CLA 2012

Proceedings of the Ninth International Conference on Concept Lattices and Their Applications

CLA Conference Series

http://cla.inf.upol.cz

Universidad de Málaga (Dept. Matemática Aplicada), Spain

The Ninth International Conference on Concept Lattices and Their Applications

CLA 2012

Fuengirola (Málaga), Spain
October 11–14, 2012

Edited by

Laszlo Szathmary
Uta Priss

Technical Editor:
Laszlo Szathmary, `szathmary.laszlo@inf.unideb.hu`

Page count:	xii+357
Impression:	75
Edition:	1$^{\text{st}}$
First published:	2012

Printed version published by:
Universidad de Málaga (Dept. Matemática Aplicada), Spain
ISBN 978–84–695–5252–0

Organization

CLA 2012 was organized by the Universidad de Málaga (Dept. Matemática Aplicada)

Steering Committee

Radim Belohlavek	Palacky University, Olomouc, Czech Republic
Sadok Ben Yahia	Faculté des Sciences de Tunis, Tunisia
Jean Diatta	Université de la Réunion, France
Peter Eklund	University of Wollongong, Australia
Sergei O. Kuznetsov	State University HSE, Moscow, Russia
Michel Liquière	LIRMM, Montpellier, France
Engelbert Mephu Nguifo	LIMOS, Clermont-Ferrand, France

Program Chairs

Laszlo Szathmary	University of Debrecen, Hungary
Uta Priss	Ostfalia University, Wolfenbüttel, Germany

Program Committee

Jaume Baixeries	Polytechnical University of Catalonia
Radim Belohlavek	Palacky University, Olomouc, Czech Republic
Sadok Ben Yahia	Faculty of Sciences, Tunis, Tunisia
Karell Bertet	University of La Rochelle, France
François Brucker	University of Marseille, France
Claudio Carpineto	Fondazione Ugo Bordoni, Roma, Italy
Pablo Cordero	Universidad de Málaga, Spain
Felix Distel	TU Dresden, Germany
Florent Domenach	University of Nicosia, Cyprus
Mireille Ducassé	IRISA/INSA Rennes, France
Ramon Fuentes-Gonzalez	Universidad Publica de Navarra, Spain
Cynthia Vera Glodeanu	TU Dresden, Germany
Marianne Huchard	LIRMM, University of Montpellier 2, France
Vassilis G. Kaburlasos	TEI of Kavala, Greece
Mehdi Kaytoue	LIRIS/INSA Lyon, France
Stanislav Krajci	University of P.J. Safarik, Kosice, Slovakia
Marzena Kryszkiewicz	Warsaw University of Technology, Poland
Sergei O. Kuznetsov	State University HSE, Moscow, Russia
Jesús Medina-Moreno	University of Cadiz, Spain
Rokia Missaoui	UQO, Gatineau, Canada

Amedeo Napoli	INRIA NGE/LORIA, Nancy, France
Lhouari Nourine	LIMOS, University of Clermont Ferrand, France
Sergei Obiedkov	State University HSE, Moscow, Russia
Jan Outrata	Palacky University, Olomouc, Czech Republic
Pascal Poncelet	LIRMM, Montpellier, France
Camille Roth	EHESS, Paris, France
Baris Sertkaya	SAP Research Center, Dresden, Germany
Henry Soldano	Université of Paris 13, France
Gerd Stumme	University of Kassel, Germany
Petko Valtchev	Université du Québec à Montréal, Canada

Additional Reviewers

Zeina Azmeh	LIRMM, University of Montpellier 2, France
Daniel Borchmann	TU Dresden, Germany
Stephan Doerfel	University of Kassel, Germany
Xavier Dolques	INRIA Rennes, France
Robert Jäschke	L3S Research Center, Hannover, Germany

Local Organizing Committee

Manuel Ojeda-Aciego (chair)	Universidad de Málaga, Spain
Pablo Cordero	Universidad de Málaga, Spain
Inma P. Cabrera	Universidad de Málaga, Spain

Table of Contents

Preface

The Ninth International Conference "Concept Lattices and Applications (CLA 2012)" is held in Fuengirola (Málaga), Spain from October 11th until October 14th 2012. CLA 2012 is aimed at providing to everyone interested in Formal Concept Analysis and more generally in Concept Lattices or Galois Lattices, students, professors, researchers and engineers, a global and an advanced view of some of the latest research trends and applications in this field. As the diversity of the selected papers shows, there is a wide range of theoretical and practical research directions, in the field of data and knowledge processing, such as data mining, knowledge discovery, knowledge representation, reasoning, pattern recognition, together with logic, algebra and lattice theory.

This volume includes the selected papers and the abstracts of the 4 invited talks. This year there were initially 44 submissions from which 28 papers were accepted as full papers and 3 papers as short papers. We would like to thank the authors for their work, often of very good quality, the members of the program committee and the external reviewers without whose tremendous efforts this conference would not have been possible. This is evidence of the growing quality and importance of CLA, highlighting its leading position in the field.

We would like to thank the financial support that this conference has received from the Research Promotion Programme of the Universidad de Málaga. We would also like to thank the steering committee of CLA for giving us the opportunity of leading this edition of CLA, the conference participants for their participation and support, and people in charge of the organization.

Finally, we also should not forget that the conference was managed (quite easily) with the Easychair system, paper submission, selection, and reviewing, and that Jan Outrata has offered his files for preparing the proceedings.

October 2012

Laszlo Szathmary
Uta Priss
Program Chairs of CLA 2012

Some results on the L-Fuzzy Concept Analysis

Ana Burusco

Dpt. Automática y Computación, Univ. Pública de Navarra
Campus de Arrosadía, 31006 - Pamplona (Spain)
burusco@unavarra.es

Abstract. The main goal of this talk is the study of some extensions of the Formal Concept Analysis to the L-fuzzy case as the interval-valued L-fuzzy contexts or the K-labeled L-fuzzy contexts. These results are applied to the extraction of information when we do not have all the necessary elements replacing the absent values by means of implications between attributes associated with labels. I will show an application to the diagnosis of the short-circuits produced in an electrical network. Moreover, using a fuzzy tolerance relation R, certain fuzzy relations are characterized as solutions of $X \lhd R = X$, proving that they can be determined by means of the L-fuzzy concepts associated with the K-labeled L-fuzzy contexts. In the last part of the talk the L-fuzzy context sequences are studied using OWA operators. A particular case of this situation appears when we want to study the evolution of an L-fuzzy context in time.

