called formal context (G, M, I), where G is the set of objects, M is the set of attributes, and $I \subseteq G \times M$ is a relation between G and M, and proceeds to a lattice of formal concepts [1]. Fuzzy FCA is an extension of standard FCA that allows for fuzzy sets of objects and attributes in order to express uncertainty.

Pattern structures is another extension of FCA that allows processing complex data, e.g., graph or sequence datasets. It is a quite general framework and the question if fuzzy FCA can be represented within Pattern Structures and vice versa is still open. In this paper we make a step in this direction and study the connections between pattern structures and fuzzy FCA.

We show how a fuzzy context can be scaled to a "Minimum Pattern Structure" (MnPS), a special kind of pattern structures, that is close to interval pattern structures when considering numerical data. A scaling is needed, since pattern structures deal with crisp sets of objects and, thus, fuzzy extents cannot be expressed within the formalism of pattern structures. For such a kind of scaling we add new objects to the fuzzy context that express objects with uncertain membership in fuzzy sets, allowing expressing fuzzy sets of objects in the formalism of pattern structures. The resulting context is processed by MnPS. This kind of scaling is applicable to fuzzy FCA based on residuated lattices, a special kind of lattices expressing uncertain membership degrees in fuzzy sets.

Table 1: A toy dataset of transactions for a supermaket and the related similarity matrix.

(a) A dataset with 5 transactions.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7
t_1	x	x			x	x	x
t_2	x	x		x		x	x
t_3	x	x			x	x	
t_4		x	x				x
t_5	x			x	x	x	

(b) Similarity matrix.

	t_1	t_2	t_3	t_4	t_5
t_1	1.000	0.714	0.857	0.429	0.429
t_2	0.714	1.000	0.571	0.429	0.429
t_3	0.857	0.571	1.000	0.286	0.714
t_4	0.429	0.429	0.286	1.000	0.000
t_5	0.429	0.429	0.714	0.000	1.000

The rest of the paper is organized as follows. Section 2 describes a running example. Later, in Section 3 we introduce main definitions of fuzzy FCA and pattern structures. The main contribution of this paper is located in Section 4, where we introduce and discuss the scaling procedure of fuzzy FCA to pattern structures. Finally, at the end of the paper we discuss some related works.

2 Running Example

Let us consider a toy dataset of transactions within a supermarket. It is shown in Table 1a. Every row corresponds to a basket bought by a customer and every attribute corresponds to an item that can be bought in the supermarket. A cross in a cell (i, j) means that in the basket i there is the item j.

For making the example concrete, let us consider a clustering task. When dealing with clustering one typically needs a similarity or a distance measure. Such distance and similarity measures for the purpose of this example could be the fraction of different items shared by two baskets $\texttt{Dist}(t_1, t_2) = \frac{|t_1' + t_2'|}{|M|}$ and $\texttt{Sim}(t_1, t_2) = 1 - \texttt{Dist}(t_1, t_2)$, where operation '+' between sets is an exclusive OR (a so-called XOR or the symmetric difference, i.e., $A+B = (A \setminus B) \cup (B \setminus A)$). The similarity measure for any pair of transactions is shown in Table 1b. For example, similarity between t_1 and t_2 is equal 0.714. These baskets are different in two items i_4 and i_5. Thus $\texttt{Dist}(t_1, t_2) = \frac{2}{7} = 0.286$, where 7 is the number of items in the supermarket, and $\texttt{Sim}(t_1, t_2) = 1 - \texttt{Dist}(t_1, t_2) = 0.714$.

3 Definitions

Formal concept analysis (FCA) is a formalism for dealing with data mining and knowledge discovery tasks. It starts from a binary context (G, M, I), where G is the set of objects, M is the set of attributes and $I \subseteq G \times M$ is a relation between G and M. There are a number of extensions of Formal Concept Analysis (FCA) for dealing with complexity of descriptions, e.g., pattern structures [2], and with uncertainty, e.g., fuzzy FCA [3]. Below we give definitions of this two directions

without many details. The interested reader can address the original works on pattern structures and fuzzy FCA for more details and examples.

3.1 Fuzzy FCA

Fuzzy FCA works with fuzzy logic instead of crisp-logic, used in standard FCA. There are several generalizations of FCA to the fuzzy case [3]. Here the approach of Belohlavek is considered [4]. In fuzzy logic formulas can be valid up to a certain degree. It means that the formula can be completely valid, completely invalid, or between these two states. This fuzziness in fuzzy FCA is represented by a so-called residuated lattice, where the top of the lattice $\top$ corresponds to "completely valid" state of the logic and the bottom $\bot$ corresponds to "completely invalid" state.

Definition 1. *A Residuated Lattice is an algebra* $\boldsymbol{L} = \langle L, \vee, \wedge, \otimes, \rightsquigarrow, 0, 1\rangle$*, where* $\langle L, \vee, \wedge, 0, 1\rangle$ *is a complete lattice;* $\langle L, \otimes, 1\rangle$ *is a commutative monoid, i.e.* $\otimes$ *is commutative, associative, and* $\forall a(a \otimes 1 = 1 \otimes a = a)$*;* $\rightsquigarrow$ *and* $\otimes$ *form an adjiont pair, i.e.,* $a \otimes b \leq c \Leftrightarrow a \leq b \rightsquigarrow c$*.*

For the following, L refers to the set of elements of some residuated lattice and $\mathbf{L}$ for the residuated lattice itself.

An important residuated lattice based on a linearly ordered set is Göodel residuated lattice, which is used in examples of this paper. In Göodel residuated lattices the fuzzzy implication is defied as following:

$$a \rightsquigarrow b = \begin{cases} \top & a \leq b \\ b & a > b \end{cases} \tag{1}$$

In the crisp logic the implication $\top \to \bot$ is not valid, i.e., $\top \to \bot = \bot$, while other three possible implications are valid, i.e., $\top \to \top = \top$, $\bot \to \top = \top$, and $\bot \to \bot = \top$. The formula (1) generalizes this behavior. If the premise is less certain than the conclusion, then the implication is valid ($\top$), otherwise the validity of the implication is equal to the certainty of the conclusion.

In Definition 1 it is required that the fuzzy implication is *adjoint* (related) with an $\otimes$-operation. For Göodel residuated lattices the fuzzy implication is adjoint with $a \otimes b = \min(a, b)$.

A fuzzy dataset is encoded by means of a fuzzy context as defined below.

Definition 2. *A Fuzzy Relation between two sets* X *and* Y *is a function* $I : X \times Y \to L$*, for some residuated lattice* $\boldsymbol{L}$*.*

Definition 3. *A Fuzzy Context is a triple* (X, Y, I) *where* X *is a set of objects,* Y *is a set of attributes,* I *is a fuzzy relation,* $I : X \times Y \to L$*.*

Let us now define what is a fuzzy set, the next building block of fuzzy FCA.

Definition 4. *Given a crisp set* X*, a fuzzy set* A *is a function* $A : X \to L$*, mapping each element of the crisp set to an element of the residuated lattice. A fuzzy set is denoted as* $\{{}^{l_i \in L}/_{x_i \in X}\}$*, where* $\bigcup x_i = X$*, and for simplicity elements* $A(x \in X) = \bot$ *are omitted.*

In the fuzzy case of FCA one also defines Galois connections between a fuzzy set of objects $A : X \to L$ and a fuzzy set of attributes $B : Y \to L$.

Definition 5 (Derivation Operators). *Given a fuzzy context (X, Y, I), a fuzzy set of objects $A : X \to L$, a fuzzy set of attributes $B : Y \to L$, the fuzzy membership for object $x \in X$ and for attribute $y \in Y$ in the corresponding sets $A^{\uparrow}$ and $B^{\downarrow}$ are as follows:*

$$A^{\uparrow}(y) = \bigwedge_{\forall x \in X} (A(x) \rightsquigarrow I(x, y))$$
$$B^{\downarrow}(x) = \bigwedge_{\forall y \in Y} (B(y) \rightsquigarrow I(x, y))$$

Definition 6. *A fuzzy concept is a pair (A, B), where A is a fuzzy set of objects, $A : X \to L$ and B is a fuzzy set of attributes $B : Y \to L$, such that $A^{\uparrow} = B$ and $A = B^{\downarrow}$.*

In particular there is the following fuzzy concept.

$$\big(\{{}^{1.0}/_{t_1}, {}^{1.0}/_{t_2}, {}^{0.286}/_{t_3}\}, \{{}^{0.714}/_{t_1}, {}^{0.714}/_{t_2}, {}^{0.571}/_{t_3}, {}^{0.429}/_{t_4}, {}^{0.429}/_{t_5}\}\big). \qquad (2)$$

The set of fuzzy concepts is ordered such that $(A, B) \leq (X, Y)$ iff $A \subseteq X$ (or dually $B \supseteq Y$) forming a complete lattice, called *fuzzy concept lattice*.

3.2 Pattern Structures

A concept lattice $\mathfrak{L}(G, M, I)$ is constructed from a (binary) formal context (G, M, I) [1]. For non-binary data, such as sequences or graphs, lattices can be constructed in the same way using pattern structures [2].

Definition 7. *A pattern structure $\mathbb{P}$ is a triple $(G, (D, \sqcap), \delta)$, where G, D are sets, called the set of objects and the set of descriptions, and $\delta : G \to D$ maps an object to a description. Respectively, $(D, \sqcap)$ is a meet-semilattice on D w.r.t. $\sqcap$, called similarity operation such that $\delta(G) := \{\delta(g) \mid g \in G\}$ generates a complete subsemilattice $(D_\delta, \sqcap)$ of $(D, \sqcap)$.*

Derivation operator for a pattern structure $(G, (D, \sqcap), \delta)$, relating sets of objects and descriptions, is defined as follows:

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

Given a subset of objects A, $A^{\diamond}$ returns the description which is common to all objects in A. Given a description d, $d^{\diamond}$ is the set of all objects whose description subsumes d. The natural partial order (or subsumption order between descriptions) $\sqsubseteq$ on D is defined w.r.t. the similarity operation $\sqcap$: $c \sqsubseteq d \Leftrightarrow c \sqcap d = c$ (in this case we say that c is subsumed by d).

Definition 8. *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 pattern extent and* d *is called the pattern intent.*

The set of all pattern concepts is partially ordered w.r.t. inclusion of extents or, dually, w.r.t. subsumption of pattern intents within a concept lattice, these two anti-isomorphic orders form a lattice, called pattern lattice.

Let us return to the example in Table 1b. Let us consider a special case of pattern structures, a so-called Minimum Pattern Structure (MnPS), that is close to interval pattern structures [5]. MnPS is based on the minimum of two numbers as the similarity operation rather than on the convex hull of two intervals. We will show that MnPS is well adapted for formalizing fuzzy FCA within the framework of pattern structures.

In Table 1b we have the set G as both, a set of objects and a set of attributes. Let us first consider only one attribute. Then the set of descriptions D is just the interval $[0, 1]$ of real numbers and the similarity operation between two descriptions (numbers) is the minimum. When there are several attributes, the set of descriptions is just an element of $\mathbb{R}^{|N|}$, where $\mathbb{R}$ is the set of real numbers and N is the set of numerical attributes.

In particular, in our example the set of objects is G. The set D of descriptions is $\mathbb{R}^5$, since we have 5 numerical attributes. The mapping function δ is given in Table 1b, e.g., $\delta(t_2) = \langle 0.714, 1, 0.571, 0.429, 0.429\rangle$. The similarity operation is the component-wise minimum, e.g., the similarity between descriptions of t_2 and t_3 is given by

$$\begin{aligned}
&\{t_2\}^\diamond \sqcap \{t_3\}^\diamond = \\
&\quad = \langle 0.714, 1, 0.571, 0.429, 0.429\rangle \sqcap \langle 0.857, 0.571, 1, 0.286, 0.714\rangle = \\
&\quad = \langle \min(0.714, 0.857), \min(1, 0.571), \\
&\qquad \min(0.571, 1), \min(0.429, 0.286), \min(0.429, 0.714)\rangle \\
&\quad = \langle 0.714, 0.571, 0.571, 0.286, 0.429\rangle
\end{aligned}$$

4 From Fuzzy FCA to Pattern Structures with Scaling

Let us now discuss a possible connection between fuzzy FCA and pattern structures. A certain connection was already proposed in [6]. In particular, every *crisply* closed subset of objects is an extent of an interval pattern structure. Here, *crisply closed subset of objects* means that the fuzzy closure of this set contains no additional objects g with a membership degree coinciding with the top of the residuated lattice, i.e., $A(g) = \top$.

Here, we discuss a loss-less scaling from a fuzzy formal context (X, Y, I) to a pattern structure, that allows a more efficient processing than the loss-less scaling to crisp formal context and highlights another connection between fuzzy FCA and pattern structures.

Table 2: Scaling of the fuzzy context from table Table 1b to a number-minimum pattern structure.

	t_1	t_2	t_3	t_4	t_5
$\langle t_1, 1.000\rangle$	1.000	0.714	0.857	0.429	0.429
$\langle t_1, 0.857\rangle$	1.000	0.714	1.000	0.429	0.429
$\langle t_1, 0.714\rangle$	1.000	1.000	1.000	0.429	0.429
$\cdots$					
$\langle t_2, 1.000\rangle$	0.714	1.000	0.571	0.429	0.429
$\cdots$					
$\langle t_2, 0.571\rangle$	1.000	1.000	1.000	0.429	0.429
$\cdots$					
$\langle t_3, 1.000\rangle$	0.857	0.571	1.000	0.286	0.714
$\cdots$					

	t_1	t_2	t_3	t_4	t_5
$\cdots$					
$\langle t_3, 0.429\rangle$	1.000	1.000	1.000	0.286	1.000
$\cdots$					
$\langle t_4, 1.000\rangle$	0.429	0.429	0.286	1.000	0.000
$\cdots$					
$\langle t_4, 0.286\rangle$	1.000	1.000	1.000	1.000	0.000
$\cdots$					
$\langle t_5, 1.000\rangle$	0.429	0.429	0.714	0.000	1.000
$\cdots$					
$\langle t_5, 0.286\rangle$	1.000	1.000	1.000	1.000	0.000

Since pattern structures can deal with any kind of descriptions, they should take into account fuzziness on the intent side. However, for the extent side it is not so straightforward, since pattern structures deal only with crisp sets of objects. Accordingly, we should somehow "scale" object sets, in order to express fuzzy sets of objects.

4.1 On expressing Fuzziness on the Extent Side of Pattern Structures

A natural way is to scale the object set X from a fuzzy context (X, Y, I) by substituting it with the direct product of the crisp set of objects and the residuated lattice (the degrees of confidence), $X \times L$. For every scaled object from this new set, we should compute a description. Let us consider the scaled description for the pair $\langle x, l\rangle$, where $x \in X$ is an object and $l \in L$ is the membership degree of this object. The description of this element should correspond to the description of the fuzzy set $\{{}^{l}/_{x}\}$, since $\langle x, l\rangle$ is "a model of" this fuzzy set.

The derivation operator $\{{}^{l}/_{x}\}^{\uparrow}(y \in Y) = \{{}^{l}/_{x}\}(x) \rightsquigarrow I(x, y) = l \rightsquigarrow I(x, y)$ gives the description of the element $\langle x, l\rangle$ and allows computing the fuzzy relation $\tilde{I}$ between $X \times L$ and Y.

Let us return to our example. Let T be the set of transaction IDs. The scaled fuzzy context is partially shown in Table 2. It consist of $|T| \cdot |L| = 5 \cdot 7 = 35$ objects, 5 attributes and the fuzzy relation between them. Every subset of objects corresponds to a fuzzy set of objects by joining corresponding fuzzy representation for every object. This is made precise in the next subsection.

4.2 Relation between fuzzy and pattern extents and intents

Let (X, Y, I) be a fuzzy context with a residuated lattice $\mathbf{L}$ and (G, D, δ) be a pattern structure, where G is the scaled set of objects $G = X \times L$. Let us formally define the correspondence between fuzzy sets of objects and scaled sets of objects.

Definition 9 (Object sets equivalence). *A fuzzy object set $A : X \to L$ is equivalent to a scaled object set $N \subseteq G$, denoted as $A \sim N$, when*

$$(\forall \langle x, l\rangle \in G)(A(x) \geq l \Leftrightarrow \langle x, l\rangle \in N)$$

Then object sets are equivalent when all scaled objects with membership degree smaller than or equal to $A(x)$ (w.r.t. the residuated lattice) are present in the scaled object set. For example, the fuzzy set $\{^{0.286}/_{t_1}, ^{0.429}/_{t_4}\}$ is equivalent to the scaled set $\{\langle t_1, 0.286\rangle, \langle t_4, 0.429\rangle, \langle t_4, 0.286\rangle\}$[3], where $\langle g, l\rangle \in X \times L$ is an element of the direct product of the set of objects and the residuated lattice.