Multi-adjoint concept lattices

Jesús Medina

Dept. Mathematics, University of Cádiz, Spain*
jesus.medina@uca.es

Abstract. Multi-adjoint concept lattices were introduced [2,3] as a new general approach to formal concept analysis, in which the philosophy of the multi-adjoint paradigm [1,4] to formal concept analysis is applied. With the idea of providing a general framework in which different approaches could be conveniently accommodated, the authors worked in a general non-commutative environment; and this naturally lead to the consideration of adjoint triples, also called implication triples or bi-residuated structure as the main building blocks of a multi-adjoint concept lattice.
In recent years there has been an increased interest in studying formal concept analysis on the perspective of using non-commutative conjunctors. This is not a mere mathematical generalization, but a real need since, for instance, when one learns a conjunction from examples it is not unusual that the resulting conjunction does not satisfy commutativity. Different authors have argued in favour of considering non-commutative conjunctors. Actually, there exist quite reasonable examples of non-commutative and even non-associative conjunctors defined on a regular partition of the unit interval.
Hence, the possibility of considering non-commutative conjunctors provides more flexibility and increases the number of applications.
This is one of the properties that the multi-adjoint concept lattice framework offers. Another important feature is that different preferences among the set of attributes or/and objects can be considered.
Moreover, this framework has been extended following different lines and has been applied to define general extensions of fuzzy rough sets theory, to solve fuzzy relation equations, etc.

References

1. P. Julian, G. Moreno, and J. Penabad. On fuzzy unfolding: A multi-adjoint approach. *Fuzzy Sets and Systems*, 154(1):16–33, 2005.
2. J. Medina, M. Ojeda-Aciego, and J. Ruiz-Calviño. Formal concept analysis via multi-adjoint concept lattices. *Fuzzy Sets and Systems*, 160(2):130–144, 2009.
3. J. Medina, M. Ojeda-Aciego, and J. Ruiz-Calviño. On multi-adjoint concept lattices: definition and representation theorem. *Lecture Notes in Artificial Intelligence*, 4390:197–209, 2007.
4. J. Medina, M. Ojeda-Aciego, and P. Vojtáš. Similarity-based unification: a multi-adjoint approach. *Fuzzy Sets and Systems*, 146:43–62, 2004.

* Partially supported by Spanish Ministry of Science and FEDER funds through project TIN09-14562-C05-03 and Junta de Andaluca project P09-FQM-5233.

Data characteristics and their relation to closed patterns discovery algorithms

Engelbert Mephu Nguifo

LIMOS, Clermont University, Blaise Pascal University and CNRS
Clermont-Ferrand, France
`mephu@isima.fr`

Abstract. Closed frequent patterns discovery remains a challenge in data mining. During the last decade, different works on data mining algorithms have based their performance evaluation on one dataset characteristic: its density (or on the contrary its sparseness). The incoming of massive datasets in different applications, points out the important goal to design efficient algorithms. The density measurement have shown to be a direction to reach such goal, especially when dealing with formal context of concept lattices. This talk will discuss this notion and describe some metrics defined to characterize dataset density for patterns discovery purpose.

References

1. Yahia, S.B., Hamrouni, T., Nguifo, E.M.: Frequent closed itemset based algorithms: a thorough structural and analytical survey. SIGKDD Explor. Newsl. **8**(1) (June 2006) 93–104
2. Boley, M., Grosskreutz, H.: Approximating the number of frequent sets in dense data. Knowl. Inf. Syst. **21**(1) (October 2009) 65–89
3. Emilion, R., Lévy, G.: Size of random galois lattices and number of closed frequent itemsets. Discrete Appl. Math. **157**(13) (July 2009) 2945–2957
4. Flouvat, F., Marchi, F., Petit, J.M.: A new classification of datasets for frequent itemsets. J. Intell. Inf. Syst. **34**(1) (February 2010) 1–19
5. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental & Theoretical Artificial Intelligence **14**(2-3) (2002) 189–216
6. Hamrouni, T., Yahia, S.B., Nguifo, E.M.: Looking for a structural characterization of the sparseness measure of (frequent closed) itemset contexts. Information Sciences (0) (2012) –
7. Négrevergne, B., Termier, A., Méhaut, J.F., Uno, T.: Discovering closed frequent itemsets on multicore: Parallelizing computations and optimizing memory accesses. In: HPCS. (2010) 521–528
8. Uno, T., Asai, T., Uchida, Y., Arimura, H.: An efficient algorithm for enumerating closed patterns in transaction databases. In: In Proc. DS '04, LNAI 3245. (2004) 16–31

On the names of implication

Sergei O. Kuznetsov

Higher School of Economics, Moscow
skuznetsov@hse.ru

Abstract. We discuss relationships of attribute implications to various tools in computer science and artificial intelligence: functional dependencies, horn theories, emergent patterns, disjunctive version spaces, and concept-based hypotheses. The intractability of computing implication bases seems to be the main challenge for the use of implications in analyzing large data collections. Alternatives to generation of implication bases such as lazy-learning classification, target-driven generation of classifiers, and sampling are considered.

A Generalized Next-Closure Algorithm – Enumerating Semilattice Elements from a Generating Set

Daniel Borchmann

TU Dresden, Institute of Algebra
daniel.borchmann@mailbox.tu-dresden.de

Abstract. A generalization of the well known Next-Closure algorithm is presented, which is able to enumerate finite semilattices from a generating set. We prove the correctness of the algorithm and apply it on the computation of the intents of a formal context.

1 Introduction

Next-Closure is one of the best known algorithms in Formal Concept Analysis [8] to compute the concepts of a formal context. In its general form it is able to efficiently enumerate the closed sets of a given closure operator on a finite set. This generality might be a drawback concerning efficiency compared to other algorithms like Close-by-One [1,9,11]. On the other hand, the general formulation of Next-Closure widens its field of application. However, there are still applications where Next-Closure might be useful, but is not applicable, because a closure operator on a finite set is not explicitly available. One such example might be the computation of concepts of a fuzzy formal context [3]. In those cases most often an ad hoc variation of Next-Closure can be constructed. The aim of this paper is to provide a generalization of Next-Closure which covers those cases, and may even go beyond them.

As it turns out, Next-Closure is not about enumerating closed sets of a closure operator, even not on an abstract ordered set. The algorithm is merely about enumerating elements of a certain semilattice, given as an operation together with a generating set. This observation shall turn out to be quite natural.

It has to be noted that there have been prior attempts to generalize Next-Closure to a more general setting [7]. But this approach is, as far as the author can tell, not related to the one presented in this paper.

This paper is organized as follows. First of all we shall revisit the original version of Next-Closure, together with the basic definitions. Then we present our generalized version working on semilattices, together with a complete proof of its correctness. Then we show how this generalized form is indeed a generalization of the original Next-Closure. Additionally, we present another algorithm for enumerating the intents of a given formal context, which is very similar to Close-by-One. Finally, we give some outlook on further questions which might be interesting within this line of research.

2 The Next-Closure Algorithm

Before we are going to discuss our generalized form of Next-Closure, let us revisit the original version as it is given in [6,8]. To make our discussion a bit more consistent, we shall allow ourselves a minor deviation from the standard description of the algorithm, which we shall mention explicitly.

Let M be a finite set and let $c : \mathfrak{P}(P) \longrightarrow \mathfrak{P}(P)$ be a function such that

a) c is idempotent, i.e. $c(c(A)) = c(A)$ for all $A \subseteq M$,
b) c is monotone, i.e. if $A \subseteq B$, then $c(A) \subseteq c(B)$ for all $A, B \subseteq M$, and
c) c is extensive, i.e. $A \subseteq c(A)$ for all $A \subseteq M$.

The mapping c is then said to be a *closure operator* on P. A set $A \subseteq M$ is called *closed (with respect to c)* if $A = c(A)$, and the *image of* c is defined as

$$c[\mathfrak{P}(M)] := \{\, c(A) \mid A \subseteq M \,\}.$$

Without loss of generality, let $M = \{\, 1, \ldots, n \,\}$ for some $n \in \mathbb{N}$. For two sets $A, B \in c[\mathfrak{P}(M)]$ with $A \neq B$ and $i \in M$ we say that A is *lectically smaller* than B *at position* i if and only if

$$i = \min(A \mathbin{\Delta} B) \quad \text{and} \quad i \in B,$$

where $A \mathbin{\Delta} B = (A \backslash B) \cup (B \backslash A)$ denotes the *symmetric difference* of A and B. We shall write $A \prec_i B$ if A is lectically smaller than B at position i. Finally, we say that A is *lectically smaller* than B, for $A, B \in c[\mathfrak{P}(M)]$, if $A = B$ or $A \prec_i B$ for some $i \in M$. We shall write $A \preceq B$ in this case.

It has to be noted that, in contrast to our definition, the lectic order is normally defined for all sets $A, B \subseteq M$ in the very same spirit as given above. However, as we shall see, this is not necessary, which is why we have restricted our definition to closed sets only.

Now let us define for $A \in c[\mathfrak{P}(M)]$ and $i \in M$

$$A \oplus i := c(\{\, j \in A \mid j < i \,\} \cup \{\, i \,\}).$$

Then we have the following result.

Theorem 1 (Next-Closure [6]). *Let $A \in c[\mathfrak{P}(M)]$. Then the next closed set $A^+ \in c[\mathfrak{P}(M)]$ after A with respect to the lectic order $\preceq$, if it exists, is given by*

$$A^+ = A \oplus i$$

with $i \in M$ being maximal with $A <_i A \oplus i$.

This is the original version of Next-Closure, as it is given in [6,8]. Therein, the term "next closed set" has the obvious meaning, namely

$$A^+ = \min\nolimits_{\prec}\{\, B \in c[\mathfrak{P}(M)] \mid A \prec B \,\}.$$

Our generalization now starts with the following observation: the set $A \oplus i$ can be seen as the smallest closed set containing both $\{ j \in A \mid j < i \}$ and $\{ i \}$, or equivalently, both $c(\{ j \in A \mid j < i \})$ and $c(\{ i \})$. This means that we can rewrite $A \oplus i$ as

$$A \oplus i = c(\{ j \in A \mid j < i \}) \vee c(\{ i \}),$$

where $X \vee Y$ is the smallest closed set containing both $X, Y \in c[\mathfrak{P}(M)]$, the *supremum of* X *and* Y, which is simply given by $X \vee Y = c(X \cup Y)$. This observation suggests to consider Next-Closure on abstract algebraic structures with a binary operation $\vee$ with some certain properties, namely on *semi-lattices*. To do so we need a more general notion of $c(\{ i \})$, since we do not necessarily deal with subsets, and a more general notion of $\{ j \in A \mid j < i \}$, which likewise might not be expressible in a more general setting. Finally, we need to find a starting point for our enumeration, which is $c(\varnothing)$ in the original description of Next-Closure, but may vary in other cases. Luckily, all this is possible and quite natural, as we shall see in the next section.

3 Generalizing Next-Closure for Semilattices

The aim of this section is to present a generalization of the Next-Closure algorithm that works on semilattices. For this recall that a *semilattice* $\underline{L} = (L, \vee)$ is an algebraic structure with a binary operation $\vee$ which is associative, commutative and idempotent. It is well known that by

$$x \leqslant_{\underline{L}} y :\iff x \vee y = y, \quad (x, y \in L)$$

an order relation on L is defined in such a way that for every two elements $a, b \in L$ the element $a \vee b$ is the least upper bound of both a and b with respect to $\leqslant_{\underline{L}}$.

For the remainder of this section let $\underline{L} = (L, \vee)$ be an arbitrary but fixed semilattice. Furthermore, let $(x_i \mid i \in I)$ be an enumeration of a finite generating set $\{ x_i \mid i \in I \} \subseteq L$ of $\underline{L}$. Finally, let $\leqslant_I$ be a total order on I.

Definition 1. *Let $a, b \in L$ and let $i \in I$. Set*

$$\Delta_{a,b} := \{ j \in I \mid (x_j \leqslant_{\underline{L}} a \text{ and } x_j \nleqslant_{\underline{L}} b) \text{ or } (x_j \nleqslant_{\underline{L}} a \text{ and } x_j \leqslant_{\underline{L}} b) \}.$$

We then define

$$a <_i b :\iff i = \min \Delta_{a,b} \text{ and } x_i \leqslant_{\underline{L}} b.$$

Furthermore we write $a < b$ if $a <_i b$ for some $i \in I$ and write $a \leqslant b$ if $a = b$ or $a < b$.

One can see the similarity of this definition to the one of the lectic order. Here, the set $\Delta_{a,b}$ generalizes $a \mathbin{\Delta} b$ and $x_i \leqslant_{\underline{L}} b$ somehow represents the fact that $i \in b$, or equivalently $\{ i \} \subseteq b$, in the special case of $L = \mathfrak{P}(M)$ and $i \in M$.

Note that if $a <_i b$ and $k \in I$ with $k <_I i$, then

$$x_k \leqslant_{\underline{L}} a \iff x_k \leqslant_{\underline{L}} b.$$

This observation is quite useful and will be used in some of the proofs later on.

The first thing we want to consider now are two easy results stating that $\leqslant$ is a total order relation on L extending $\leqslant_{\underline{L}}$.

Lemma 1. *The relation $<$ is irreflexive and transitive. Furthermore, for every two elements $a, b \in L$ with $a \neq b$, it is either $a < b$ or $b < a$.*

Proof. If $a = b$, then the set $\Delta_{a,b}$ defined above is empty, therefore we cannot have $a <_i a$ for some $i \in I$. This shows the irreflexivity of $<$. Let us now consider the transitivity of $<$. For this let $a, b, c \in L$, $i, j \in I$ and suppose that $a <_i b$ and $b <_j c$. We have to show that $a < c$. Let us consider the following cases.