Given a scaled fuzzy context $(X \times L, Y, \tilde{I})$ we can process it as a minimum pattern structure $(X \times L, D, \delta)$, where $D = L^{|Y|}$ is a tuple of elements from the residuated lattice $\mathbf{L}$ and the semilattice operation is given by the component-wise infimum of $\mathbf{L}$. In particular, we have discussed that for the numerical case, the similarity operation is the component-wise minimum. Indeed, fuzziness on the extent side is expressed by means of scaled object sets, and fuzziness on the intent side is directly processed by the pattern structure. Let us discuss the correspondence between fuzzy intents and patterns.

Definition 10. *A fuzzy attribute set $B : Y \to L$ is equivalent to a pattern $d \in D$, written as $B \sim d$, iff $(\forall y \in Y)(B(y) = d(y))$, where $d(y)$ is the value of the tuple d corresponding to the attribute y.*

A fuzzy attribute set B is equivalent to a pattern d iff for any attribute $y \in Y$, the membership degree $B(y)$ in the fuzzy set is equal to the value in the pattern tuple in the position corresponding to the attribute y, e.g., the pattern $\langle 0.5, 0.7\rangle$ corresponds to the fuzzy set $\{^{0.5}/_{y_1}, ^{0.7}/_{y_2}\}$.

It should be noticed that the definition of equality between fuzzy sets of attributes and patterns is a bijection, while there are scaled sets of objects that have no equivalent fuzzy set of objects. Indeed, there is no equivalent fuzzy set to the scaled set $\{\langle t_1, 0.286\rangle, \langle t_4, 0.429\rangle\}$, since according to Definition 9 all $\langle x, l\rangle$ such that $A(x) \geq l$ should be in this set. And since we have $\langle t_4, 0.429\rangle$ in this set, we should also have $\langle t_4, 0.286\rangle$ in the set. We can notice here that in our particular example the residuated lattice has only the element 0.286 that is smaller than 0.429. By contrast, if we take the real interval $[0, 1]$, then all points smaller than 0.429 should be added to the scaled set.

Let us define equivalence classes of scaled sets of objects in order to have a bijection between the equivalence classes and the fuzzy sets of objects.

Definition 11. *A scaled object set $N \subseteq G$ is* complete *iff a scaled object $\langle x \in X, l \in L\rangle$ belongs to N, then $(\forall l^* \in L, l^* \leq l)\langle x, l^*\rangle \in N$.*

It can be checked that for any scaled object set $N \subseteq G$ there is only one minimal complete superset of N. Let us denote this complete set by $\phi(N)$.

For example, the set $N = \{\langle t_1, 0.286\rangle, \langle t_4, 0.429\rangle\}$ is not complete, since the scaled object $\langle t_4, 0.286\rangle$ is not in N.

By contrast, $N_c = \phi(N) = \{\langle t_1, 0.286\rangle, \langle t_4, 0.429\rangle, \langle t_4, 0.286\rangle\}$ is complete. Moreover, it can be seen that this set is equivalent to $\{^{0.286}/_{t_1}, ^{0.429}/_{t_4}\}$ according to Definition 9. Furthermore, it can be checked, that any complete scaled set of objects is equivalent to a fuzzy set and accordingly the function $\phi(\cdot)$ defines the required equivalence classes.

[3] We notice that $\langle t_4, 0.429\rangle$ and $\langle t_4, 0.286\rangle$ are two different scaled objects.

4.3 Isomorphism of fuzzy and pattern lattices

In this subsection we show that our scaling procedure is correct. And the resulting pattern lattice and the fuzzy lattice are isomorphic. Moreover, the extents and intents of these lattices are connected by means of Definitions 9 and 10. The first lemma (a standard property of residuated lattices) shows that fuzzy implications are related if their premises are comparable.

Lemma 1 *If there are $l_1, l_2, l \in L$ such that $l_1 \leq l_2$ then*

$$l_1 \rightsquigarrow l \geq l_2 \rightsquigarrow l$$

Proof. Let $l_2 \rightsquigarrow l = r$ then according to Def. 1:

$$(\forall f \in \mathbf{L}, f \leq r)(f \otimes l_2 \leq l) \Leftrightarrow (\forall f \leq r)(l_2 \otimes f \leq l)$$
$$\Leftrightarrow (\forall f \leq r)(f \rightsquigarrow l \geq l_2 \geq l_1) \Leftrightarrow (\forall f \leq r)(l_1 \rightsquigarrow l \geq f) \Rightarrow l_1 \rightsquigarrow l \geq r.$$

Let us now show that starting from two (fuzzy and scaled) equivalent sets of objects the resulting descriptions are also equivalent.

Lemma 2 *Given a fuzzy set of objects $A : X \to L$ and a scaled set of objects $N \subseteq G$, such that $A \sim \phi(N)$, we have $A^{\uparrow} \sim N^{\diamond}$.*

Proof. Consider the value of the pattern tuple $N^{\diamond}$ corresponding to an attribute y: $N^{\diamond}(y) = \big(\bigsqcap_{\forall g \in N} \delta(g)\big)(y)$. The semilattice operation of the minimum pattern structure corresponds to the infimum in the residuated lattice:

$$N^{\diamond}(y) = \bigwedge_{\forall \langle x,l\rangle \in N} \tilde{I}(\langle x, l\rangle, y) = \bigwedge_{\forall \langle x,l\rangle \in N} l \rightsquigarrow I(x,y) =$$
$$= \Big(\bigwedge_{\forall x \in X} A(x) \rightsquigarrow I(x,y)\Big) \wedge \Big(\bigwedge_{\forall \langle x,l\rangle \in N : l < A(x)} l \rightsquigarrow I(x,y)\Big) =$$
$$= \underset{\text{Lemma 1}}{} \Big(\bigwedge_{\forall x \in X} A(x) \rightsquigarrow I(x,y)\Big) = A^{\uparrow}(y).$$

Finally let us show, that starting from equivalent fuzzy set of attributes and pattern, the sets of objects given by the derivation operators are also equivalent.

Lemma 3 *Given a fuzzy set of attributes $B : Y \to L$ and a pattern $d \in D$, such that $B \sim d$, we have $B^{\downarrow} \sim d^{\diamond}$.*

Proof. Let us study when object $\langle x \in X, l \in L\rangle$ can be included into $d^{\diamond}$.

$$\langle x \in X, l \in L\rangle \in d^{\diamond} \Leftrightarrow \delta(\langle x, l\rangle) \sqsupseteq d \Leftrightarrow (\forall y \in Y)(\delta(\langle x, l\rangle)(y) \geq d(y))$$
$$\Leftrightarrow (\forall y \in Y)(l \rightsquigarrow I(x,y) \geq d(y))$$
$$\Leftrightarrow (\forall y \in Y)(d(y) \otimes l \leq I(x,y)) \Leftrightarrow (\forall y \in Y)(l \otimes d(y) \leq I(x,y))$$
$$\Leftrightarrow (\forall y \in Y)(d(y) \rightsquigarrow I(x,y) \geq l)$$
$$\Leftrightarrow (\forall y \in Y)(B(y) \rightsquigarrow I(x,y) \geq l)$$
$$\Leftrightarrow l \leq \bigwedge_{\forall y \in Y} B(y) \rightsquigarrow I(x,y)$$
$$\Leftrightarrow l \leq B^{\downarrow}(x)$$

Thus an object $\langle x, l\rangle \in G$ is included in $d^\diamond$ iff $B^\downarrow(x) \geq l$ which is the definition of the equality of a fuzzy set of objects and a scaled set of objects.

Theorem 1 *The fuzzy lattice $\mathfrak{L}_f$ corresponding to the context (X, Y, I) and the pattern lattice $\mathfrak{L}_p$ corresponding to the pattern structure $(G, \underline{D}, \delta)$, where $G = X \times L$, $D = L^{|Y|}$ with component-wise minimum as the semilattice operation, and $\delta(\langle x \in X, l \in L\rangle)(y) = l \rightsquigarrow I(x, y)$ are isomorphic. The extents and intents of the corresponding concepts are equivalent.*

Proof. Let us show, that for any concept in one lattice there is a concept in the other lattice with equivalent extents and intents. Lemmas 2 and 3 are symmetric w.r.t. the type of extents and intents. Accordingly, we can just denote by $\mathfrak{L}_1$ and $\mathfrak{L}_2$ fuzzy and pattern lattices and prove the theorem in both directions. If we take an intent i_1 from $\mathfrak{L}_1$, we can always find an equivalent pattern p (for simplicity, fuzzy set of attributes is also referred as a pattern). Applying appropriate derivation operators to i_1 and p we get equivalent sets of objects according to Lemma 3. Both sets are closed and are extents of $\mathfrak{L}_1$ and $\mathfrak{L}_2$. Applying derivation operators to the extents we get equivalent intents according to Lemma 2. Thus, for any concept of $\mathfrak{L}_1$ there is an equivalent concept in $\mathfrak{L}_2$ and *vice versa*.

4.4 Application of the Theorem

Let us demonstrate how the theorem works in the running example. The proof is based on search of concepts with equivalent extents and intents. Let us find the scaled concept corresponding to the fuzzy concept (2). In the theorem we start from the intent. It can be seen that

$$\{{}^{0.714}/_{t_1}, {}^{0.714}/_{t_2}, {}^{0.571}/_{t_3}, {}^{0.429}/_{t_4}, {}^{0.429}/_{t_5}\} \sim \langle 0.714, 0.714, 0.571, 0.429, 0.429\rangle. \quad (3)$$

For the moment we are not sure that the pattern on the right side is an intent. Accordingly we apply derivation operators to the left and right hand sides and according to Lemma 3 the resulting object sets should be equivalent. Indeed,

$$\{{}^{1}/_{t_1}, {}^{1}/_{t_2}, {}^{0.286}/_{t_3}\} \sim$$
$$\sim \{\langle t_1, 1\rangle, \langle t_1, 0.857\rangle, \ldots, \langle t_1, 0.286\rangle, \langle t_2, 1\rangle, \ldots \langle t_2, 0.286\rangle, \langle t_3, 0.286\rangle\}.$$

On the left side we have the extent of the concept, while on the right side we have a closed scaled set of objects, since the result of the derivation operator is always closed. If we apply the derivation operators to these two sets of objects, we have equivalent patterns according to Lemma 2. In fact we have exactly the patterns from (3). Thus, we have found the scaled concept corresponding to the fuzzy concept. Similarly, we can start from a scaled concept and find the corresponding fuzzy concept.

5 Discussion and Conclusion

In this paper we highlighted the relation between fuzzy FCA and pattern structures. Our result is related to the work of [6]. Indeed, the authors have shown that extents of crisply closed fuzzy concepts are also closed in the interval pattern structure. In our work, we used the Minimum Pattern Structure that can be considered as a projection of the interval pattern structure. Indeed, let us consider the following component-wise projection. If $[a, b]$ is an interval, than the projection $\psi([a, b]) = [a, +\inf]$ changes the IPS to the MnPS. Accordingly the set of extents of the MnPS is the subset of the extents of the IPS. However, in our work we have shown, that the MnPS lattice is isomorphic to a fuzzy lattice under the scaling. It can be seen, that if we do not apply the scaling we generate exactly the lattice of the crisply generated fuzzy concepts. And this set of concepts is the subset of the concepts of the corresponding IPS.

The introduced scaling procedure can be useful, first, for migrating results between pattern structure community and fuzzy FCA community, and, second, for efficient implementation of software dealing with both pattern structures and fuzzy FCA at the same time.

Finally, we notice that such a work naturally raises (as it was already mentioned in [6]) the question of a "two-sided" pattern structure as a generalization of both pattern structures and fuzzy FCA. Some suggestions going in this directions can be found in the work of Soldano et al. [7], where the authors discussed projections applied to the extent side.

References

1. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. 1st edn. Springer (1999)
2. Ganter, B., Kuznetsov, S.O.: Pattern Structures and Their Projections. In Delugach, H.S., Stumme, G., eds.: Concept. Struct. Broadening Base. Volume 2120 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2001) 129–142
3. Belohlavek, R.: What is a fuzzy concept lattice? II. In: Rough Sets, Fuzzy Sets, Data Min. Granul. Comput. Springer Berlin / Heidelberg (2011) 19–26
4. Belohlávek, R.: Lattices generated by binary fuzzy relations (extended abstract). In: Int. Conf. Fuzzy Set Theory Appl. (1998) 11
5. Kaytoue, M., Kuznetsov, S.O., Napoli, A., Duplessis, S.: Mining gene expression data with pattern structures in formal concept analysis. Inf. Sci. (Ny). **181**(10) (2011) 1989 – 2001
6. Pankratieva, V.V., Kuznetsov, S.O.: Relations between proto-fuzzy concepts, crisply generated fuzzy concepts, and interval pattern structures. Fundam. Informaticae **115**(4) (2012) 265–277
7. Soldano, H., Santini, G.: Graph abstraction for closed pattern mining in attributed network. In: Eur. Conf. Artif. Intell. (ECAI), IOS Press. (2014) 849–854

A Tool for Classification of Sequential Data

Giacomo Kahn[1], Yannick Loiseau[1,2], and Olivier Raynaud[1,2]

[1] Université Clermont Auvergne, Université Blaise Pascal, BP 10448, F-63000 CLERMONT-FERRAND, FRANCE
[2] CNRS, UMR 6158, LIMOS, F-63178 AUBIERE, FRANCE
giacomo.kahn@isima.fr, home page: http://fc.isima.fr/~kahngi/

Abstract. Classification of sequential data (data obtained from series of actions in chronological order) has many applications in security, marketing or ergonomy. In this paper, we present a tool for classification of sequential data. We introduce a new clean dataset of web-browsing logs, and study the case of implicit authentification from web-browsing. We then detail more of the functioning of the tool and some of its parameters.

Keywords: Machine Learning, Classification, Web Usage Mining

1 Introduction and related work

Event-related data can have the form of a succession of actions or events in chronological order. Data mining of such data has many applications in fields such as security (intrusion detection [1]), marketing (e.g. navigation in e-commerce hierarchy) or ergonomy (study of succession of actions in work-related applications). Those applications require the search of some meaningful patterns in the data. A pattern is a structure that appears with regularity in the data. It can be an itemset, a sequence, a sub-word, an association rule... In this context, meaningful means maximizing some metric, such as the *support* or the *lift*. Different algorithms exist to mine either of those. An interesting property for patterns is the closure. A pattern p is closed if there is no pattern p', superset of p and $support(p) = support(p')$. Formal Concept Analysis (FCA) is a mathematical framework that deals with closed sets. Many algorithms from FCA allow to enumerate closed sets (in the form of concepts) and there exist a number of interesting metrics based on concept lattices such as stability or robustness of a concept.

The enumeration of these patterns alone is not sufficient in many cases and is only one step of a decision-making process. For example, in a context of security, one might want to find meaningful patterns as the first step of classification or prediction. In marketing, one might use patterns to construct groups of consumers or to find interesting association rules.

In [2,3], the authors introduce a tool for classification in the binary case, based on positive and negative examples in concept lattices. However, by using this binary classifier to the $1-n$ case (n being the number of classes), all anonymous

behaviours will be classified as contradictory. Other works of mining in FCA include the mining of sequences in [4] and of graphs in [5]. In [6], the authors defined *emerging patterns* as patterns appearing frequently in a class, but being hard to find in other classes. Confer [7, 8] for surveys on emerging patterns. An emerging closed-pattern classifier can be described as an extension to the $1 - n$ case of the binary concept lattice classifier, and can be used to predict the class on previously unseen objects. In [9], the authors present another generalisation to n classes of the closed-set based classifier. In particular, the authors introduced the use of the $tf \times idf$ for the selection of the closed patterns.

In this paper, we present a tool for classification of sequential data, based on closed-patterns. This tool implements the classifier presented in [9]. We show some results of our tool on a dataset of web navigation logs from more than 3000 users over a six-month period.

This paper is organised as follow: in section 2 we explain the functioning of the classifier and give more details about the tool and its parameters, in section 3 we show a case study and propose a clean dataset for experimentation, finally we conclude and give some perspectives of our work.

2 Implementation

2.1 General parameters

In this section, we describe the classifier implemented by our tool. The tool includes a whole experimental process, from the building of transactions from raw data to detailed results of classification. We mention some of the parameters accepted by each steps.

Building transactions Our tool allows us to group the data into transactions. The transactions can be of fixed size, or created with respect to a time stamp present in the original data. In our case study, the size is fixed and is equal to 10. The data file from where the transactions are built can be of arbitrary size.

Extraction of own patterns We call own patterns the patterns we believe to be respresentative of each class. For each class, we compute the patterns that verify some property or threshold for a given metric (e.g. support or $tf \times idf$). With some metrics, the space of those patterns is prunable. The number of patterns we want to keep as well as their maximum size is a parameter. The nature of the pattern is also a parameter: as of today, one can choose between closed itemset or sequence. For a given class c, we denote the set of own patterns by P_c.