Case $i <_I j$. We have $x_i \nleqslant_{\underline{L}} a$ and $x_i \leqslant_{\underline{L}} b$ because of $a <_i b$. Due to $i <_I j$ it follows that $x_i \leqslant_{\underline{L}} c$. Suppose that there exists $k \in I$, $k <_I i$ with $x_k \leqslant_{\underline{L}} a$ and $x_k \nleqslant_{\underline{L}} c$. Then if $x_k \nleqslant_{\underline{L}} b$ we would have $x_k \nleqslant_{\underline{L}} a$ because of $k <_I i$, a contradiction. But if $x_k \leqslant_{\underline{L}} b$, then $x_k \leqslant_{\underline{L}} c$ because of $k <_I i <_I j$, again a contradiction. Thus we have shown that $a < c$.

Case $j <_I i$. We have $x_j \nleqslant_{\underline{L}} b$, $x_j \leqslant_{\underline{L}} c$ because of $b <_j c$. Due to $j <_I i$ it follows that $x_j \nleqslant_{\underline{L}} a$. Now if there were a $k \in I$, $k <_I j$ with $x_k \leqslant_{\underline{L}} a$ and $x_k \nleqslant_{\underline{L}} c$, then $x_k \leqslant_{\underline{L}} b$ would imply $x_k \leqslant_{\underline{L}} c$ and $x_k \nleqslant_{\underline{L}} b$ would imply $x_k \nleqslant_{\underline{L}} a$, analogously to the first case, a contradiction. Hence such a k cannot exist and $a < c$.

Case $i = j$. This cannot occur since otherwise $x_i \leqslant_{\underline{L}} b$, because of $a <_i b$, and $x_i \nleqslant_{\underline{L}} b$, because of $b <_i c$, a contradiction.

Overall we have shown that $a < c$ in any case and therefore $<$ is a transitive relation.

Finally let $a, b \in L$ with $a \neq b$. Then because $\{ x_i \mid i \in I \}$ is a generating set, the set $\Delta_{a,b}$ is not empty, since otherwise $a = b$. With $i := \min \Delta_{a,b}$ we either have $a <_i b$ if $x_i \leqslant_{\underline{L}} b$ and $b <_i a$ otherwise.

Lemma 2. *Let $a, b \in L$ with $a \leqslant_{\underline{L}} b$. Then $a \leqslant b$. In particular, if $a \leqslant_{\underline{L}} c$ and $b \leqslant_{\underline{L}} c$ for $a, b, c \in L$, then $a \vee b \leqslant c$.*

Proof. We show $x_i \leqslant_{\underline{L}} a \implies x_i \leqslant_{\underline{L}} b$ for all $i \in I$. This shows $b \nless a$, hence $a \leqslant b$ by Lemma 1. Now if $x_i \leqslant_{\underline{L}} a$, then because of $a \leqslant_{\underline{L}} b$ we see that $x_i \leqslant_{\underline{L}} b$ and the claim is proven.

The next step towards a general notion of Next-Closure is to provide a generalization of $\oplus$.

Definition 2. *Let $a \in L$ and $i \in I$. Then define*

$$a \oplus i := \bigvee_{\substack{j <_I i \\ x_j \leqslant_{\underline{L}} a}} x_j \vee x_i.$$

With all these definitions at hand we are now ready to formulate and prove the promised generalization. For this, we generalize the proof of Next-Closure as it is given in [8, page 67].

Lemma 3. *Let $a, b \in L$ and $i, j \in I$. Then the following statements hold:*

i) $a <_i b, a <_j c, i <_I j \implies c <_i b$.
ii) $a < a \oplus i$ *if* $x_i \nleqslant_{\underline{L}} a$.
iii) $a <_i b \implies a \oplus i \leqslant b$.
iv) $a <_i b \implies a <_i a \oplus i$.

Proof. i) It is $x_i \nleqslant_{\underline{L}} a$ and due to $i <_I j$ we get $x_i \nleqslant_{\underline{L}} c$ as well. Furthermore, $x_i \leqslant_{\underline{L}} b$ because of $a <_i b$. Now if there would exist a $k \in I$ with $k <_I i$ such that $x_k \leqslant_{\underline{L}} c, x_k \nleqslant_{\underline{L}} b$, then $x_k \leqslant_{\underline{L}} a$ because of $k <_I i$ and $x_k \nleqslant_{\underline{L}} a$ because of $k <_I i <_I j$, a contradiction. With the same argumentation a contradiction follows from the assumption that there exists a $k \in I$, $k <_I i$ with $x_k \nleqslant_{\underline{L}} c, x_k \leqslant_{\underline{L}} b$. In sum we have shown $c <_i b$, as required.

ii) We have $x_i \nleqslant_{\underline{L}} a$ and $x_i \leqslant_{\underline{L}} a \oplus i$. Furthermore, for $k \in I$, $k <_I i$ and $x_k \leqslant_{\underline{L}} a$ we have $x_k \leqslant_{\underline{L}} a \oplus i$ by definition. This shows $a < a \oplus i$.

iii) Let $a <_k b$ for some $k \in I$. Then $\bigvee_{j <_I k, x_j \leqslant_{\underline{L}} a} x_j \leqslant_{\underline{L}} b$ and $x_k \leqslant_{\underline{L}} b$, hence with Lemma 2 we get $a \oplus k \leqslant b$.

iv) Let $a <_i b$. Then $x_i \nleqslant_{\underline{L}} a$ and with (ii) we get $a < a \oplus i$. By (iii), $a \oplus i \leqslant b$. If for $k \in I$, $k <_I i$ it holds that $x_k \leqslant_{\underline{L}} a \oplus i$ and $x_k \nleqslant_{\underline{L}} a$, then we also have $x_k \leqslant_{\underline{L}} a \oplus i \leqslant_{\underline{L}} b$, i.e. $x_k \leqslant_{\underline{L}} b$, contradicting the minimality of i.

Theorem 2 (Next-Closure for Semilattices). *Let $a \in L$. Then the next element $a^+ \in L$ with respect to $<$, if it exists, is given by*

$$a^+ = a \oplus i$$

with $i \in I$ being maximal with $a <_i a \oplus i$.

Proof. Let

$$a^+ = \min_< \{ b \in L \mid a < b \}$$

be the next element after a with respect to $<$. Then $a <_i a^+$ for some $i \in I$ and by Lemma 3.iv we get $a <_i a \oplus i$ and with Lemma 3.iii we see $a \oplus i \leqslant a^+$, hence $a \oplus i = a^+$. The maximality of i follows from Lemma 3.i.

To find the correct element $i \in I$ such that $a^+ = a \oplus i$ we can utilize Lemma 3.ii. Because of this result, only elements $i \in I$ with $x_i \nleqslant_{\underline{L}} a$ have to be considered, a technique which is also known for the original form of Next-Closure.

However, to make the above theorem practical for enumerating the elements of a certain semilattice, one has to start with some element, preferably the smallest element in L with respect to $\leqslant$. This element must also be minimal in L with respect to $\leqslant_{\underline{L}}$, by Lemma 2. Since $\{ x_i \mid i \in I \}$ is a generating set of $\underline{L}$, and $a \leqslant a \vee b$ for all $a, b \in L$, the minimal elements of L with respect to $\leqslant_{\underline{L}}$ must be among the elements x_i, $i \in I$. So to find the first element of $\underline{L}$ with respect to $\leqslant$, find all minimal elements in $\{ x_i \mid i \in I \}$ and choose the smallest element with respect to $\leqslant$ from them. But because of $x_i \leqslant x_j$ if and only if $j <_I i$, one just has to take the largest index j with respect to $<_I$ to find the smallest element in L with respect to $\leqslant$.

As a final remark for this section note that the set $\{ x_i \mid i \in I \}$ must always include the $\vee$-irreducible elements of $\underline{L}$. These are all those elements $a \in L$ that cannot be represented as a join of other elements, or, equivalently,

$$\{ b \in L \mid b <_{\underline{L}} a \} = \varnothing \quad \text{or} \quad \bigvee_{b <_{\underline{L}} a} b <_{\underline{L}} a.$$

It is also easy to see that the set of $\vee$-irreducible elements of $\underline{L}$ is also sufficient, i. e. that it is a generating set of $\underline{L}$.

4 Computing the Intents of a Formal Context

We have seen an algorithm that is able to enumerate the elements of a semilattice from a given generating set. We have also claimed that this is a generalization of Next-Closure, which we want to demonstrate in this section. Furthermore, we want to give another example of an application of this algorithm, namely the computation of the intents of a given formal context.

Firstly, let us reconstruct the original Next-Closure algorithm from Theorem 2 and the corresponding definitions. For this let M be a finite set and let c be a closure operator on $M = \{ 0, \dots, n-1 \}$, say. We then apply Theorem 2 to the semilattice $\underline{P} = (c[\mathfrak{P}(M)], \vee)$. We immediately see that $\leqslant_{\underline{P}} = \subseteq$ and that $<_i$ is the usual lectic order on $\underline{P}$. Then the set

$$\{ c(\{ i \}) \mid i \in M \} \cup \{ c(\varnothing) \}$$

is a finite generating set of $\underline{P}$ and we can define $x_i := c(\{ i \})$ and $x_n := c(\varnothing)$, i.e. $I = \{ 0, \dots, n \}$. For a closed set $A \subseteq M$ and $i \in I$ then follows

$$\begin{aligned} A \oplus i &= \bigvee_{\substack{j<i \\ x_j \subseteq A}} x_i \vee x_i \\ &= c(\bigcup_{\substack{j<i \\ x_j \subseteq A}} x_j) \vee x_i \\ &= c(\bigcup \{ c(\{ j \}) \mid j < i, j \in A \}) \vee c(\{ i \}) \\ &= c(\{ j \mid j < i, j \in A \}) \vee c(\{ i \}) \\ &= c(\{ j \mid j < i, j \in A \} \cup \{ i \}) \end{aligned}$$

which is the original definition of $\oplus$ for Next-Closure. Furthermore, it is $A \oplus n = A$ since $c(\varnothing) \subseteq A$ for each closed set A. We therefore do not need to consider x_n when looking for the next closed set, and indeed, the only reason why $x_n = c(\varnothing)$ has been included is that it is the smallest closed set in $\underline{P}$. All in all, we see that Next-Closure is a special case of Theorem 2.

However, for a closure operator c on a finite set M it seems more natural to consider the semilattice $\underline{P} = (c[\mathfrak{P}(M)], \cap)$, because the intersection of two closed sets of c again yields a closed set of c. One sees that $\leqslant_{\underline{P}} = \supseteq$. As a generating set we take the set of $\cap$-irreducible elements $\{ X_i \mid i \in G \}$ for some index set G. Let $A, B \in c[\mathfrak{P}(M)]$ and let $<_G$ be a linear ordering on G. Then $A < B$ if and only if there exists $i \in G$ such that

$$i = \min\{ j \in G \mid (X_i \supseteq A, X_i \not\supseteq B) \text{ or } (X_i \not\supseteq A, X_i \supseteq B) \} \quad \text{and} \quad X_i \supseteq B$$

and $\oplus$ is just given by

$$A \oplus i = \bigcap_{\substack{j <_G i \\ X_j \supseteq A}} X_j \cap X_i.$$

Now note that $\oplus$ does not need the closure operator c anymore. This means that if the computation of c is very costly and the $\cap$-irreducible elements (or a superset thereof) is known, this approach might be much more efficient. In general, however, it is not known how to efficiently determine the $\cap$-irreducible closed sets of c. But if c is given as the $\cdot''$ operator of a formal context, these irreducible elements can be determined quickly [8]. We shall describe this idea in more detail.

Let G and M be two finite sets and let $J \subseteq G \times M$. We then call the triple $\mathbb{K} := (G, M, J)$ a *formal context*, G the *objects* of the formal context and M the *attributes* of the formal context. For $g \in G$ and $m \in M$ we write $g\ J\ m$ for $(g, m) \in J$ and say that object g *has* attribute m.

Let $A \subseteq G$ and $B \subseteq M$. We then define the *derivations* of A and B to be

$$\begin{aligned} A' &:= \{ m \in M \mid \forall g \in A : g\ J\ m \} \\ B' &:= \{ g \in G \mid \forall m \in B : g\ J\ m \}. \end{aligned}$$

Then the $\cdot''$ operator is just the twofold derivation of a given set of attributes. It turns out that this is indeed a closure operator, and that every closure operator can be represented as a $\cdot''$ operator of a suitable formal context [4,8]. The closed sets of $\cdot''$, i.e. all sets $B \subseteq M$ with $B = B''$, are called the *intents* of $\mathbb{K}$ and shall be denoted by $\operatorname{Int}(\mathbb{K})$. It is clear from the previous remarks that $(\operatorname{Int}(\mathbb{K}), \cap)$ is a semilattice.