Profile of a class There exist different ways to compute the profile of a class. In our tool, we chose to define a common vector profile $\mathcal{V} = \bigcup_{c \in C} P_c$ that is the union of all own patterns for all classes. We then compute its numerical components for each classes from either the *support*, the *lift* or the $tf \times idf$. This vector allows us to embed all classes in a common space. This numerical value can be seen as the distance from the origin of the space, in each dimensions of

the vector. For exemple, let α and β two classes. $P_\alpha = \{A, B, C\}$ and $P_\beta = \{C, D, E\}$ then the vector $\mathcal{V} = P_\alpha \cup P_\beta$ will have 5 component (A, B, C, D, E). For each class c, we compute a numerical value k_i^c for each component, giving $\mathcal{V}_\alpha = (k_A^\alpha, k_B^\alpha, k_C^\alpha, 0, 0)$ and $\mathcal{V}_\beta = (0, 0, k_C^\beta, k_D^\beta, k_E^\beta)$.

Profile of an anonymous transaction This step accepts the same parameters as the construction of the profile of a class. We can also choose the number of anonymous transaction that will be submitted to the classifier in the next step. For example, in Fig. 3, the number of anonymous transactions recieved by the classifier goes from 1 to 30.

Identification step The goal is to guess the class corresponding to an anonymous set of transactions. After the computation of a profile for this anonymous set, we compute the nearest neighbor in the common space defined previously. The tool implements different similarity functions: *euclidean* distance, *cosine* similarity, *Kulczynski* measure, and *Dice* similarity. The heuristics gain in efficiency when they are provided with a higher number of anonymous transactions that allows them to construct a finer profile for the anonymous user.

Global parameters Other parameters for experimentations include the number of runs, the verbosity level, the format of the data, the possibility to only compute stats on the data, the use of a fuzzy approach and some parameters for binary classification.

Bayesian Classifier Our tool implements two smoothed Bayesian classifiers: a traditional Bayes classifier and a pattern-based Bayes classifier. Those classifiers allow to compare the results during the experimentations.

2.2 Fuzzy approach

The inclusion of a pattern in a transaction is a binary measure. When working with own patterns of significant size, this strict inclusion will often be false. We consider a fuzzy approach for the support during the computation step of an anonymous profile. We will use a inclusion level instead of a binary measure. The fuzzy support may then be computed as the average of the inclusion levels on the set of transitions.

The fuzzy inclusion level $inc(P, T)$ can be computed as the proportion of the own pattern P included in the transaction T:

$$inc(P, T) = \frac{||P \cap T||}{||P||} \tag{1}$$

To adjust to different cases and be able to represent a wide range of inclusion, from intersection to strict inclusion, we use a transfer function to transform the simple level of inclusion of eq. 1. In the tool, those functions are defined by specifying points on a 2-dimensional space. Two points are fixed, $(0, 0)$ and $(1, 1)$. Some transfer functions are illustrated in Fig. 1.

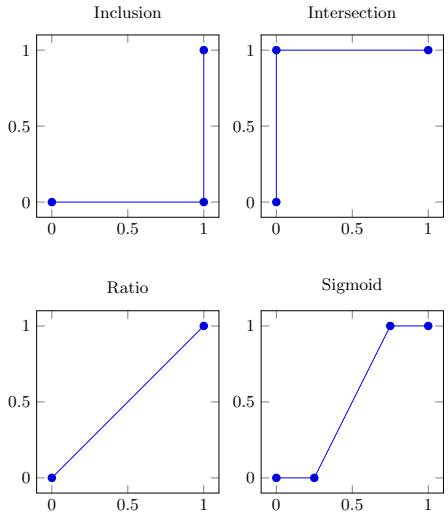


Fig. 1. Different transfer functions

The coordinates of the two points that define the transfer function are configuration parameters. In Fig. 1, the parameters for the inclusion are $[(1,0);(1,0)]$. This is equivalent to the binary measure of inclusion. For the intersection, the parameters are $[(0,1);(0,1)]$. Those parameters mean the measure is equal to one as soon as the intersection is not empty. For the simple ratio or a more sigmoidal function, the parameters are resp. $[(0,0);(1,1)]$ and $[(0.25,0);(0.75,1)]$.

2.3 Configuration file

The parameters are given to the tool by a configuration file in *.yml* format. For the results of our case study, presented in Table 2, the file is presented in Fig. 2.

With this file as argument, the tool will recieve from 1 to 30 anonymous transactions, and run 10 executions. The random seed can be fixed to reproduce experimentations. The data comes from the directory `Data/150users`, and is in *csv* format. The transactions are built of fixed size 10. The identification method is H_1 (closed itemsets and $tf \times idf$ metric), with at most 40 own closed-patterns of maximum size 5. The similarity measure used is Kulczynski. The profiler is the metric used to compute the numerical coordinate of the common vector. When not specified, the method used for inclusion of the pattern is the strict inclusion.

3 Case study

Our case study is about implicit identification in web-browsing. Implicit identification is studied in [10] and in a web-browsing context in [11–14]. The challenge

```
---
name: H1 on csv data
verbose-level: WARNING
number-of-categories: [150]
anonymous-transactions-sizes: [1, 2, 5, 10, 20, 30]
nb-runs: 10
random-seed: 0

data:
        source: Data/150users
        format: csv-normal
        transaction-size: 10

identification-methods:
        - type: Closed
        name: H1
        closed-method: Charm
        weight: TfIdf
        max-pattern-size: 5
        max-own-patterns: 40
        distance: Kulczynski
        profiler: Support
```

Fig. 2. Configuration file for the case study

is to recognise a user amongst n. The classifier has to guess the corresponding user from an anonymous behaviour. If it fails to recognise the declared user, then the identity is not confirmed. In a security context, this situation can lead to restrictions in the system, or to the request of some explicit means of identification. The parameters used in this study are detailed in the configuration file of Fig. 2.

3.1 Data description

Our data comes from Blaise Pascal university proxy servers. It consists of 17×10^6 lines of connection logs from more than 3, 000 users and contains the user ID, the time stamp and a domain name for each line. We applied two types of filters on the domain names: blacklist filters and HTTP-request based filters. We used some lists[3] of domain names to remove all domains regarded as advertising. We also filtered the data by the status code obtained after a simple HTTP request on the domain name. After those steps, we still have 4×10^6 lines. We divide the file between the 3K users to obtain the class files. This dataset is available at http://fc.isima.fr/~kahngi/cez13.zip. The studies were conducted on the 150 users with the higher number of requests.

[3] http://winhelp2002.mvps.org/hosts.htm and https://pgl.yoyo.org/as.

Some information about the data is available in Table 1. The table shows some statistics from before preprocessing and after the filters were applied. $\#Users$ represent the number of users, $\#Sites$ represents the cardinal of the whole set of websites for all users and Avg#lines/user is the average number of line per user. We can see that the number of users decreases because some users did not have a single line after the filters. Roughly 40% of the websites were deleted by the filters, and the average number of lines by user was divided by 5.

Table 1. Descriptive statistics of the dataset

	$\#Users$	$\#Sites$	Avg#lines/user
Raw Data	3388	96184	5082
After preprocessing	3370	57654	1145

3.2 Experimental parameters

Our tool implements several heuristics. H_0 considers frequent 1-patterns with the best support, H_0Lift considers frequent 1-patterns with the best lift, H_1 considers closed k-patterns with the best $tf \times idf$, and B is a smoothed Bayes classifier. The $tf \times idf$ is a metric that comes from information retrieval and text mining. It is the product of term frequency and inverse document frequency. It reflects how discriminating a pattern is for a given class. The experiments in [9] show that $tf \times idf$ produces better results than the lift or the support.

Figure 3 shows the kind of results that can be obtained with our tool. The abscissa is the number of anonymous transactions given to the classifier and the ordinate the accuracy of the different heuristics. The dataset is divided as follows: $\frac{2}{3}$ of learning base for the learning step and $\frac{1}{3}$ for the identification step. The division is random. Each test session consists of multiple runs of both those steps. That allows us to smooth the results by using the average accuracy.

$\mathcal{N}_C$	$\mathcal{A}$	Method	Avg accuracy	Min accuracy	Max accuracy	T_C
150	1	H_1	0.31266	0.30213	0.32118	75.796%
150	2	H_1	0.34378	0.32827	0.35666	93.335%
150	5	H_1	0.46909	0.4505	0.48996	99.676%
150	10	H_1	0.67352	0.63714	0.69905	100%
150	20	H_1	0.87778	0.85111	0.92	100%
150	30	H_1	0.94833	0.92333	0.96333	100%

Table 2. Output of the tool

The table generated by our tool contains 15 columns. Those include the number of classes $\mathcal{N}_C$, the number of anonymous transactions recieved $\mathcal{A}$, the method used, the average accuracy, the average number of transactions successfully and

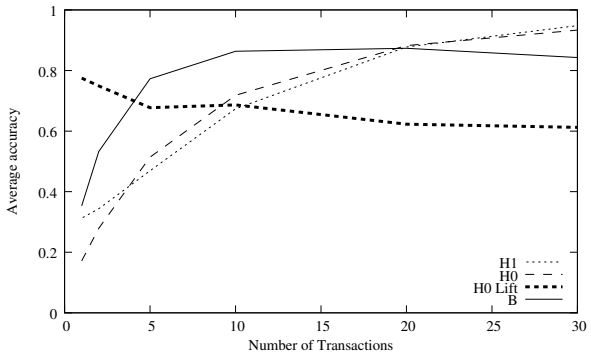


Fig. 3. Average accuracy of the different heuristics as a function of the number of anonymous transactions recieved by the classifier.

not successfully classified, the ratio of classified and not classified tests, and the run time in second. T_C represents the ratio of classified tests. All this information allows the analysis of various aspects of the result. Some are presented in Table 2.

4 Conclusion and perspectives

We presented a tool for classification of sequential data. It includes a lot of features in the construction of the transactions, and different parameters and heuristics for classification. The tool is flexible and adaptable to many contexts of classification and types of data.

The perspectives of our work are to add others means of classification based on other types of patterns (such as closed-sequences, pattern structures, or class association rules), and other types of metrics (for example structural metrics such as stability). We are also considering the use of aggregation functions other than the average for fuzzy support, such as ordered weighted averaging (OWA) operators [15, 16], or some power-means. Moreover, we are considering the integration of different paradigms of user profiles. Another way to construct the profile of a class is using association rules. Class association rules are association rules of the form $A \rightarrow C$ where C is a class and A a subset of items. They are studied in [17]. By attributing scores to the rules and searching for the premisses of the rule in an anonymous transaction, we could classify the anonymous transaction in a given class.

Acknowledgement

This research was partially supported by the European Union's "*Fonds Européen de Développement Régional (FEDER)*" program.

References

1. Azkia, H., Cuppens-Boulahia, N., Cuppens, F., Coatrieux, G.: Log content extraction engine based on ontology for the purpose of a posteriori access control. IJKL **9**(1/2) (2014) 23–42
2. Kuznetsov, S.O.: Complexity of learning in concept lattices from positive and negative examples. Discrete Applied Mathematics **142**(1-3) (2004) 111–125
3. Kuznetsov, S.O.: Machine learning and formal concept analysis. In: Concept Lattices, Second International Conference on Formal Concept Analysis, ICFCA 2004, Sydney, Australia, February 23-26, 2004, Proceedings. (2004) 287–312
4. Buzmakov, A., Egho, E., Jay, N., Kuznetsov, S.O., Napoli, A., Raïssi, C.: On mining complex sequential data by means of FCA and pattern structures. Int. J. General Systems **45**(2) (2016) 135–159
5. Kuznetsov, S.O.: Learning of simple conceptual graphs from positive and negative examples. In: Principles of Data Mining and Knowledge Discovery, Third European Conference, PKDD '99, Prague, Czech Republic, September 15-18, 1999, Proceedings. (1999) 384–391
6. Ramamohanarao, K., Fan, H.: Patterns based classifiers. World Wide Web (1) (2007) 71–83
7. Novak, P.K., Lavrac, N., Webb, G.I.: Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. Journal of Machine Learning Research **10** (2009) 377–403
8. García-Borroto, M., Trinidad, J.F.M., Carrasco-Ochoa, J.A.: A survey of emerging patterns for supervised classification. Artif. Intell. Rev. **42**(4) (2014) 705–721
9. Coupelon, O., Dia, D., Labernia, F., Loiseau, Y., Raynaud, O.: Using closed itemsets for implicit user authentication in web browsing. In: Proceedings of the Eleventh International Conference on Concept Lattices and Their Applications, Košice, Slovakia, October 7-10, 2014. (2014) 131–144
10. Shi, E., Niu, Y., Jakobsson, M., Chow, R.: Implicit authentication through learning user behavior. In: Information Security - 13th International Conference, ISC 2010, Boca Raton, FL, USA, October 25-28, 2010, Revised Selected Papers. (2010) 99–113
11. Kumar, R., Tomkins, A.: A characterization of online browsing behavior. In: Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010. (2010) 561–570
12. Yang, Y.C.: Web user behavioral profiling for user identification. Decision Support Systems **49**(3) (2010) 261–271
13. Goel, S., Hofman, J.M., Sirer, M.I.: Who does what on the web: A large-scale study of browsing behavior. In: Proceedings of the Sixth International Conference on Weblogs and Social Media, Dublin, Ireland, June 4-7, 2012. (2012)
14. Abramson, M., Aha, D.W.: User authentication from web browsing behavior. In: Proceedings of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2013, St. Pete Beach, Florida. May 22-24, 2013. (2013)
15. Yager, R.R.: Families of OWA operators. **59** (1993) 125–148
16. Yager, R.R.: Constraint satisfaction using soft quantifiers. International Journal of Intelligent Systems in Accounting and Finance Management **12**(3) (2004) 177–186
17. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98), New York City, New York, USA, August 27-31, 1998. (1998) 80–86

Interval Pattern Concept Lattice as a Classifier Ensemble

Yury Kashnitsky, Sergei O. Kuznetsov

National Research University Higher School of Economics, Moscow, Russia
{ykashnitsky,skuznetsov}@hse.ru

Abstract. Decision tree learning is one of the most popular classification techniques. However, by its nature it is a greedy approach to finding a classification hypothesis that optimizes some information-based criterion. It is very fast but may lead to finding suboptimal classification hypotheses. Moreover, in spite of decision trees being easily interpretable, ensembles of trees (random forests and gradient-boosted trees) are not, which is crucial in some domains, like medical diagnostics or bank credit scoring. In case of such "small, but important-data" problems one is not obliged to perform a greedy search for classification hypotheses, and therefore alternatives to decision tree learning techniques may be considered. In this paper, we propose an FCA-based classification technique where each test instance is classified with a set of the best (in terms of some information-based criterion) classification rules. In a set of benchmarking experiments, the proposed strategy is compared with decision tree and nearest neighbor learning.

Keywords: machine learning, classification, decision tree learning, formal concept analysis, pattern structures

1 Introduction

The classification task in machine learning aims to use some historical data (a training set) to predict unknown discrete variables in unknown data (a test set). While there are dozens of popular methods for solving the classification problem, usually there is an accuracy-interpretability trade-off when choosing a method for a particular task. Neural networks, random forests and ensemble techniques (boosting, bagging, stacking etc.) are known to outperform simple methods in difficult tasks. Kaggle competitions also bear testimony for that – usually, winners resort to ensemble techniques, mainly to gradient boosting [13]. The mentioned algorithms are widely spread in those application scenarios where classification performance is the main objective. In Optical Character Recognition, voice recognition, information retrieval and many other tasks typically we are satisfied with a trained model if it has a low generalization error.

However, in lots of applications we need a model to be interpretable as well as accurate. Some classification rules, built from data and examined by experts, may be justified or proved. In medical diagnostics, when making highly responsible

decisions (e.g., predicting whether a patient has cancer), experts prefer to extract readable rules from a machine learning model in order to "understand" it and justify the decision. In credit scoring, for instance, applying ensemble techniques can be very effective, but the model is often obliged to have "sound business logic", that is, to be interpretable [10].

In what follows, we introduce some notions from Formal Concept Analysis (FCA) [5] and provide a technique to express decision tree learning in terms of a search for a hypothesis in a concept lattice (section 3). In section 4, we propose an algorithm which by its design guarantees that each test object is classified with a better (in terms of some criterion such as information gain or Gini impurity) rule than in case of applying a decision tree. Finally, we discuss the results of the experiments with several popular datasets (section 5), make conclusions and directions of further work on developing the performed ideas.

2 Pattern Structures and Projections

Pattern structures are natural extension of Formal Concept Analysis to objects with arbitrary partially-ordered descriptions [4].

Definition 1. *Let G be a set (of objects), let $(D, \sqcap)$ be a meet-semi-lattice (of all possible object descriptions) and let $\delta : G \to D$ be a mapping between objects and descriptions. Set $\delta(G) := \{\delta(g) | g \in G\}$ generates a complete subsemilattice $(D_\delta, \sqcap)$ of $(D, \sqcap)$, if every subset X of $\delta(G)$ has infimum $\sqcap X$ in $(D, \sqcap)$.* ***Pattern structure*** *is a triple $(G, \underline{D}, \delta)$, where $\underline{D} = (D, \sqcap)$, provided that the set $\delta(G) := \{\delta(g) \mid g \in G\}$ generates a complete subsemilattice $(D_\delta, \sqcap)$ [4, 9].*

Definition 2. ***Patterns*** *are elements of D. Patterns are naturally ordered by subsumption relation $\sqsubseteq$: given $c, d \in D$ one has $c \sqsubseteq d \Leftrightarrow c \sqcap d = c$. Operation $\sqcap$ is also called a* ***similarity*** *operation. A pattern structure $(G, \underline{D}, \delta)$ gives rise to the following* ***derivation operators*** *$(\cdot)^\diamond$:*

$$A^\diamond = \bigsqcap_{g \in A} \delta(g) \qquad \text{for } A \in G,$$
$$d^\diamond = \{g \in G \mid d \sqsubseteq \delta(g)\} \qquad \text{for } d \in (D, \sqcap).$$

Pairs (A, d) satisfying $A \subseteq G$, $d \in \underline{D}$, $A^\diamond = d$, and $A = d^\diamond$ are called ***pattern concepts*** *of $(G, \underline{D}, \delta)$.*

As in classical FCA, pattern concepts form a pattern concept lattice. In case it is too computationally demanding to build the whole lattice, projections are used to simplify object descriptions and boost the formation of a pattern concept lattice.

Definition 3. *A projection [4] of a semilattice $(D, \sqcap)$ is a kernel function $\psi : D \to D$, i.e $\forall x, y \in D$:*

- *$x \sqsubseteq y \Rightarrow \psi(x) \sqsubseteq \psi(y)$ (monotonicity)*
- *$\psi(x) \sqsubseteq x$ (contractivity)*
- *$\psi(\psi(x)) = \psi(x)$ (idempotence)*

3 Interval Pattern Structure Projections

The theoretical part of the proposed approach is based on Formal Concept Analysis and pattern structures, in particular, on Interval Pattern Structures [6] that provide a way to apply FCA techniques to data with numeric attributes. Unfortunately, the size of the concept lattice is usually too large to be used efficiently in learning [11]. Hence, we introduce a so-called discretizing projection on interval pattern structures which helps to build more general object descriptions based on numeric attributes.

Definition 4. *Let $(G, (D, \sqcap), \delta)$ be an interval pattern structure.*
Let $T_i = \{\tau_{i1}, \ldots, \tau_{it_i}\}, i = 1, \ldots, m$ be m sets of real numbers where m is a cardinality of each $d \in D$. Then, $\psi(\langle [a_i, b_i] \rangle_{i \in [1,m]}) =$
$\langle [max\{\tau \mid \tau \in T_i \cup \{-\infty, +\infty\}, \tau \leq a_i\}, min\{\tau \mid \tau \in T_i \cup \{-\infty, +\infty\}, \tau \geq b_i\}] \rangle$
*is called a **discretizing projection** of a semilattice $(D, \sqcap)$.*

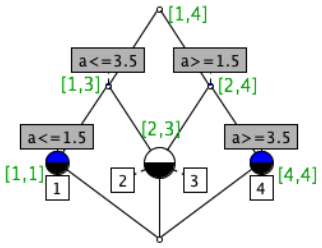
The discretizing projection, as defined in Def. 4, is a projection according to the definition Def. 3.

Example 1. Consider a toy dataset with only 4 objects and 1 numeric attribute as shown in Fig. 1 (left). In order to apply decision tree learning for some classification task with this dataset, one would apply some discretization method to produce binary attributes from attribute a. Consider the discretization shown in Fig. 1 (middle). The corresponding concept lattice is shown in the same figure on the right-hand side.

Fig. 1. A toy many-valued context, it's discretization and the corresponding concept lattice.

	a
1	1
2	2
3	3
4	4

	$a \leq 1.5$	$a \leq 3.5$	$a \geq 1.5$	$a \geq 3.5$
1	×	×		
2		×	×	
3		×	×	
4			×	×



$\psi([a, b]) = [max\{\tau \mid \tau \in T^+, \tau \leq a\}, min\{\tau \mid \tau \in T^+, \tau \geq b\}]$ with $T^+ = \{-\infty, 1.5, 3.5, +\infty\}$ is a projection of the semilattice built for a context that arises from the interordinal scaling of the initial many-valued numerical context. Address to [8] for more details on the link between interordinal scaling

and interval pattern structures. The pattern concept lattice corresponding to the discretizing projection $\psi([a, b])$ is isomorphic to the concept lattice of the discretized context shown in Fig. 1 (middle).

Introducing a discretizing projection is a general way to express any discretizing procedure (essential part of decision tree learning algorithms) in terms of FCA.

4 Learning with Pattern Concept Lattices

For classification tasks with complex data we propose Algorithm 1. The main idea is to find the classification rule for each test instance that maximizes some information criterion (Gini index, pairwise mutual information etc.). In case of interval pattern structures, by its design, the algorithm guarantees to classify each test instance with at least as good rule (in terms of an information criterion) as a decision tree. We apply a modification of the CloseByOne algorithm [7] to build all pattern concepts – the search space for classification rules.

Let $PS_{train} = (G_{train}, ((D, \sqcap), c_{train}), \delta_{train})$ and $PS_{test} = (G_{test}, (D, \sqcap), \delta_{test})$ be two pattern structures corresponding to a train and a test set in a classification task. Let $CbO_{PS}(PS, min\_supp)$ be the algorithm used to find all pattern concepts of a pattern structure PS with support greater or equal to min_supp. Let $inf : D \cup c_{train} \to \mathbb{R}$ be an information criterion used to rate classification rules (we use Gini impurity by default). Finally, let min_supp and n_rules be the parameters of the algorithm (the minimal support of each classification rule's premise and the number of rules to be used for prediction of each test instance's class attribute).

With this designations, the main steps of the proposed algorithm are the following:

1. Initialize a list of predicted labels for test instances c_{test} and a dictionary of classification rules r_{test} for each test instance.
2. Calculate the proportion of positive objects in the training set: $f_{pos} = \frac{|c'_{train}|}{|G_{train}|}$
3. With the CbO_{PS} algorithm, find $\mathcal{S}$ – a dictionary of all pattern concepts (with support greater or equal to min_supp) of a pattern structure PS_{train} Meanwhile, calculate the value of the criterion inf (values in the dictionary $\mathcal{S}$) for each concept intent (keys in the dictionary $\mathcal{S}$).
4. Sort $\mathcal{S}$ by its values.
5. For each test instance $g_t \in G_{test}$:
   - Find first n_{rules} concept intents from $\mathcal{S}$ such that $(A_i, d_i) \in \mathcal{S}, g_t^{\diamond} \sqsubseteq d_i, i = 1, \ldots, n_{rules}$
   - For each "top-ranked" concept intent d_i determine c_i – the proportion of positive objects among $d_i^{\diamond}$: $f_i^+ = \frac{|d_i^{\diamond} \cap c'_{train}|}{|d_i^{\diamond}|}$.
   - Thus, form $\{d_i \to f_i^+\}_{i \in [1, n\_rules]}$ – a set of classification rules for g_t. Set $r_{test}[t]$ be equal to this set of rules.

– Predict the value of the class attribute for g_t as an indicator of the average antecedent of $r_{test}[t]$ being greater or equal to the proportion of positive objects in the training set:

$$c_{test}[i] = [\sum_{i=1}^{n\_rules} f_i^+ \geq f_{pos} * n\_rules]$$

Algorithm 1 Concept Lattice-Based Rule-learner (CoLiBRi)

Input: $PS_{train} = (G_{train}, ((D, \sqcap), c_{train}), \delta_{train})$
$PS_{test} = (G_{test}, (D, \sqcap), \delta_{test})$
$min\_supp \in \mathbb{R}^+$, $n_{rules} \in \mathbb{N}$;
$CbO_{PS}(PS, min\_supp) : PS \to \mathcal{S}$;
$inf : D \times c_{train} \to \mathbb{R}$;
$sort(\mathcal{S}, inf) : \mathcal{S} \to \mathcal{S}$

Output: c_{test}, r_{test}

$c_{test} = \emptyset, r_{test} = \emptyset$
$f_{pos} = \frac{|c'_{train}|}{|G_{train}|}$
$\mathcal{S} = \{(A, d) : inf(d, c_{train}) \mid A \subseteq G_{train}, d \in D, A^\diamond = d, d^\diamond = A, |A| \geq min\_supp\} = CbO_{PS}(PS_{train}, min\_supp)$
$\mathcal{S} = sort(\mathcal{S}, inf)$
for $g_t \in G_{test}$ **do**
  $\{d_i\}_{i \in [1, n_{rules}]} = \{d \mid (A, d) \in \mathcal{S}, g_t^\diamond \sqsubseteq d\}$
  $f_i^+ = \frac{|d_i^\diamond \cap c'_{train}|}{|d_i^\diamond|}$
  $r_{test}[i] = \{d_i \to f_i^+\}_{i \in [1, n_{rules}]}$
  $c_{test}[i] = [\sum_{i=1}^{n\_rules} f_i^+ \geq f_{pos} * n\_rules]$
end for

In case of a classification task with numeric attributes we apply the same Algorithm 1 for interval pattern structures. To make it tractable, we apply it to projections $\psi(PS_{train})$ and $\psi(PS_{test})$ of a training and a test interval pattern structure. Here $\psi(PS)$, is a discretizing pattern structure projection as defined in Def. 4.

5 Experiments

We compare the proposed classification algorithm (denoted as "CoLiBRi" for "Concept Lattice-Based Rule-learner") with Scikit-learn [12] implementations of CART [2], Random Forest [1] and kNN on several datasets from the UCI machine learning repository.[1]

[1] http://repository.seasr.org/Datasets/UCI/csv/

dataset	DT acc	RF acc	kNN acc	CoLiBRi acc	DT F1	RF F1	kNN F1	CoLiBRi F1
audiology	0.75	**0.8**	0.63	0.79*	0.71	**0.74**	0.58	**0.74**
balance-scale	0.63	0.66	**0.76**	0.65	0.58	0.63	**0.75**	0.61
breast_cancer	0.7	0.74	0.73	**0.76**	**0.45**	0.42	0.38	0.44*
car	0.75	0.78*	0.71	**0.79**	0.75	**0.76**	0.71	**0.76**
hayses-roth	0.84*	0.83*	0.49	**0.86**	0.84*	0.82	0.49	**0.85**
lymph	0.8	0.83	**0.86**	0.83	0.77	**0.85**	0.84*	0.84*
mol_bio_prom	0.78	**0.83**	**0.83**	0.82*	0.78	**0.84**	0.8	0.83*
nursery	0.64	0.65	**0.72**	0.65	0.62	0.62	**0.7**	0.62
primary_tumor	0.41	**0.46**	0.41	0.45*	0.37	**0.41**	0.37	0.4*
solar_flare	0.7*	0.7*	0.63	**0.72**	0.67	0.69*	0.6	**0.71**
soybean	0.91*	0.91*	**0.92**	0.91*	0.91*	**0.93**	0.92*	0.91*
spect_train	0.61	**0.69**	0.68*	0.7	0.34	0.36*	0.23	**0.38**
tic-tac-toe	0.79	0.79	**0.85**	0.78	0.82	0.86	**0.89**	0.85

Table 1. Accuracies and F1-scores in classification experiments with the UCI machine learning datasets. "DT acc" and "DT F1" stand for average 5-run 5-fold CV accuracy and F1 score of the CART algorithm, ..., "CoLiBRi F1" stands for average 5-run 5-fold CV F1 score of the proposed CoLiBRi algorithm.

We used Gini impurity as a criterion for rule selection and MDL [3] for continuous feature discretization. CART, Random Forest and kNN parameters ($min\_samples\_leaf \in [1, 10]$ for tree-based algorithms and $k \in \{1, 2, 5, 15, 30, 50\}$ for kNN) were chosen in stratified 5-fold cross-validation. We built 10 trees for each instance of Random Forest classifier.

Parameter min_supp for "CoLiBRi" was taken equal to CART's $min\_samples\_leaf$ for each dataset. We used $n = 10$ classification rules to vote for a test instance label. The described algorithms were implemented in Python 2.7.3 and run on a 4-CPU machine with 4 GB RAM.

The results are presented in Table 1. Each entry stands for the average metric (accuracy or F1-score) in 5 runs of 5-fold cross-validation. In the table, the algorithm with the best performance on each metric is boldfaced. Other algorithm's whose performance is not statistically distinguishable from the best algorithm at p = 0.05 using paired t-tests on the 5 runs are *'ed. The best parameters for each algorithm are mentioned in Table 2.

As it can be seen, the proposed approach performs better than CART and is statistically indistinguishable from RF on most of the datasets. Surprising enough, kNN seems to be the best-performer (on average over all datasets) in terms of accuracy but not F1-score.

Conclusions and further work

In this paper, we have shown how searching for classification hypotheses in a formal concept lattice may yield accurate results while providing interpretable classification rules.

Further we plan to test the proposed strategy in classification tasks such as predicting biological activity (toxicology, mutagenicity, etc.) and telecom client

dataset	DT min_samples_leaf	RF min_samples_leaf	kNN k
audiology	1	1	2
balance-scale	6	1	50
breast cancer	4	3	5
car	3	2	5
hayses-roth	3	1	15
lymph	1	1	5
mol-bio-prom	3	3	5
nursery	3	4	50
primary tumor	4	4	30
solar flare	3	1	30
soybean	1	1	2
spect train	9	5	10
tic-tac-toe	10	3	10

Table 2. Best parameters in classification experiments with the UCI machine learning datasets. CoLiBRi's *min_supp* is taken equal to CART's *min_samples_leaf* for each dataset.

satisfaction where objects have complex descriptions (graphs and sequences correspondingly).

We also plan to introduce some randomization in mining rules for each test instance (as it is done with random forests) in order to further improve the classification quality.

References

[1] L. Breiman. "Random Forests". In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32. ISSN: 0885-6125.

[2] L. Breiman et al. *Classification and Regression Trees*. The Wadsworth and Brooks-Cole statistics-probability series. Taylor & Francis, 1984.

[3] U. M. Fayyad and K. B. Irani. "Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning." In: *IJCAI*. 1993, pp. 1022–1029.

[4] B. Ganter and S. O. Kuznetsov. "Pattern Structures and Their Projections". In: *Conceptual Structures: Broadening the Base*. Ed. by Harry Delugach and Gerd Stumme. Vol. 2120. Lecture Notes in Computer Science. Berlin/Heidelberg: Springer, 2001, pp. 129–142.

[5] B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. 1st. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1997.

[6] M. Kaytoue et al. "Mining gene expression data with pattern structures in formal concept analysis". en. In: *Information Sciences* 181.10 (May 2011), pp. 1989–2001.

[7] S. O. Kuznetsov. "A fast algorithm for computing all intersections of objects from an arbitrary semilattice". In: *Nauchno-Tekhnicheskaya Informatsiya Seriya 2-Informatsionnye Protsessy I Sistemy* 1 (1993), pp. 17–20.

[8] S. O. Kuznetsov. "Fitting Pattern Structures to Knowledge Discovery in Big Data". In: *Formal Concept Analysis, 11th International Conference, ICFCA 2013, Dresden, Germany, May 21-24, 2013. Proceedings*. Vol. 7880. Lecture Notes in Computer Science. Springer, 2013, pp. 254–266.

[9] S. O. Kuznetsov. "Scalable Knowledge Discovery in Complex Data with Pattern Structures". In: *PReMI*. Ed. by Pradipta Maji et al. Vol. 8251. Lecture Notes in Computer Science. Springer, 2013, pp. 30–39.

[10] X. Li and Y. Zhong. "An Overview of Personal Credit Scoring: Techniques and Future Work". In: *International Journal of Intelligence Science* 2.4A (2012), pp. 181–189.

[11] A. Masyutin, Y. Kashnitsky, and S. O. Kuznetsov. "Lazy classification with interval pattern structures: Application to credit scoring". In: *Proceedings of the 4th International Workshop "What can FCA do for Artificial Intelligence?", FCA4AI 2015, co-located with the International Joint Conference on Artificial Intelligence (IJCAI 2015), Buenos Aires, Argentina, July 25, 2015*. Vol. 1430. 2015, pp. 43–54.

[12] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[13] G. Tsoumakas et al. "WISE 2014 Challenge: Multi-label Classification of Print Media Articles to Topics". eng. In: *15th International Conference on Web Information Systems Engineering (WISE 2014). Proceedings Part II*. Vol. 8787. Lecture Notes in Computer Science. Springer, Oct. 2014, pp. 541–548.