The advantage of representing a closure operator is that the $\cap$-irreducible elements of $(\operatorname{Int}(\mathbb{K}), \cap)$ can be directly read off from the format context. As discussed in [8], the set

$$\{ \{ g \}' \mid g \in G \}$$

contains the irreducible elements we are looking for, except M (note that the order relation on $(\operatorname{Int}(\mathbb{K}), \cap)$ is $\supseteq$.) Furthermore, it is possible to omit certain objects g from $\mathbb{K}$ without changing $\operatorname{Int}(\mathbb{K})$. Every object $g \in G$ can be omitted from $\mathbb{K}$ for which the set $\{ g \}'$ is either equal to M or can be represented as a proper intersection of other sets $\{ g_1 \}', \dots, \{ g_n \}'$ for some elements $g_1, \dots, g_n \in G$. It is also clear that if there exist two distinct objects g_1 and g_2 with $\{ g_1 \}' = \{ g_2 \}'$, that we can remove one of them without changing $\operatorname{Int}(\mathbb{K})$. A formal context for which no such objects exist is called *object clarified* and *object reduced.* If $\mathbb{K} = (G, M, J)$ is an object clarified and object reduced formal context, then the

Listing 1.1. Compute the Next Intent of a Formal Context

```
define next-intent($\mathbb{K} = (G, M, J), A$)
  for $g \in G$, descending
    if $\{ g \}' \not\supseteq A$ then
      let ($B := A \oplus g$)
        if $\forall h \in G, h <_G g, \{ h \}' \not\supseteq A : \{ h \}' \not\supseteq B$ then
          return $B$
        end if
      end let
    end if
  end for
  return nil
end
```

set $\{ \{ g \}' \mid g \in G \}$ is exactly the set of $\cap$-irreducible intents of $\mathbb{K}$, except for the set M.

The algorithm described above now takes the following form when applied to the semilattice $(\operatorname{Int}(\mathbb{K}), \cap)$. As index set we choose the set G of object of the given formal context, ordered by $<_G$. For every object $g \in G$ we set $x_g := \{ g \}'$. Then the set $\{ x_g \mid g \in G \}$ is a generating set of the semilattice $(\operatorname{Int}(\mathbb{K}) \setminus \{ M \}, \cap)$, which we want to enumerate (since we get the set M for free). For A being an intent of $\mathbb{K}$ and $g \in G$ we have

$$A \oplus g := \bigcap_{\substack{\{ h \}' \supseteq A \\ h <_G g}} \{ h \}' \cap \{ g \}'$$

and as the first intent we take M. For an intent $A \subseteq M$ of $\mathbb{K}$ we then have to find the maximal object $g \in G$ (with respect to $<_G$) such that $A <_g A \oplus g$. This is equivalent to g being maximal with $\{ g \}' \not\supseteq A$ and $\forall h \in G, h <_G g : \{ h \}' \supseteq A \iff \{ h \}' \supseteq A \oplus g$. However, the direction "$\Longrightarrow$" is clear, hence we only have to ensure

$$\forall h \in G, h <_G g : \{ h \}' \not\supseteq A \implies \{ h \}' \not\supseteq A \oplus g.$$

All these considerations yield the algorithm shown in Listing 1.1. Of course, the derivations of the form $\{ g \}'$ should not be computed every time they are needed but rather stored somewhere for reuse.

Let us simplify Listing 1.1 a bit further. If A is an intent of $\mathbb{K}$, then for $g \in G$ it is true that

$$\{ g \}' \not\supseteq A \iff g \notin A'.$$

This is because

$$\begin{aligned} \{ g \}' \supseteq A &\implies \{ g \}'' \subseteq A' \\ &\implies g \in A' \end{aligned}$$

Listing 1.2. Simplified version of Listing 1.1

```
define next-intent($\mathbb{K} = (G, M, J), A$)
  for $g \in G$, descending
    if $g \notin A'$ then
      let ($B := A \oplus g$)
        if $B' \cap G_g \subseteq A' \cap G_g$ then
          return $B$
        end if
      end let
    end if
  end for
  return nil
end
```

and

$$\begin{aligned} g \in A' &\implies \{ g \} \subseteq A' \\ &\implies \{ g \}' \supseteq A'' = A, \end{aligned}$$

and therefore $\{ g \}' \supseteq A \iff g \in A'$, i. e. $\{ g \}' \not\supseteq A \iff g \notin A'$. Using this, we can simplify the condition

$$\forall h \in G, h <_G g, \{ h \}' \not\supseteq A : \{ h \}' \not\supseteq B$$

to

$$\forall h \in G, h <_G g : h \in B' \implies h \in A'$$

or equivalently to $B' \cap G_g \subseteq A' \cap G_g$, where $G_g := \{ h \in G \mid h <_G g \}$. This leads to the algorithm of Listing 1.2.

Curiously enough, this algorithm is now very similar to Close-by-One. To make this similarity more apparent, let us give a brief description of the algorithm Close-by-One. In contrast to Next-Closure, which enumerates the intents of the formal context $\mathbb{K}$, Close-by-One enumerates the *formal concepts* of $\mathbb{K}$. These are pairs (C, D) of sets $C \subseteq G, D \subseteq M$ such that $C' = D$ and $D' = C$. The set of all formal concepts of $\mathbb{K}$ is denoted by $\mathfrak{B}(\mathbb{K})$. The formal concepts $\mathfrak{B}(\mathbb{K})$ of $\mathbb{K}$ can be ordered by

$$(C_1, D_1) \leqslant (C_2, D_2) \iff C_1 \subseteq C_2 (\iff D_2 \subseteq D_1).$$

It turns out that the set $(\mathfrak{B}(\mathbb{K}), \leqslant)$ forms a *(complete) lattice*, i. e. an ordered set such that for each set of formal concepts there exists a smallest upper and a largest lower bound. This lattice is also called the *concept lattice* of $\mathbb{K}$.

Close-by-One now performs a depth-first search in this lattice to compute all formal concepts of $\mathbb{K}$. Following the description of [10], the main part of the work is done by the function `generate-from`, as it is described in Listing 1.3. To simplify the description of this function, we again assume that the set M of

Listing 1.3. Close-by-One [10]

```
 0 define generate-from($\mathbb{K} = (G, M, J)$, $(C, D)$, $\ell$)
 1   output $(C, D)$
 2
 3   if $D = M$ or $\ell > |M|$ then
 4     return
 5   end if
 6
 7   for $m \in \{ \ell, \dots, n \}$, ascending
 8     if $m \notin D$ then
 9       let ($E := C \cap \{ m \}'$,
10            $F := E'$)
11         if $F \cap M_m \subseteq D \cap M_m$ then
12           generate-from($(E, F)$, $m + 1$)
13         end if
14       end let
15     end if
16   end for
17 end
18
19 define all-concepts($\mathbb{K} = (G, M, J)$)
20   generate-from($\mathbb{K}$, $(\varnothing', \varnothing'')$, $1$)
21 end
```

attributes of $\mathbb{K}$ is just the set $M = \{ 1, \dots, n \}$. Furthermore, if $m \in M$, we define $M_m := \{ 1, \dots, m - 1 \}$.

We can now spot some similarities between the functions `next-intent` from Listing 1.2 and `generate-from`. The most striking similarity to note is the occurrence of the same tests in both functions: "$B' \cap G_g \subseteq A' \cap G_g$" in line 4 of Listing 1.2, and "$F \cap M_m \subseteq D \cap M_m$" in line 11 of Listing 1.3. If we substitute the definitions of the involved variables, these tests get the following form:

`next-intent`: $(A \cap \{ g \}')' \cap G_g \subseteq A' \cap G_g$
`generate-from`: $(C \cap \{ m \}')' \cap M_m \subseteq C' \cap M_m$

So except that we consider sets of objects in the function `next-intent` and sets of attributes in the function `generate-from`, both test conditions are the same.