New taxonomy of classification methods based on Formal Concepts Analysis

Marwa Trabelsi[1], Nida Meddouri[1] and Mondher Maddouri[2]

[1] Laboratory of computing, Programming, Algorithmic and Heuristic - LIPAH, Faculty of Science of Tunis - FST, El Manar University, Tunis, Tunisie
trabelsimarou@live.com nida.meddouri@gmail.com
[2] Management college, Jeddah university, Kingdom of Saoudi Arabia
maddourimondher@yahoo.fr

Abstract Data mining is an essential step in knowledge extraction from data. Various approaches have been proposed in supervised classification of data, among them approaches based on Formal Concept Analysis. In this paper we present a new taxonomy of classification methods based on Formal Concept Analysis.

Keywords: Data Mining, Supervised Classification, Formal Concept Analysis

1 Introduction

The volume of data stored in the web has undergone a significant and a continuous evolution. Several efforts focused therefore on knowledge retrieval (extraction) from data. In [4], the knowledge extraction from data is defined as an acquisition of new knowledge, which is potentially useful, from the hidden facts within great amounts of data. One of the main processes of the knowledge extraction is based on data mining. This operation collects several tasks such as prediction, clustering and supervised classification. The latter can be performed by methods based on neural networks, decision trees, nearest neighbor, support vector machines or Formal Concept Analysis (FCA). Several reasons lead to use the classification method based on FCA, among them the ability of formal concepts to process a significant quantities of data and to simplify the prediction of classes [17].

Supervised classification based on FCA consists in building functions or models called classifiers from data to predict classes for future data. It aims at extracting the classification rules based on the concepts generated previously from data [19,21].

The whole process is performed in two main steps: a **training step** where a classifier is built to describe a predetermined set of object classes from a training set. A **classification step** where trained classifiers are used to assign a class to each new object. In this paper, we focus only on supervised classification based on FCA. Particularly, we propose a new taxonomy of classification methods based on FCA. The paper is organized as follows : we present basic notions related to FCA in Sect. 2. Sect. 3 describes new taxonomy of supervised classification methods. In Sect. 4, we make a comparative study of such methods.

2 Formal Concept Analysis

Originally, FCA is conceived by R.Wille as a "restructuring" of abstract lattice theory, i.e., making it closer to the needs of other branches of science. FCA represents a theoretical background for many applications in computer science. It covers various fields such as linguistics, information retrieval, text mining, knowledge representation and more recently knowledge extraction.

A formal context is represented as a triplet $K = (G, M, I)$ witch relies a finite set of objects G to a finite set of attributes M using binary relation I ($I \subseteq G \times M$) [7]. A formal context can be illustrated by a two-dimensional table where objects are presented by lines and attributes are presented by columns. All possible formal concepts are extracted from a formal context $K = (G, M, I)$. In fact, a formal concept is represented by a pair (A, B) such that $A \subseteq G$ and $B \subseteq M$, $A' = B$ and $B' = A$ where $A' = \{\text{m} \in \text{M} \mid (\text{g,m}) \in \text{I} \; \forall \; \text{g} \in \text{A}\}$, i.e. the set of attributes common to all objects in A, and $B' = \{\text{g} \in \text{G} \mid (\text{g,m}) \in \text{I} \; \forall \; \text{m} \in \text{B}\}$, i.e. the set of objects which have all attributes in B. A and B are called, respectively, extent and intent of the concept (A, B) [7]. The set of all concepts can be organized as a complete lattice of formal concepts, called Concept Lattice [7].

3 Supervised classification based on Formal Concept Analysis

Classification methods based on FCA uses either what we call exhaustive or combinatory approaches. In this section, we describe in detail the approaches and we give an overview of existing methods for each approach.

3.1 Exhaustive classification approach

Exhaustive methods are characterized commonly by the use of one single classifier. However, they vary between them according to the criteria used on concepts selection and the size of lattices outlining formal concepts. We distinguish overall methods based on complete lattices and others based on sublattices [21].

Among the works that have focused on the classification using a complete lattice as research space, we can cite GRAND [20], RULEARNER [22], GALOIS [2], CBALattice [8], NAVIGALA [24], HMCS-FCA-SC [5] and SPFC [9].

These methods carried out the validation of the characteristics associated to each concepts in the lattices level by level. The navigation in the lattice of concepts starts from the minimal concept where all the concepts of the lattice are considered as candidates without having an idea on their validity.

GRAND[3] and GALOIS are the first methods which use complete lattices of formal concepts. They build a complete lattice using an incremental algorithm and it updates the lattices by adding new nodes and removing redundant connections. Then, GRAND chooses the more specific rules to be applied for each object [20].

[3] Graph-based induction

GALOIS also builds a complete lattice in an incremental and ascending way. In the classification step, the system computes the similarity between new object and each concept. The similarity between an object and a concept is the number of common attributes verified by the object [2].

Other classification systems based on FCA are developed following GRAND such as RULEARNER and NAVIGALA. NAVIGALA[4] uses an object context described by numerical vectors of fixed size. These vectors are stored in a discrete table which then becomes binary [24]. RULEARNER, in a similar setting, uses a complete concept lattice as a research space in classification rules. During the classification, it uses majority voting to determine the classes of new objects [22].

CBALattice builds a complete lattice and applies association rules to extract classification rules. The method is incremental and progressive. Every increase in the number of objects, attributes, and classes can be handled efficiently [8].

HMCS-FCA-SC[5] also uses a complete lattice and existing information in formal concepts. However, unlike CBALattice, it aims to create a model of hierarchical classification. In addition of that, HMCS-FCA-SC employs a cosine measure [6] between new example and the selected concepts for classification [5].

After the construction of the lattice, SPFC[7] assigns to each concept a score which indicates the relevance degree of the concepts. Then it looks for neighbors of relevant concepts. The unknown objects will be classified in the classes of their neighbors [9].

The limit of the methods based on complete lattice consists in the exponential complexity of their training algorithms in terms of time and memory resources.

In order to overcome this problem, other efforts such as LEGAL [15], CIBLE [19], CLNN & CLNB [25], IPR [16], CLANN [23] and CITREC [3], MCSD-FCA-PS [1] use sublattices instead of complete lattice as search space. A sublattice of concepts is a mathematical structure which represents a part of the concept lattice in a selective way [7]. LEGAL[8] is a method which relies on several training parameters to build a sublattice. During the learning step, it builds an ordered set of concepts based on the class of each instance. The positive and negative instances are the instances labeled by a positive or negative class in the formal context. During classification Legal applies the majority vote [15].

CIBLE[9] operates in two succeeding steps: it starts with the construction of a sublattice from a binary context then it uses a similarity measure for the classification of new objects [19].

CLNN & CLNB[10] methods build a sublattice in a decreasing way. They incorporate then respectively a Naive Bayes classifier and a Nearest Neighbors

[4] Navigation into Galois Lattice
[5] Hierarchical Multi-label Classifier System - FCA with Similarity Cosine
[6] the similarity between two vectors with n dimensions by determining the cosine of the angle between them
[7] Classification by Selecting Plausible Formal Concepts in a Concept Lattice
[8] Learning with Galois Lattice
[9] Concept Induction Based Learning
[10] Concept Lattices Nearest Neighbors and Concept Lattices Naive Bayes

classifier in each node of the sublattice built. CLNN & CLNB use the same technique of vote (majority vote) in the classification step [25].

Another classification system CITREC which uses sublattices is proposed in [3]. CITREC builds the lattice starting from a reduced context containing only a representative object of each class [3]. In the classification step CITREC uses the majority vote in the same way as the methods CLNN & CLNB.

On the other side, CLANN[11] starts by building a sublattice in the training step and processes data which have only two classes. It uses then straightforward sublattice to build a neural networks which makes the classification [23].

MCSD-FCA-PS[12] has the distinction of processing sequential data. Complex sequential data are mapped onto pattern structures whose projections are used to build a pattern concept lattice. In fact, pattern structures are a generalization of formal concept analysis designed to deal with complex objects descriptions when an object is not associated to a set of attributes. MCSD-FCA-PS select relevant patterns that can be used for the classification step [1,13,12].

IPR[13] is a method which introduces the coverage of concepts. It selects from the lattice all the relevant concepts which can help to better classification. The choice of relevant concepts is based on greedy algorithm [16]. Then to classify each new object, IPR searches rules with a premise that coincides with the object attribute. The applied rules are the most weighted rules for the involved object.

The classification methods based on sublattices proceeds in the same way as methods based on complete lattices. However, using sublattices can reduce the substantial number of generated rules and keep the most relevant among them. Such proceeding leads to reduce significantly the training time, however it causes a loss of information.

Several drawbacks are observed in exhaustive methods described above. In fact, besides to the high complexity, using one single weak classifier may not be the best solution for classification. The use of many classifiers can give better results. Subsequently, the researchers move towards the integration and the use of combinatory classification methods which are based on the ensemble methods in order to improve other operations of a low single classifier (a single learner).

3.2 Combinatory classification approach

Unlike exhaustive methods which use one classifier, combinatory methods employs many classifiers which are then combined by the techniques of votes.

In this context, various methods have been proposed among them there are methods based on sequential training like BFC [18], BNC [17] and methods based on parallel training such as DNC [17], FPS-FCA [14] and RMCS [10].

The sequential training consists in generating classifiers sequentially. In other words, a classifier is generated only after the generation of its predecessor.

[11] Concept Lattice-based Artificial Neural Network
[12] Mining Complex Sequential Data by means of FCA and Pattern Structures
[13] Induction of Production Rules

For example, BFC[14] builds from a formal context a cover formed only by relevant concepts. The latter is based on boosting which is an adaptive approach based on the use of several classifiers of the same model [11]. These classifiers use voting techniques in order to classify correctly the objects wrongly classified. The idea of BFC consists in assigning, initially, equal weights for the training examples among them subset is selected randomly. At this point, a relevant concept is extracted from a subset by selecting the attribute which minimizes Shannon entropy[15]. BFC generates then a classification rule deducted from the relevant concept (extracted from subset) and updates the weights of training examples. This process is rebuilt recursively to produce the final classifier [18].

The method BNC[16] proceeds similarly to BFC method in generating classifiers and processing training data. However, unlike BFC which makes the processing of binary data, BNC handle nominal data in order to avoid the loss of information following the binary data representation [17].

The parallel training is based on Dagging [11], it consists in dividing data set into many groups. Classifiers are generated from the groups. DNC[17] method handles nominal data. DNC algorithm proceeds as follow: a random data is run in order to create a disjoint groups containing stratified data. A classifier of nominal concept [17] is then created for each group. Finally, the method defines as output a combination of classifiers made by a vote technique [17].

FPS-FCA[18] is a symbolic classification method based on Pattern structures and FCA, witch deal with complex input objects like logical formulas, graphs, strings and tuples of numerical. In presence of large data sets, it offers natural approximation tools (projections) and achieves classification in a parallel way by dividing the initial data set [14,13,6,12].

RMCS[19] generates a set of classifiers parallely. It allows to create a classifier based on its neighbors. The classifier realizes correctly the object classification when it is classified correctly within its neighbors. RMCS starts with the construction of a classification table from a formal context. In this table, RMCS matches a set of classifiers to the objects set existing in the context. Then, RMCS looks for the neighbors of the test set of objects using a metric of similarity and selects classifiers that have the maximum number of neighbors found. The selected classifiers are then recommended for classification [10].

4 Discussion

As mentioned previously, methods based on FCA are gathered in two main categories: **exhaustive** methods and **combinatory** ones. Methods of each category vary from each other in several aspects and share some others. Exhaustive

[14] Boosting Formal Concepts
[15] The information quantity contained or supplied by a source of information
[16] Boosting Nominal Concepts
[17] Dagging Nominal Concepts
[18] Fitting Pattern Structures to Knowledge Discovery in Big Data
[19] Recommender based Multiple Classifier System

methods generate only one ordinary classifier for the classification of the objects. Table 1 presents exhaustive methods cited previously. In tables 1 and 2, n denote the number of objects and m the number of attributes. In order to identify characteristics of each method, we chose six criteria that seem to be the most distinctive.

System	GRAND	CIBLE	IPR	CITREC	CLANN	MCSD-FCA-PS
Structure concepts	Complete lattice	Sub-lattice	Coverage	Sub-lattice	Sub-lattice	Sub-lattice
Data	Binary	Numerical	Binary	Binary	Binary	Sequential data
Selection concepts	Coherence maximal	Function selection	Entropy Shannon	Support	Algorithms heuristical	Pattern structures
Combination	No	No	No	No	No	No
Classification	Majority vote	K-PPV	Weighted rules	Vote	Neural network	Projections
Theoretical complexity	$O(2^k \cdot k^4)$ $k = min(m,n)$	$O(\|L\| \cdot m^3)$ $\|L\|$=sublattice	$O(n^2 \cdot m^2 \cdot nm)$	$O(2^m \cdot n)$	$O(2^{min(m,n)})$	$O(\|L\| \cdot \|A\| \cdot m^3 \cdot n)$ m=maximium sequence size A=alphabet of sequence letters

Table 1. Theoretical comparison of exhaustive methods

As shown in Table 1, exhaustive methods have exponential complexity. It is mainly due to a navigation in the totality of the research space.

On the other side, combinatory methods share the classification process in different classifiers a combination method. The problem is thus divided into many sub-problems. Similarly, to Table 1, Table 2 provides a description of combinatory methods. We used the same criteria in two tables for comparative reasons.

Tables 1 and 2 show that GRAND, IPR, CITREC, CLANN, BFC and RMCS handle binary data, BNC and DNC manipulate nominal data while CIBLE handle numerical data. However FPS-FCA and MCSD-FCA-PS differ from the previous methods by their capacity to handle complex data like graphs and sequential data. BNC and DNC use the informational gain to select concepts, while IPR and BFC use Shannon entropy. Concerning CLANN, it utilizes heuristic algorithms for the selection.

In classification process, GRAND, CITREC and DNC use the majority voting. The weighted voting is applied in IPR, BFC and BNC. However, CLANN differ from other methods by the use of neural networks.

The combining technique (cf. section 3.2) contributed strongly in optimizing the complexity. In fact, combinatory methods generate classifiers sequentially and have a polynomial logarithmic complexity. Similarly, methods generating parallel classifiers reach a comparable complexity in the order of $nmlog(n)$ for RMCS method, nm/k for FPS-FCA method and n for DNC.

System	BFC	BNC	DNC	RMCS	FPS-FCA
Concepts structure	Sub-lattice	Sub-lattice	Sub-lattice	Complete lattice	Sub-lattice
Data	Binary	Nominal	Nominal	Binary	Graphs, formulas, strings
Concepts selection	Shannon entropy	Gain informational	Gain informational	Distance euclidean	Relevant patterns
combination	Boosting	Boosting	Dagging	Dagging	Dagging
Classification	Weighted vote	Weighted vote	Majority vote	Maximum number of neighbors	Hypotheses
Theoretical complexity	O($n$log(n)+nm)	O($n$log(n)+nm) m=nominal attribute	O(n') n'=size of stratified samples	O(nmlog(n))	O(nm/k) k=number of processors

Table 2. Theoretical comparison of combinatory methods

5 Conclusion

In this paper, we focused on supervised classification of data based on FCA. We presented firstly exhaustive classification methods which are divided into methods based on complete lattices and methods based on sublattices. Secondly, we described combinatory classification methods which are themselves divided into methods based on sequential training and others based on parallel training.

Our future work will be based on the complexity and move towards combinatory methods that offer reasonable complexity, especially methods that generate parallel classifiers.

References

1. Buzmakov, A., Egho, E., Jay, N., Kuznetsov, S.O., Napoli, A., Raïssi, C.: On mining complex sequential data by means of fca and pattern structures. International Journal of General Systems 45(2), 135–159 (2016)
2. Carpineto, C., Romano, G.: Concept data analysis: Theory and applications. John Wiley & Sons (2004)
3. Douar, B., Latiri, C.C., Slimani, Y.: Approche hybride de classification supervisée à base de treillis de galois: application à la reconnaissance de visages. In: Extraction et Gestion des Connaissances. pp. 309–320 (2008)
4. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R.: Advances in knowledge discovery and data mining (1996)
5. Ferrandin, M., Nievola, J.C., Enembreck, F., Scalabrin, E.E., Kredens, K.V., Ávila, B.C.: Hierarchical classification using fca and the cosine similarity function. In: Proceedings on the International Conference on Artificial Intelligence. p. 1. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp) (2013)
6. Ganter, B., Grigoriev, P.A., Kuznetsov, S.O., Samokhin, M.V.: Concept-based data mining with scaled labeled graphs. In: International Conference on Conceptual Structures. pp. 94–108. Springer (2004)

7. Ganter, B., Stumme, G., Wille, R.: Formal concept analysis: foundations and applications. springer (2005)
8. Gupta, A., Kumar, N., Bhatnagar, V.: Incremental classification rules based on association rules using formal concept analysis. In: Machine Learning and Data Mining in Pattern Recognition, pp. 11–20. Springer (2005)
9. Ikeda, M., Yamamoto, A.: Classification by selecting plausible formal concepts in a concept lattice. In: Workshop on Formal Concept Analysis meets Information Retrieval. pp. 22–35 (2013)
10. Kashnitsky, Y., Ignatov, D.I.: Can fca-based recommender system suggest a proper classifier? What can FCA do for Artificial Intelligence (2015)
11. Kotsianti, S., Kanellopoulos, D.: Combining bagging, boosting and dagging for classification problems. In: Knowledge-Based Intelligent Information and Engineering Systems. pp. 493–500. Springer (2007)
12. Kuznetsov, S.O.: Learning of simple conceptual graphs from positive and negative examples. In: European Conference on Principles of Data Mining and Knowledge Discovery. pp. 384–391. Springer (1999)
13. Kuznetsov, S.O.: Machine learning and formal concept analysis. In: International Conference on Formal Concept Analysis. pp. 287–312. Springer (2004)
14. Kuznetsov, S.O.: Fitting pattern structures to knowledge discovery in big data. In: International Conference on Formal Concept Analysis. pp. 254–266. Springer (2013)
15. Liquiere, M., Mephu Nguifo, E.: Legal: Learning with galois lattice. Journées Françaises sur l'Apprentissage, Lannion pp. 93–113 (1990)
16. Maddouri, M.: Towards a machine learning approach based on incremental concept formation. Intelligent Data Analysis 8(3), 267–280 (2004)
17. Meddouri, N., Khoufi, H., Maddouri, M.: Parallel learning and classification for rules based on formal concepts. Procedia Computer Science 35, 358–367 (2014)
18. Meddouri, N., Maddouri, M.: Boosting formal concepts to discover classification rules. In: Next-Generation Applied Intelligence, pp. 501–510. Springer (2009)
19. Njiwoua, P., Mephu Nguifo, E.: Améliorer l'apprentissage à partir d'instances grâce à l'induction de concepts: le système cible. Revue d'intelligence artificielle 13(2), 413–440 (1999)
20. Oosthuizen, G.D.: The use of a lattice in knowledge processing (1980)
21. Prokasheva, O., Onishchenko, A., Gurov, S.: Classification methods based on formal concept analysis. FCAIR 2012–Formal Concept Analysis Meets Information Retrieval p. 95 (2013)
22. Sahami, M.: Learning classification rules using lattices. In: Machine Learning: ECML-95, pp. 343–346. Springer (1995)
23. Tsopzé, N., Nguifo, E.M., Tindo, G.: Clann: Concept lattice-based artificial neural network for supervised classification. In: Concept Lattice and their applications. vol. 331. Citeseer (2007)
24. Visani, M., Bertet, K., Ogier, J.M.: Navigala: An original symbol classifier based on navigation through a galois lattice. International Journal of Pattern Recognition and Artificial Intelligence 25(04), 449–473 (2011)
25. Xie, Z., Hsu, W., Liu, Z., Lee, M.L.: Concept lattice based composite classifiers for high predictability. Journal of Experimental & Theoretical Artificial Intelligence 14(2-3), 143–156 (2002)

Contributions to the Formalization of Order-like Dependencies using FCA*

Victor Codocedo[1], Jaume Baixeries[2], Mehdi Kaytoue[1], and Amedeo Napoli[3]

[1] Université de Lyon. CNRS, INSA-Lyon, LIRIS. UMR5205, F-69621, France.
[2] Universitat Politècnica de Catalunya. 08032, Barcelona. Catalonia.
[3] LORIA (CNRS - Inria Nancy Grand Est - Université de Lorraine), B.P. 239, F-54506, Vandœuvre-lès-Nancy.

Abstract. Functional Dependencies (FDs) play a key role in many fields of the relational database model, one of the most widely used database systems. FDs have also been applied in data analysis, data quality, knowledge discovery and the like, but in a very limited scope, because of their fixed semantics. To overcome this limitation, many generalizations have been defined to relax the crisp definition of FDs. FDs and a few of their generalizations have been characterized with Formal Concept Analysis which reveals itself to be an interesting unified framework for characterizing dependencies, that is, understanding and computing them in a formal way. In this paper, we extend this work by taking into account order-like dependencies. Such dependencies, well defined in the database field, consider an ordering on the domain of each attribute, and not simply an equality relation as with standard FDs.

1 Introduction

Functional dependencies (FDs) are well-known constraints in the relational model used to show a functional relation between sets of attributes [10], i.e. when the values of a set of attributes are determined by the values of another set of attributes. They are also used in different tasks within the relational data model, as for instance, to check the consistency of a database, or to guide the design of a data model [9].

Different generalizations of FDs have been defined in order to deal with imprecision, errors and uncertainty in real-world data, or simply, to mine and discover more complex patterns and constraints within data when the semantics of FDs have shown to be too restrictive for modeling certain attribute domains. For example, consider the database in the table above as

id	Month	Year	Av. Temp.	City
t_1	1	1995	36.4	Milan
t_2	1	1996	33.8	Milan
t_3	5	1996	63.1	Rome
t_4	5	1997	59.6	Rome
t_5	1	1998	41.4	Dallas
t_6	1	1999	46.8	Dallas
t_7	5	1996	84.5	Houston
t_8	5	1998	80.2	Houston

* Note: A longer version of this paper has been accepted at the conference *Concept Lattices and their Applications*, Moscow, Russia, July 2016.

an example[4]. Attributes of these 8 tuples are city names, month identifiers, years and average temperatures. From this table, we could expect that the value for average temperature is determined by a city name and a month of the year (e.g. the month of May in Houston is hot, whereas the month of January in Dallas is cold). Therefore, we would expect that this relationship should be somehow expressed as a (functional) dependency in the form *city name, month* $\rightarrow$ *average temperature*. However, while the average temperature is truly determined by a city and a time of the year, it is very hard that it will be exactly the same from one year to another. Instead, we can expect that the value will be *similar*, or *close* throughout different years, but rarely the same. Unfortunately, semantics of FDs is based on an equivalence relation and fail to grasp the dependencies among these attributes.

To overcome the limitations of FDs while keeping the idea that some attributes are functionally determined by other attributes, different generalizations of functional dependencies have been defined, as recently deeply reviewed in a comprehensive survey [3]. Actually, the example presented in the last paragraph is a so-called *similarity dependency* [1,3].

In this paper we present an FCA-based characterization of order-like dependencies, a generalization of functional dependencies in which the equality of values is replaced by the notion of order. Firstly, we show that the characterization of *order dependencies* in their general definition [7] can be achieved through a particular use of general ordinal scaling [6]. Secondly, we extend our characterization in order to support *restricted order dependencies* through which other FDs generalizations can be modeled, namely sequential dependencies and trend dependencies [3].

The rest of this paper is organized as follows. In Section 2 we formally introduce the definition of functional dependencies, formal concept analysis and the principle of the characterization of FDs with FCA. In Section 3, we characterize *order dependencies* in their general definition. We show that our formalization can be adapted to *restricted ordered dependencies* in Section 4 before to conclude.

2 Preliminaries

2.1 Functional dependencies

We deal with datasets which are sets of tuples. Let $\mathcal{U}$ be a set of attributes and Dom be a set of values (a domain). For the sake of simplicity, we assume that Dom is a numerical set. A tuple t is a function $t : \mathcal{U} \mapsto Dom$ and then a table T is a set of tuples. We define the functional notation of a tuple for a set of attributes $X \subseteq \mathcal{U}$ as follows, assuming that there exists a total ordering on $\mathcal{U}$. Given a tuple $t \in T$ and $X = \{x_1, x_2, \ldots, x_n\}$, we have: $t(X) = \langle t(x_1), t(x_2), \ldots, t(x_n)\rangle$.

[4] Example from The University of Dayton, that shows the month average temperatures for different cities: `http://academic.udayton.edu/kissock/http/Weather/`

Definition 1 (Functional dependency [10]). *Let T be a set of tuples (data table), and $X, Y \subseteq \mathcal{U}$. A* functional dependency (FD) $X \to Y$ *holds in T if:*

$$\forall t, t' \in T : t(X) = t'(X) \to t(Y) = t'(Y)$$

id	a	b	c	d
t_1	1	3	4	1
t_2	4	3	4	3
t_3	1	8	4	1
t_4	4	3	7	8

Table 1

Example. The table on the right presents 4 tuples $T = \{t_1, t_2, t_3, t_4\}$ over attributes $\mathcal{U} = \{a, b, c, d\}$. We have that $t_2(\{a, c\}) = \langle t_2(a), t_2(c)\rangle = \langle 4, 4\rangle$. Note that the set notation is usually omitted and we write ab instead of $\{a, b\}$. In this example, the functional dependency $d \to c$ holds and $a \to c$ does not hold.

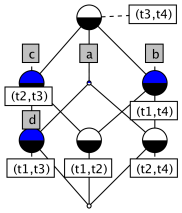
2.2 Characterization of Functional Dependencies with FCA

It has been shown in previous work that functional dependencies can be characterized with FCA. For example, Ganter & Wille [6] presented a data transformation of the initial set of tuples into a formal context. In this context, implications are in 1-to-1 correspondence with the functional dependencies of the initial dataset.

On the bottom-right figure, we illustrate this characterization with the set of tuples from Table 1 where each possible pair of tuples is an object in the formal context. Attributes remain the same. Object (t_i, t_j) has attribute m iff $t_i(m) = t_j(m)$.

The concept lattice in the bottom right: there are two implications, namely $d \to c$ and $d \to a$, which are also the functional dependencies in the original set of tuples. However, this approach implies that a formal context much larger than the original dataset must be processed. It was then shown that this formal context can actually be encoded with a pattern structure [5]: each attribute of the original dataset becomes an object of the pattern structure and is described by a partition on the tuple set. Actually, each block of the partition is composed of tuples taking the same value for the given attribute [8]. For example, in Table 1, the partition describing a is $\{\{t_1, t_3\}, \{t_2, t_4\}\}$. Then, the implications in the pattern concept lattice are here again in 1-to-1 correspondence with the functional dependencies of the initial dataset [2]. What is important to notice is that this formalization is possible as a partition is an *equivalence relation*: a symmetric, reflexive and transitive binary relation. In [1], another kind of dependencies was formalized in a similar way, i.e. similarity dependencies, where the equality relation is relaxed to a similarity relation when comparing two tuples. An attribute is not anymore described by a partition, but by a tolerance relation, i.e. a symmetric, reflexive, but not necessarily transitive binary relation. Each original attribute is then described by a set of tolerance blocks, each being a maximal set of tuples that have pairwise similar values (instead of equal values for classical dependencies).

$\mathbb{K}$	a	b	c	d
(t_1, t_2)		×	×	
(t_1, t_3)	×		×	×
(t_1, t_4)		×		
(t_2, t_3)			×	
(t_2, t_4)	×	×		
(t_3, t_4)				



As we will show next, this way of characterizing FDs and similarity dependencies actually fails for order dependencies, as the relation in this case is not

symmetric: it is neither an equality nor a similarity but a partial order in the general case.

3 Characterization of Order Dependencies with FCA

Although functional dependencies are used in several domains, they cannot be used to express some relationships that exist in data. Many generalizations have been proposed and we focus in this article on order dependencies [7,3]. Such dependencies are based on the *attribute-wise* order on tuples. This order assumes that each attribute follows a partial order associated to the values of its domain. For the sake of generality, we represent this order with the symbol $\sqsubseteq_x$ for all $x \in \mathcal{U}$. In practice, this symbol will be instantiated by intersections of any partial order on the domain of this attribute, as, for instance, $<, \leq, >, \geq$, etc. We remark that this order on the set of values of a *single* attribute does not need to be a total order, although in many different instances, like numeric or character strings domains, this will be the case. Now we formalize operator $\sqsubseteq_x$ (Definition 2) and define accordingly order dependencies (Definition 3).

Definition 2 (Attribute-wise ordering). *Given two tuples $t_i, t_j \in T$ and a set of attributes $X \subseteq \mathcal{U}$, the attribute-wise order of these two tuples on X is:* $t_i \sqsubseteq_X t_j \Leftrightarrow \forall x \in X : t_i[x] \sqsubseteq_x t_j[x]$

This definition states that one tuple is *greater* –in a sense involving the order of all attributes– than another tuple if their attribute-wise values meet this order. This operator induces a partial order $\Pi_X = (T, \prec_X)$ on the set T of tuples.

Definition 3 (Order dependency). *Let $X, Y \subseteq \mathcal{U}$ be two subsets of attributes in a dataset T. An order dependency $X \rightarrow Y$ holds in T if and only if:* $\forall t_i, t_j \in T : t_i \sqsubseteq_X t_j \rightarrow t_i \sqsubseteq_Y t_j$

Example. Consider the table on the right with six tuples and three attributes. Taking $\sqsubseteq_a$, $\sqsubseteq_b$ and $\sqsubseteq_c$ defined as the ordering $\leq$. The orders induced by the sets of attributes $\{a\}$,$\{b\}$,$\{c\}$ and $\{a, b\}$ are:

$\Pi_a = (T, \prec_a) = \{\{t_1\} \prec \{t_2\} \prec \{t_3, t_6\} \prec \{t_5\} \prec \{t_4\}\}$
$\Pi_b = (T, \prec_b) = \{\{t_5\} \prec \{t_1, t_4\} \prec \{t_3\} \prec \{t_2\} \prec \{t_6\}\}$
$\Pi_c = (T, \prec_c) = \{\{t_1\} \prec \{t_2\} \prec \{t_3, t_6\} \prec \{t_5\} \prec \{t_4\}\}$
$\Pi_{ab} = (T, \prec_{ab}) = \{\{t_1\} \prec \{t_2\} \prec \{t_6\}; \{t_1\} \prec \{t_3\}; \{t_1\} \prec \{t_4\}; \{t_5\} \prec \{t_4\}\}$

id	a	b	c
t_1	1	3	1
t_2	2	7	2
t_3	3	4	4
t_4	5	3	9
t_5	4	2	5
t_6	3	8	4

Table 2

These orders are such that the order dependency $\{a, b\} \rightarrow \{c\}$ holds. Remark that Definition 3 is generic since the orders that are assumed for each attribute need to be instantiated: we chose $\leq$ in this example for all attributes, while taking the equality would produce standard *functional dependencies*.

To achieve the characterization of order dependencies with FCA, we propose to represent the partial order $\Pi_X = (T, \prec_X)$ associated to each subset of attribute $X \subseteq \mathcal{U}$ as a formal context $\mathbb{K}_X$ (a binary relation on $T \times T$ thanks to a general ordinal scaling [6]). Then, we show that an order dependency $X \rightarrow Y$ holds iff $\mathbb{K}_X = \mathbb{K}_{XY}$.

$\sqsubset_c$	t_1	t_2	t_3	t_4	t_5	t_6
t_1		×	×	×	×	×
t_2			×	×	×	×
t_3				×	×	×
t_4						
t_5				×		
t_6				×	×	

Table 3: $(T, T, \sqsubset_c)$

$\sqsubset_{ab}$	t_1	t_2	t_3	t_4	t_5	t_6
t_1		×	×			×
t_2						×
t_3						
t_4						
t_5				×		
t_6						

Table 4: $(T, T, \sqsubset_{ab})$

$\sqsubset_{abc}$	t_1	t_2	t_3	t_4	t_5	t_6
t_1		×	×			×
t_2						×
t_3						
t_4						
t_5				×		
t_6						

Table 5: $(T, T, \sqsubset_{abc})$

Definition 4 (General ordinal scaling of the tuple set). *Given a subset of attributes $X \subseteq \mathcal{U}$ and a table dataset T, we define a formal context for $\Pi_X = (T, \prec_X)$ (the partial order it induces) as follows: $\mathbb{K}_X = (T, T, \sqsubset_X)$ where $\sqsubset_X = \{(t_i, t_j) \mid t_i, t_j \in T, t_i \sqsubseteq_X t_j\}$. This formal context is the **general ordinal scale** of Π_X [6]. All formal concepts $(A, B) \in \mathbb{K}_X$ are such that A is the set of lower bounds of B and B is the set of upper bounds of A. Its concept lattice is the smallest complete lattice in which the order Π_X can be order embedded.*

This way to characterize a partial order is only one among several possibilities. However, the choice of formal contexts is due to their versatility, since they can characterize binary relations, hierarchies, dependencies, different orders [6] and graphs [4]. In the next section we will see how this versatility allows us to generalize similarity dependencies. Given the set of attributes $X \subseteq \mathcal{U}$, an associated partial order $\Pi_X = (T, \prec_X)$ and the formal context $(T, T, \sqsubset_X)$, it is easy to show that the later is a composition of contexts defined as: $(T, T, \sqsubset_X) = (T, T, \bigcap_{x \in X} \sqsubset_x)$.