Still, the functions `next-intent` and `generate-from` are very different in how they compute the next intent and formal concept, respectively. While `next-intent` computes for a given intent just a new one, `generate-from` performs a depth-first search and enumerate *all* formal concepts of $\mathbb{K}$.

5 Conclusion

We have seen a natural generalization of the Next-Closure algorithm to enumerate elements of a semilattice from a generating set. We have proven the algorithm to be correct and applied it to the standard task of computing the intents of a given formal context, yielding a another algorithm to accomplish this. This algorithm turned out to have a lot of similarity to Close-by-One.

There are still some interesting ideas one might want to look at.

Firstly, a variation of the original Next-Closure algorithm is able to compute the canonical base of a formal context, a very compact representation of its implicational knowledge. It would be interesting to know whether the generalization given in this paper gives more insight into the computation, and therefore into the nature of the canonical base. This is, however, quite a vague idea.

Secondly, the algorithm discussed above to compute the intents of a formal context enumerates them in a certain order, which might or might not be a lectic one. Understanding this order relation might be fruitful, especially with respect to the complexity results obtained recently which consider enumerating *pseudo-intents* of a formal context in lectic order [5].

Thirdly, there exists a variant of the Next-Closure algorithm that is able to compute intents of a formal context *up to symmetry* [8, Theorem 51]. Given a set Γ of *automorphisms* of a formal context, this variant is able to compute for each orbit of intents under Γ exactly one representative. It would be interesting to know whether we can find a variant of our generalized Next-Closure algorithm that accomplishes the same thing.

Finally, and more technically, it might be interesting to look for other applications of our general algorithm. As already mentioned in the introduction, the enumeration of fuzzy concepts of a fuzzy formal context might be worth investigating (and it might be interesting comparing it to [2]).

Acknowledgments

The author is grateful for the useful comments of the anonymous reviewers, especially for pointing out the similarity of Listing 1.1 to Close-by-One, which the author had overlooked.

References

1. Simon Andrews. In-Close, a fast algorithm for computing formal concepts. In Sebastian Rudolph, Frithjof Dau, and Sergei O. Kuznetsov, editors, *ICCS Supplementary Proceedings*, Moscow, 2009.
2. Radim Belohlávek, Bernard De Baets, Jan Outrata, and Vilém Vychodil. Computing the Lattice of All Fixpoints of a Fuzzy Closure Operator. *IEEE T. Fuzzy Systems*, 18(3):546–557, 2010.
3. Radim Belohlávek and Vilém Vychodil. Attribute Implications in a Fuzzy Setting. In Rokia Missaoui and Jürg Schmid, editors, *ICFCA*, volume 3874 of *Lecture Notes in Computer Science*, pages 45–60. Springer, 2006.
4. Daniel Borchmann. Decomposing Finite Closure Operators by Attribute Exploration. In *ICFCA 2011, Supplementary Proceedings*, 2011.

5. Felix Distel. Hardness of Enumerating Pseudo-intents in the Lectic Order. In Léonard Kwuida and Baris Sertkaya, editors, *ICFCA*, volume 5986 of *Lecture Notes in Computer Science*, pages 124–137. Springer, 2010.
6. Bernhard Ganter. Two basic algorithms in concept analysis. FB4-Preprint Nr. 831, 1984.
7. Bernhard Ganter and Klaus Reuter. Finding all closed sets: A general approach. *Order*, 8(3):283–290, 1991.
8. Bernhard Ganter and Rudolph Wille. *Formal Concept Analysis: Mathematical Foundations.* Springer, Berlin-Heidelberg, 1999.
9. Sergei O. Kuznetsov. A fast algorithm for computing all intersections of objects in a finite semi-lattice. *Automatic Documentation and Mathematical Linguistics*, 27:11–21, 1993.
10. Jan Outrata and Vilém Vychodil. Fast algorithm for computing fixpoints of galois connections induced by object-attribute relational data. *Inf. Sci.*, 185(1):114–127, 2012.
11. Vilém Vychodil, Petr Krajča, and Jan Outrata. Advances in algorithms based on CbO. In Marzena Kryszkiewicz and Sergei Obiedkov, editors, *Concept Lattices and Their Application*, pages 325–337, 2010.

Hermes: an efficient algorithm for building Galois sub-hierarchies

Anne Berry[1], Marianne Huchard[2], Amedeo Napoli[3], and Alain Sigayret[1]

[1] LIMOS - CNRS UMR 6158 - Université Clermont-Ferrand II (France)[**]
{berry, sigayret}@isima.fr
[2] LIRMM - CNRS UMR 5506 - Université de Montpellier II - Montpellier (France)
huchard@lirmm.fr
[3] LORIA - CNRS UMR7503 - Vandoeuvre-lès-Nancy (France)
amedeo.napoli@loria.fr

Abstract. Given a relation $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$ on a set $\mathcal{O}$ of objects and a set $\mathcal{A}$ of attributes, the Galois sub-hierarchy (also called AOC-poset) is the partial order on the introducers of objects and attributes in the corresponding concept lattice. We present a new efficient algorithm for building a Galois sub-hierarchy which runs in $O(min\{nm, n^{\alpha}\})$, where n is the number of objects or attributes, m is the size of the relation, and n^{α} is the time required to perform matrix multiplication (currently $\alpha = 2.376$).

1 Introduction

Galois lattices (also called concept lattices) are a powerful tool for data modeling. Such a lattice is built on a relation between a set of objects and a set of attributes. The main drawback of this structure is that it may have an exponential size in the number n of objects or attributes. A canonical sub-order of the lattice, its Galois sub-hierarchy (GSH, also called AOC-Poset), of much smaller size, is recommended whenever possible. This GSH preserves only the key elements of the lattice: object-concepts and attribute-concepts (also called introducers); the number of these key elements is at most equal to the total number n of objects and attributes.

Galois sub-hierarchies were introduced in software engineering by Godin et al. [10] for class hierarchy reconstruction and successfully applied in later research work (see e.g. [15]). The AOC-poset (Attribute/Object Concepts poset [9]) has been used in applications of FCA to non-monotonic reasoning and domain theory [12], and to produce classifications from linguistic data [18, 20]. Specific parts of the GSH (mainly attribute-concepts) have been used in several works, including approaches for refactoring a class hierarchy [14] and recently for extracting a feature tree from a set of products in Software Product Lines [21].

[**] Research partially supported by the French Agency for Research under the DEFIS program TODO, ANR-09-EMER-010.

Three algorithms for building GSH already exist: ARES [7], CERES [14], and PLUTON [1]. Each of them has a time complexity of $O(n^3)$, and they are somewhat complicated. Their comparative experimental running times were investigated in [1].