We can now propose a characterization of order dependencies with FCA.

Proposition 1. *An order dependency $X \to Y$ holds in T iff $\mathbb{K}_X = \mathbb{K}_{XY}$.*

Proof. Recall that $\mathbb{K}_{XY} = (T, T, \sqsubset_{XY}) = (T, T, \sqsubset_X \cap \sqsubset_Y)$. We have that

$$X \to Y \iff \sqsubset_X = \sqsubset_X \cap \sqsubset_Y \iff \sqsubset_X \subseteq \sqsubset_Y$$
$$\iff \forall t_i, t_j \in T,\ t_i \sqsubseteq_X t_j \to t_i \sqsubseteq_Y t_j$$

The fact that the order dependency $\{a, b\} \to \{c\}$ holds can be illustrated with the formal contexts in Tables 3,4 and 5. We have indeed that $\mathbb{K}_{ab} = \mathbb{K}_{abc}$.

Order dependencies and other FDs generalizations. We have seen that the definition of order dependencies replaces the equality condition present in FDs or other similarity measures present in other dependencies, by an order relation. This may suggest that order dependencies and other kinds of FDs generalizations are structurally very similar, whereas this is not the case. Functional dependencies generate a reflexive, symmetric and transitive relation in the set of tuples, i.e. an equivalence relation. Then the set of tuples can be partitioned into *equivalence classes* that are used to characterize and compute the set of FDs holding in a dataset, as presented in a previous work [2].

In the generalization of functional dependencies that replaces the equality condition by a *similarity* measure or a distance function, this measure generates a symmetric relation in the set of tuples, but not necessarily a transitive relation. In turn, this implies that the set of tuples can be partitioned into *blocks of tolerance* instead of equivalence classes, as shown in [1].

In this article, the novelty is that we are dealing with a transitive relation, but not necessarily a symmetric relation. That means that we are not dealing with equivalence classes nor blocks of tolerance any longer, but, precisely, with orders. Since the characterization of these dependencies cannot be performed in terms of equivalence classes nor blocks of tolerance, it requires for a more general approach: *general ordinal scaling*.

4 Characterization of Restricted Order Dependencies

Order dependencies allow taking into account the ordering of the values of each attribute when looking for dependencies in data. However, violations of the ordering due to value variations should sometimes not be considered in many real world scenarios. Consider the example given in Table 6: it gives variations on the number of people waiting at a bus station over time.

	$Time$	$People\_waiting$
t_1	10:00	101
t_2	10:20	103
t_3	10:40	105
t_4	11:00	77
t_5	11:20	80
t_6	11:40	85

Table 6

In such a scenario we can expect that more people will be waiting in the station as time moves on ($People\_waiting \rightarrow Time$). However, at some point, a bus arrives and the number of people waiting decreases and starts increasing again. It is easy to observe that the order dependency $People\_waiting \rightarrow Time$ does not hold as we have the counter-example: $t_4 \sqsubseteq_{People\_waiting} t_3$ and $t_3 \sqsubseteq_{Time} t_4$.

However, the gap between the values 77 and 105 is significant enough to be considered as a different instance of the ordering. We can formalize this idea by introducing a *similarity threshold* $\theta = 10$ for the attribute $People\_waiting$ such that the ordering between values is checked iff the difference is smaller than θ. In this way, the previous counter-example is avoided (*restricting* the binary relation) along with any other counter-example and we have that the restricted order dependency $People\_waiting \rightarrow Time$ holds.

We now formalize the tuple ordering relation, and consequently the notion of restricted order dependencies.

Definition 5. *Given two tuples $t_i, t_j \in T$ and a set of attributes $X \subseteq \mathcal{U}$, the attribute-wise order on X is: $t_i \sqsubseteq_x^* t_j \Leftrightarrow \forall x \in X : 0 \leq t_j[x] - t_i[x] \leq \theta_x$.*

Definition 6. *Let $X, Y \subseteq \mathcal{U}$ two sets of attributes in a table T such that $|T| = n$, and let θ_X, θ_Y be thresholds values of tuples in X and Y respectively. A restricted order dependency $X \rightarrow Y$ holds in T iff: $t[X] \sqsubseteq_X^* t'[X] \rightarrow t[Y] \sqsubseteq_Y^* t'[Y]$*

Using these definitions we can encode the tuple ordering relations as formal contexts for any subset of attributes $X \subseteq \mathcal{U}$. Indeed, the binary relations between tuples by operator $\sqsubseteq_X^*$ can be encoded in a formal context $\mathbb{K}_X^* = (T, T, \sqsubseteq_X^*)$

$\sqsubseteq^*_{Tm}$	t_1	t_2	t_3	t_4	t_5	t_6
t_1	×	×	×	×	×	×
t_2		×	×	×	×	×
t_3			×	×	×	×
t_4				×	×	×
t_5					×	×
t_6						×

Table 7: $(T, T, \sqsubseteq^*_{Tm})$

$\sqsubseteq^*_{Pp}$	t_1	t_2	t_3	t_4	t_5	t_6
t_1	×	×	×			
t_2		×	×			
t_3			×			
t_4				×	×	×
t_5					×	×
t_6						×

Table 8: $(T, T, \sqsubseteq^*_{Pp})$

$\sqsubseteq^*_{Tm,Pp}$	t_1	t_2	t_3	t_4	t_5	t_6
t_1	×	×	×			
t_2		×	×			
t_3			×			
t_4				×	×	×
t_5					×	×
t_6						×

Table 9: $(T, T, \sqsubseteq^*_{Tm,Pp})$

which in turn, can be composed from single attributes $x \in \mathcal{U}$: $\sqsubseteq^*_X = \bigcap_{x \in X} \sqsubseteq^*_x$. Moreover, we can use the same rationale we used to mine order dependencies to find restricted order dependencies.

Proposition 2. *A restricted order dependency $X \to Y$ holds in T iff*

$$X \to Y \iff \mathbb{K}^*_X = \mathbb{K}^*_{XY}$$

Proof. This proposition can be proved similarly to Proposition 1.

Example. For the previous example, we calculate the corresponding formal contexts shown in Tables 7 and 8 ($\sqsubseteq^*_{Tm}$ for *Time*, and $\sqsubseteq^*_{Pp}$ for *People_waiting*). It is easy to observe that the restricted order dependency $People\_waiting \to Time$ holds as we have that $\mathbb{K}^*_{Pp} = \mathbb{K}^*_{Pp,Tm}$.

Restricted order dependencies and other FDs generalizations. Similarity dependencies (SDs) generalize functional dependencies through the use of a tolerance relation instead of an equivalence relation between values of tuples for a given set of attributes. A tolerance relation is a reflexive, symmetric and non-transitive binary association between two tuples given a threshold θ. In a nutshell, a SD is established between two tuples if their values are within a given *distance* controlled by the threshold. Such dependencies were studied in a previous work [1]. However, from the perspective of order dependencies, we can request that such distance has a certain polarity. As we have previously discussed, order dependencies arise from anti-symmetric, not necessarily reflexive, and transitive binary relations $(<, \leq)$. Then, it can be expected that using a *threshold of distance* θ between tuple values for a given set of attributes requires an antisymmetric, non-transitive relation between the values of tuples w.r.t. a set of attributes X, that we have defined as $\sqsubseteq^*_X$.

The current approach has the potential to implement some other FD generalizations of such as sequential dependencies and trend dependencies [3]. The latter is actually a particular case of restricted order dependencies where the threshold is applied to an attribute not contained in the attributes of the dependency. Instead, it is applied to a *time* attribute that allows defining *snapshots* of a database. In sequential dependencies, the antecedent is a mapping of a set of attributes with a *complete unrestricted* order (without a threshold). Details on both these dependencies have been left out from this paper for space reasons.

5 Conclusion

We have presented a characterization of order dependencies with FCA, which can be potentially extended to other types of order-like dependencies, used in different fields of database theory, knowledge discovery and data quality. These dependencies are part of a set of functional dependencies generalizations where equality condition is replaced with a more general relation. In some cases, the equality is replaced by an approximate measure, in other cases, like in order dependencies, by an order relation.

We have seen that order dependencies are based on a transitive, but not necessarily symmetric relation, contrasting similarity dependencies, which are based on a symmetric, but not necessarily transitive relation. It is precisely this formalization in terms of FCA that allows us to find these structural differences between these types of dependencies.

Nevertheless, the present work needs to be extended to other kinds of order-like dependencies while experimentation needs to be performed in order to verify the computational feasibility of this approach.

Acknowledgments. This research work has been supported by the SGR2014-890 (MACDA) project of the Generalitat de Catalunya, the MINECO project APCOM (TIN2014-57226-P) and the French National Project FUI AAP 14 Tracaverre.

References

1. J. Baixeries, M. Kaytoue, and A. Napoli. Computing similarity dependencies with pattern structures. In *CLA 2013*, CEUR Workshop Proceedings, 2013.
2. J. Baixeries, M. Kaytoue, and A. Napoli. Characterizing functional dependencies in formal concept analysis with pattern structures. *Annals of Mathematics and Artificial Intelligence*, 72(1-2):129–149, Oct. 2014.
3. L. Caruccio, V. Deufemia, and G. Polese. Relaxed functional dependencies - A survey of approaches. *IEEE Trans. Knowl. Data Eng.*, 28(1):147–165, 2016.
4. S. Ferré. A Proposal for Extending Formal Concept Analysis to Knowledge Graphs. In *Formal Concept Analysis*, volume LNCS 9113, pages 271–286, June 2015.
5. B. Ganter and S. O. Kuznetsov. Pattern structures and their projections. In *ICCS 2001*, LNCS 2120, pages 129–142. Springer, 2001.
6. B. Ganter and R. Wille. *Formal Concept Analysis*. Springer, Berlin, 1999.
7. S. Ginsburg and R. Hull. Order dependency in the relational model. *Theoretical Computer Science*, 26(1):149 – 195, 1983.
8. Y. Huhtala, J. Kärkkäinen, P. Porkka, and H. Toivonen. Tane: An efficient algorithm for discovering functional and approximate dependencies. *Computer Journal*, 42(2):100–111, 1999.
9. H. Mannila and K.-J. Räihä. *The Design of Relational Databases*. Addison-Wesley, Reading (MA), USA, 1992.
10. J. Ullman. *Principles of Database Systems and Knowledge-Based Systems, volumes 1–2*. Computer Science Press, Rockville (MD), USA, 1989.

A Hybrid Approach for Mining Metabolomic Data

Dhouha Grissa[1,3], Blandine Comte[1], Estelle Pujos-Guillot[2], and Amedeo Napoli[3]

[1] INRA, UMR1019, UNH-MAPPING, F-63000 Clermont-Ferrand, France,
[2] INRA, UMR1019, Plateforme d'Exploration du Métabolisme, F-63000 Clermont-Ferrand, France
[3] LORIA, B.P. 239, F-54506 Vandoeuvre-lès-Nancy, France

Abstract. In this paper, we introduce a hybrid approach for analyzing metabolomic data about the so-called "diabetes of type 2". The identification of biomarkers which are witness of the disease is very important and can be guided by data mining methods. The data to be analyzed are massive and complex and are organized around a small set of individuals and a large set of variables (attributes). In this study, we based our experiments on a combination of efficient numerical supervised methods, namely Support Vector Machines (SVM), Random Forests (RF), and ANOVA, and a symbolic non supervised method, namely Formal Concept Analysis (FCA). The data mining strategy is based on ten specific classification processes which are organized around three main operations, filtering, feature selection, and post-processing. The numerical methods are mainly used in filtering and feature selection while FCA is mainly used for visualization and interpretation purposes. The first results are encouraging and show that the present strategy is well-adapted to the mining of such complex biological data and the identification of potential predictive biomarkers.

Keywords: hybrid knowledge discovery, Random Forest, SVM, ANOVA, Formal Concept Analysis, feature selection, discrimination, predication, visualization

1 Introduction

Metabolomics allows the analysis of a biological system by measuring metabolites, i.e. small molecules, present and accessible in the system. Usually different techniques are necessary for such an analysis. In particular, one challenge of metabolomics is to identify, among thousands of features, predictive biomarkers, i.e. a measurable indicator of some biological status, of a disease development [7]. This can be viewed as a hard data mining task as data generated by metabolomic platforms are massive, complex and noisy. In the present study, we aim at identifying predictive metabolic biomarkers of future T2D –type 2 diabetes– development, a few years before occurrence, in an homogeneous population considered

as healthy at the time of the analysis. The datasets include a limited number of individuals, a large number of variables or attributes, e.g. molecules or fragments of molecules, and one binary target variable, i.e. developing or not the disease a few years after the analysis.

One important problem here is to distinguish between discriminant and predictive features. A feature is said to be "discriminant" if it separates individuals in distinct classes, such as for example healthy vs not healthy. A feature is said to be "predictive" if it enables predicting the evolution of individuals towards the disease a few years later. However, the most discriminant features are not necessarily the best predictive features. Thus, it is essential to compare different feature selection methods and to evaluate their capabilities to select relevant features for further use in prediction.

Accordingly, we propose a knowledge discovery process which is based on a combination of numeric-symbolic techniques for different purposes, such as noise filtration for avoiding overfitting –which occurs when the analysis describes random error or noise instead of the underlying relationships–, feature selection for reducing dimension, and checking the relevance of selected features w.r.t. prediction. FCA [3] is then applied to the resulting reduced dataset for visualization and interpretation purposes. More precisely, this hybrid data mining process combines FCA with several numerical classifiers including Random Forest (RF) [1], Support Vector Machine (SVM) [8], and Analysis of Variance (ANOVA) [2]. RF, SVM and ANOVA are used to discover discriminant biological patterns which are then organized and visualized thanks to FCA.

Actually, the initial problem statement is based on a data table *individuals* $\times$ *features*. The data preparation step involves filtering methods based on the correlation coefficient ("Cor") and mutual information ("MI") to eliminate redundant and dependent features, to reduce the dimensions of the data table and to prepare the application of RF, SVM and ANOVA. The initial data table *individuals* $\times$ *features* is transformed into a binary data table *features* $\times$ *classification-process* in the following way. Ten different classifications processes (CPs hereafter) are defined and applied to the initial data table. Every CP provides a ranking of features. Then, the N best classified features are kept for being processed by FCA. Actually, a feature is selected when it is ranked among the six first features. This leads us to a selection of $N = 48$ features and to a binary data table, 48-*features* $\times$ 10-*CPs*, which is in turn considered as a formal context for the application of FCA and the construction of concept lattices. These $N = 48$ features are shared by all CPs and are interpreted as potential biomarkers of disease development.

Meanwhile, biological experts want to classify the selected features as potential predictive biomarkers, i.e. biomarkers able to predict the disease development a few years before occurrence. Predictive biomarkers can be detected thanks to ROC curves [6]. In the current study, such an analysis produces a short list of the best predictive features which are selected as a core set of biomarkers. Finally, FCA is used again to build the best ranking within this core set of biomarkers and for visualization and interpretation purposes. This is one originality of

this paper to present a combination of numerical data mining methods based on RF and SVM with FCA, which in turn is mainly used for interpretation and visualization.

The paper is organized as follows. Section 2 presents preparation and mining of the data for discovering the potential predictive features. Then Section 3 describes experiments performed on real-world metabolomic data set. A discussion and a conclusion complete the paper.

2 The preparation and the mining of metabolomic data

All experiments in the following were carried out on a Dell machine running Ubuntu GNU/Linux 14.04 LTS, a 3.60 GHZ $\times$ 8 CPU and 16 GB RAM. The data analysis methods are taken from the RStudio software environment (Version 0.98.1103, R 3.1.1). Rstudio is freely available and offers a selection of packages suitable for many different types of data[1].

2.1 The dataset description

The dataset which is analyzed is based on a case-control study from the GAZEL French population-based cohort (20000 subjects). This set includes numeric and symbolic data about 111 male subjects (54-64 years old) free of T2D at baseline. 55 subjects developed T2D at the follow-up belong to class "1" (non healthy or diabetic subjects) while 56 subjects belong to class "-1" (controls or healthy subjects). Three thousand features are generated after carrying out mass spectrometry (MS) analysis. After noise filtration, 1195 features are remaining for describing every subject.

The reference dataset is composed of homogeneous individuals considered healthy at the beginning of the study. The binary variable describing the two target classes, i.e. healthy and not healthy, is based on the health status of the same individuals at another time, actually five years after the initial analysis. Meanwhile, some individuals developed the disease. Thus, discriminant features which enable a good separation between target data classes (healthy vs. not healthy) are not necessarily the best features predicting the disease development five years later.

2.2 Data preprocessing

Only a few features allow a good separation between the target classes. Therefore, it is necessary to reduce data dimension to select a small number of relevant features for further use in prediction. Reducing the dimensionality of the data is a challenging step, requiring a prior filtering of the initial data. Metabolomic data contain highly correlated features, which can have an impact on feature selection and data mining [4]. Thus, two filtering methods are chosen, namely the

[1] https://www.rstudio.com/

coefficient of correlation ("Cor") and mutual information ("MI"). Both filtering methods are used to discard correlated features and dependent features.

Figure 1 describes the global classification workflow. At the beginning, the filtering methods "Cor" and "MI" eliminate highly correlated features. Afterward, two reduced subsets are generated: a first subset contains 963 features (after "Cor" filtering) while the second subset contains 590 features (after "MI" filtering). Both reduced subsets are used as input for the application of RF and SVM classifiers.

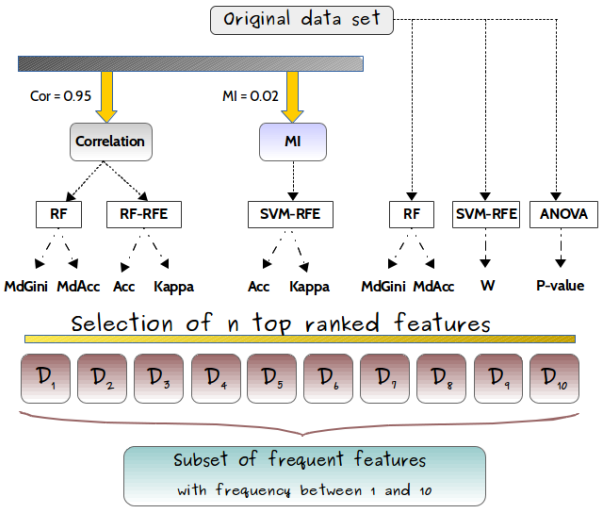


Fig. 1. Feature selection and dimensionality reduction process.

2.3 Feature selection

Two main classification techniques are then applied to the resulting filtered subsets of features, namely RF and SVM. Moreover, for improving the process, RF and SVM are combined with RFE ("Recursive Feature Elimination"), which is a backward elimination method used for feature selection [5]. Finally, three different classification processes are defined: (i) RF applied to data filtered with "Cor", (ii) RF+RFE applied to data filtered with "Cor", (iii) SVM+RFE applied to data filtered with "MI".

In parallel, we apply the the ANOVA method directly on the original dataset as this is a common practice in metabolomics (without any filtering process). This time, SVM+RFE, RF and ANOVA are directly applied to the original dataset.

The measure of the importance of each selected feature in the output of the classification process is the purpose of post-processing. There, several measures of interest (accuracy metrics) enable the ranking of the features, namely MdGini, MdAcc, Accuracy and Kappa. MdGini stands for "Mean decrease in Gini index" is used as an impurity function. MdAcc stands for "Mean decrease in accuracy" and measures the importance/performance of each feature in the classification. Kappa is a statistical measure comparing observed accuracy with expected accuracy. The general idea about the use of these metrics is to measure the decrease in accuracy after permutation of the values of each variable. The scores given by these metrics allow to rank the features (highest discriminative power) within each classification process.

On the same basis, when no filtering is applied, post-processing is based on MdGini and MdAcc for RF, on the weight magnitude of features "W" for SVM+REF, and on the "p-value" for ANOVA. The p-value determines the statistical significance of the results when a hypothesis test is performed.

Based on these different processes, various forms of results, e.g. feature ranking and feature weighting, and as well multiple sets of ranked features are produced. Actually, 10 sets are generated, corresponding to the different CPs and ranking scores. They are denoted by $D_i, i = 1, \ldots, 10$ in Figure 1.

For each classification process, a corresponding name is created which describe the set of operations the process is based on. We have the ten following processes: (1) "Cor-RF-MdAcc", i.e. filtering with "Cor", feature selection with RF and post-processing with MdAcc, (2) "Cor-RF-MdGini", (3) "Cor-RF-RFE-Acc", (4) "Cor-RF-RFE-Kap", (5) "MI-SVM-RFE-Acc", (6) "MI-SVM-RFE-Kap", (7) "RF-MdAcc", (8) "RF-MdGini", (9) "SVM-RFE-W" and finally (10) "ANOVA-pValue".

To select the most important features, we retain the 200 first ranked features, except for "ANOVA-pValue" where we only selected 107 features with a reasonable p-value for filtering purposes (lower than 0.1). Finally, ten reduced sets of ranked features, i.e. $D_i, i = 1, \ldots, 10$, are obtained and should be analyzed for discovering the "best features". Then, the visualization of these "best features" is carried out thanks to FCA.

2.4 Visualization and interpretation with FCA

In this section, we show how to compare the highly ranked features in the reduced subsets $D_i, i = 1, \ldots, 10$. For this purpose, a binary data table *features* $\times$ *CPs* is built (see Table 1), where objects in rows correspond to features and attributes in columns correspond to the 10 classification processes (CPs). The presence of 1 in a cell of the data table means that the feature in the row is ranked for the CP in the column. Every feature has a support, i.e. the number of 1 in the row, which should be at least of 6/ This means that every feature appears among the best ranked features with a frequency between 6 and 10. This leads us to consider $N = 48$ such features. The new binary data table 48-*features* $\times$ 10-*CPs* is presented in Table 1. Starting from the initial data table 111-*individuals* $\times$

1195-*features* we finally get a binary data table 48-*features* $\times$ 10-*CPs*. The "m/z" label of features stands for "mass per charge".

Applying FCA on the 48-*features* $\times$ 10-*CPs* data table considered as a context produces a concept lattice with 272 concepts (Figure 2). This concept lattice illustrates the combination of numerical classification methods with FCA, and allow an interpretation of the relations between features and classification processes, and further on the discriminative and predictive powers of the features. Four features "m/z 383", "m/z 227", "m/z 114" and "m/z 165" have a maximal support of 10 (see the maximum rectangle full of 1 in Table 1). There are strong relationships among the 44 remaining features, especially involving "m/z 284", "m/z 204", "m/z 132", "m/z 187", "m/z 219", "m/z 203", "m/z 109", "m/z 97" and "m/z 145". Moreover, among the 48 features, 39 are significant w.r.t. ANOVA (with a p-value < 0.05).

The concept lattice highlights the potential of the feature selection approach for analyzing metabolomic data. It enables discriminating direct and indirect associations, e.g. highly linked metabolites belonging to the same concept. The links between the concepts in the lattice can be interpreted as interdependency between concept and metabolites.

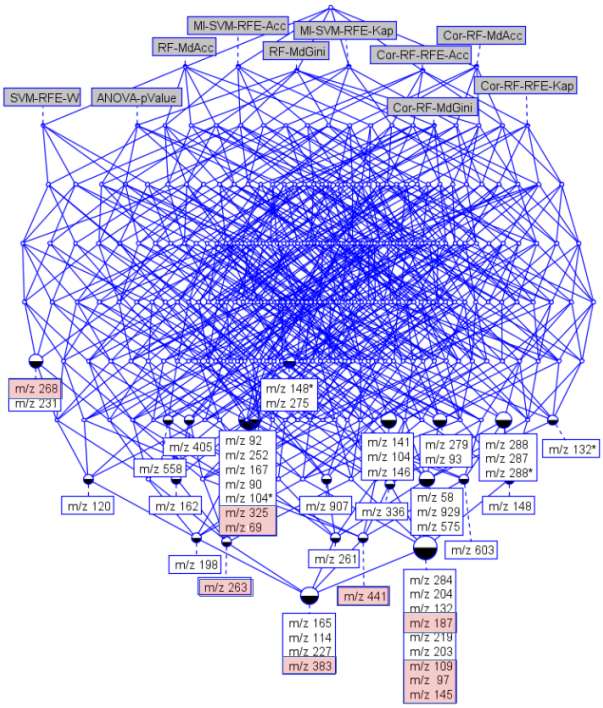


Fig. 2. The concept lattice derived from the 48 $\times$ 10 binary table (Table 1).

Table 1. Input binary table describing the 48 frequent features with the 10 CPs.

Features	Cor-RF-MdGini	Cor-RF-MdAcc	Cor-RF-RFE-Acc	Cor-RF-RFE-Kap	RF-MdGini	RF-MdAcc	MI-SVM-RFE-Acc	MI-SVM-RFE-Kap	SVM-RFE-W	ANOVA-pValue
m/z 383	1	1	1	1	1	1	1	1	1	1
m/z 227	1	1	1	1	1	1	1	1	1	1
m/z 114	1	1	1	1	1	1	1	1	1	1
m/z 165	1	1	1	1	1	1	1	1	1	1
m/z 145	1	1	1	1	1	1	1	1		1
m/z 97	1	1	1	1	1	1	1	1		1
m/z 441	1	1	1	1	1	1	1		1	1
m/z 109	1	1	1	1	1	1	1	1		1
m/z 203	1	1	1	1	1	1	1	1		1
m/z 219	1	1	1	1	1	1	1	1		1
m/z 198	1		1	1	1	1	1	1	1	1
m/z 263	1	1	1		1	1	1	1	1	1
m/z 187	1	1	1	1	1	1	1	1		1
m/z 132	1	1	1	1	1	1	1	1		1
m/z 204	1	1	1	1	1	1	1	1		1
m/z 261	1	1	1	1	1	1	1	1	1	
m/z 162	1	1			1	1	1	1	1	1
m/z 284	1	1	1	1	1	1	1	1		1
m/z 603	1	1	1	1	1	1			1	1
m/z 148	1	1	1	1	1	1		1		1
m/z 575	1	1	1	1	1	1	1	1		
m/z 69	1	1			1	1	1	1		1
m/z 325	1	1			1	1	1	1		1
m/z 405	1	1			1	1	1		1	1
m/z 929	1	1	1	1	1	1	1	1		
m/z 58	1	1	1	1	1	1	1	1		
m/z 336	1	1	1	1	1	1	1			1
m/z 146	1		1	1	1		1	1		1
m/z 104	1		1	1	1		1	1		1
m/z 120			1	1	1	1	1	1	1	1
m/z 558	1	1			1	1	1	1	1	
m/z 231					1	1	1	1	1	1
m/z 132*	1	1	1	1	1		1	1		
m/z 93		1	1	1	1	1	1	1		
m/z 907		1	1	1		1	1	1	1	
m/z 279		1	1	1	1	1	1	1		
m/z 104*	1	1			1	1	1	1		1
m/z 90	1	1			1	1	1	1		1
m/z 268					1	1	1	1	1	1
m/z 288*	1	1	1	1	1	1				1
m/z 287	1	1	1	1	1	1				1
m/z 167	1	1			1	1	1	1		1
m/z 288	1	1	1	1	1	1				1
m/z 252	1	1			1	1	1	1		1
m/z 141	1		1	1	1		1	1		1
m/z 275			1	1	1		1	1		1
m/z 148*			1	1	1		1	1		1
m/z 92	1	1			1	1	1	1		1

3 Evaluation and discussion

Considering the 48 most frequent features previously identified, we evaluate their predictive capabilities using ROC curves (Figure 3). This analysis was carried out

using the ROCCET tool (`http://www.roccet.ca`), with calculation of the area under the curve ("AUC") and confidence intervals (CI), calculation of the true positive rate (TPR), where $TPR = TP/(TP+FN)$, and the false discovery rate (FDR), where $FDR = TN/(TN+FP)$. The p-values of these relevant features are also computed using t-test.

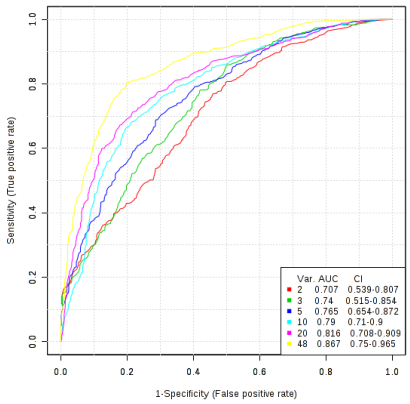


Fig. 3. The ROC curves of at least 2 and at most 48 combined frequent features based on RF model and AUC ranking.

The analysis based on ROC curves is considered as being one of the most objective and statistically valid method for biomarker performance evaluation [6]. ROC curves are commonly used to evaluate the prediction performance of a set of features, or their accuracy to discriminate diseased cases from normal cases.

Since the number of features to propose as predictive biomarkers should be rather small (because of clinical constraints), we rely on the ROC curves of 2, 3, 5, 10, 20 and 48 of features ranked w.r.t. AUC values. The ROC curves enable identifying this best combination of predictive features. Figure 3 shows that the best performance is given to the 48 features all together (with $AUC = 0.867$). But a predictive model with 48 features is not usable for clinical purposes. The set of best features with the smallest p-values and the highest accuracy values is selected and yields a short list of "potential biomarkers". For the ten first features in Table 2, we have $AUC = 0.79$ and $CI = 0.71 - 0.9$. For the four first features, we have $AUC = 0.75$. These high AUC values are witness of a good predictive behavior.

Then we selected as "potential biomarkers" the 10 first features with an AUC greater than 0.74 and significantly small t-test values ($< 10E-5$) (Table 2). We compare this subset with the four most frequent features (features whose frequency is 10 in Table 1) and we find only one feature in common, namely "m/z

383". This confirms that the most frequent features are not the best predictive ones, as biologically suspected, because the metabolomic analysis is performed 5 years before disease occurrence. Moreover, these best "predictive features" or "potential biomarkers" are not belonging to the same concept.

Figure 2 shows that the best predictive biomarkers are lying in different concepts, depicted by red squares in the lattice. For example, the features "m/z 145", "m/z 97", "m/z 109" and "m/z 187" are in the extent of a concept whose intent includes all CPs but "SVM-RFE-W". By contrast, the feature "m/z 268" belongs to another concept whose intent includes 6 CPs, namely "RF-MdGini", "RF-MdAcc", "MI-SVM-RFE-Acc", "MI-SVM-RFE-Kap", "SVM-RFE-W", "ANOVA-pValue". Here again, the direct visualization through the concept lattice shows the position of the predictive features among the discriminant ones and their associations with CPs. This information is very interesting for the domain experts for choosing the best combinations of feature selection methods that can identify the predictive biomarkers.

Name	AUC	T-tests
m/z 145	0.79	1.4483E-6
m/z 383	0.79	5.0394E-7
m/z 97	0.78	1.5972E-6
m/z 325	0.77	2.2332E-5
m/z 69	0.76	1.2361E-5
m/z 268	0.75	4.564E-6
m/z 441	0.75	9.0409E-5
m/z 263	0.75	5.996E-6
m/z 187	0.74	9.0708E-6
m/z 109	0.74	2.6369E-5

Table 2. Table of performance of the best 10 AUC ranked features.

4 Conclusion and future work

In this paper, we presented a hybrid approach for the identification of predictive biomarkers from complex metabolomic dataset. The nature of metabolomic data, i.e. highly correlated and noisy, leads us to build and analyze reduced datasets for identifying important features to be interpreted as potential biomarkers. Moreover, the present hybrid approach is based on an original combination of numerical supervised classification methods (mainly RF, SVM and ANOVA) and a symbolic unsupervised method such as FCA. This study shows the interest of such a combination to reveal hidden information in such high dimensional datasets and how FCA can be used for visualization and interpretation purposes. Based on the resulting lattice, experts in biology will be able to lead a deeper interpretation. Finally, additional experiments on different metabolomic datasets are required to confirm the success of this hybrid approach.

References

1. L. Breiman. Random forests. In *Machine Learning*, pages 5–32, 2001.
2. H. W. Cho, S. B. Kim, and M. K. Jeong et al. Discovery of metabolite features for the modelling and analysis of high-resolution nmr spectra. *International Journal of Data Mining and Bioinformatics*, 2(2):176–192, 2008.
3. B. Ganter and R. Wille. *Formal concept analysis - mathematical foundations.* Springer, 1999.
4. PS. Gromski, H. Muhamadali, DI. Ellis, Y. Xu, E. Correa, ML. Turner, and R. Goodacre. A tutorial review: Metabolomics and partial least squares-discriminant analysis–a marriage of convenience or a shotgun wedding. *Anal Chim Acta.*, 879:10–23, 2015.
5. I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Mach. Learn.*, 46:389–422, 2002.
6. Xia J, Broadhurst DI, Wilson M, and Wishart DS. Translational biomarker discovery in clinical metabolomics: an introductory tutorial. *Metabolomics*, 9(2):280–99, 2013.
7. M. Mamas, WB. Dunn, L. Neyses, and R. Goodacre. The role of metabolites and metabolomics in clinically applicable biomarkers of disease. *Arch Toxicol.*, 85(1):5–17, 2011.
8. V.N. Vapnik. *Statistical Learning Theory.* Wiley-Interscience, John Willey & Sons, 1998.