In this paper, we present a new algorithm for building Galois sub-hierarchies, which we call HERMES, with a better complexity. HERMES runs in $O(nm)$ time, where m is the size of the relation, and is very easy to understand and implement. With more effort invested in the implementation, Hermes can be made to run in $O(n^{\alpha})$ (*i.e.* $O(n^{2.376})$) time, which is the time for performing matrix multiplication. Hermes works by simplifying and then extending the input relation into a relation which contains in a compact fashion all the necessary information on the elements of the GSH.

The paper is organized as follows: after this introduction, we give some notations and definitions. Section 3 briefly outlines how previous algorithms work. Section 4 proves some preliminary results and presents the algorithmic tools necessary to ensure our good complexity. Section 5 describes and analyzes in detail the successive steps of our algorithmic process. Section 6 gives the algorithm. Section 7 describes the special case for chordal bipartite relations, where the final relation can easily be obtained in $O(n^2)$ time. We conclude in Section 8.

2 Definitions and notations

The different communities handling Galois lattices and Galois sub-hierarchies use various notations. Here, we will use algebraic notations detailed below.

Given two finite sets $\mathcal{O}$ (of 'objects') and $\mathcal{A}$ (of 'attributes'), a binary **relation** $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$ indicates which objects of $\mathcal{O}$ are associated with which attributes of $\mathcal{A}$. $\mathcal{O}$ is called the **starting set** of the relation. We will denote $n = |\mathcal{O}| + |\mathcal{A}|$ and $m = |\mathcal{R}|$. For $(x, y) \in \mathcal{R}$, we will say that x is an **antecedent** of y, and y is an **image** of x. For $x \in \mathcal{O}$, $\mathcal{R}(x) = \{y \in \mathcal{A} \,|\, (x, y) \in \mathcal{R}\}$ is the **image set** (**row**) of x, and for $y \in \mathcal{A}$, $\mathcal{R}^{-1}(y) = \{x \in \mathcal{O} \,|\, (x, y) \in \mathcal{R}\}$ is the **antecedent set** (**column**) of y. Note that notation x' is used in FCA [9] for $R(x)$ and $R^{-1}(x)$. The term **line** will be indifferently used for row and column. The triple $(\mathcal{O}, \mathcal{A}, \mathcal{R})$ is called a **context**.

A maximal rectangle of $\mathcal{R}$, also called a **concept**, is a maximal Cartesian sub-product of $\mathcal{R}$, *i.e.* $X \times Y$ such that $\forall x \in X$, $\forall y \in Y$, $(x, y) \in \mathcal{R}$ and $\forall w \in \mathcal{O} - X$, $\exists y \in Y \mid (w, y) \notin \mathcal{R}$ and $\forall z \in \mathcal{A} - Y$, $\exists x \in X \mid (x, z) \notin \mathcal{R}$. X is called the **extent** and Y the **intent** of concept $X \times Y$, which is denoted (X, Y). In our examples, we may omit set brackets when the meaning is clear. The extent and intent of concept C will be denoted **Extent(C)** and **Intent(C)**. The concepts, ordered by inclusion on their extents (or dually by inclusion on their intents) form a lattice $\mathcal{L}(\mathcal{R})$ called a **Galois lattice** or a **concept lattice**. For two concepts C and C', $C <_{\mathcal{L}(\mathcal{R})} C'$ will denote Extent(C)$\subset$Extent(C'). A lattice is represented by its **Hasse diagram**, where reflexivity and transitivity edges are omitted.

An **object-concept** is a concept C_x which *introduces* some object x: x is in the extent of C_x but is not in the extent of any smaller concept $C' <_{\mathcal{L}(\mathcal{R})} C_x$. Dually, an **attribute-concept** is a concept C_y which *introduces* some attribute y: y is in the intent of C_y but is not in the intent of any greater concept $C' >_{\mathcal{L}(\mathcal{R})} C_y$. Thus, the intent of object-concept C_x is $\mathcal{R}(x)$, and the extent of attribute-concept C_y is $\mathcal{R}^{-1}(y)$. Object-concepts and attribute-concepts are also called **introducer concepts** or simply **introducers**. Objects are introduced from bottom to top and attributes from top to bottom in $\mathcal{L}(\mathcal{R})$. A given concept may introduce several objects and/or attributes. Note that [9] uses arrow relations to characterize the relationship between attribute-concepts and object-concepts, but without referring to Galois sub-hierarchies.

A relation is said to be **clarified** when it has no identical lines. A relation is said to be **reduced** when it is clarified and has no line which is the intersection of several other lines. When a relation is reduced, the irreducible elements of the lattice are exactly the introducers, whereas in a non-reduced relation there will be extra introducers.

$\mathcal{H}(\mathcal{R})$ denotes the **Galois sub-hierarchy** (GSH) of relation $\mathcal{R}$, defined by the set of introducer concepts ordered as in $\mathcal{L}(\mathcal{R})$. $\mathcal{H}(\mathcal{R})$ is then a sub-order of $\mathcal{L}(\mathcal{R})$. The elements of $\mathcal{H}(\mathcal{R})$ are generally labeled by the objects and/or attributes they introduce, defining the **simplified labeling**. The same simplified labeling applies to $\mathcal{L}(\mathcal{R})$, in which some concepts may have an empty label. $<_{\mathcal{H}(\mathcal{R})}$ will be used to compare two elements of $\mathcal{H}(\mathcal{R})$, as $<_{\mathcal{L}(\mathcal{R})}$ is used for $\mathcal{L}(\mathcal{R})$.

A **linear extension** of a partially ordered set P is a total order in which P is included.

Running example. Figure 1 shows the Galois lattice $\mathcal{L}(\mathcal{R})$ (as drawn by Context Explorer [24]) and the Galois sub-hierarchy $\mathcal{H}(\mathcal{R})$ of relation $\mathcal{R}$. In $\mathcal{L}(\mathcal{R})$, concept $(1, acdeg)$ introduces 1 (simplified label: 1), concept $(1346, c)$ introduces c (simplified label: c), and concept $(3, abcdfg)$ introduces 3 and b (simplified label: $3, b$). All these concepts are in $\mathcal{H}(\mathcal{R})$; concept $(13, acdg)$ introduces nothing (simplified label empty) and as such is not in $\mathcal{H}(\mathcal{R})$.

$\mathcal{R}$	a	b	c	d	e	f	g
1	×		×	×	×		×
2	×				×	×	×
3	×	×	×	×		×	×
4			×			×	
5				×			
6			×	×			
7	×				×		×
8	×				×	×	×

3 Previous algorithms

We present a brief description of the existing algorithms for building Galois sub-hierarchies; all run in $O(n^3)$ time, where n stands for the number of objects and attributes in the input relation. The reader is referred to the corresponding publications for detailed descriptions and to [1] for a comparative experimental study of those algorithms.

Pruned lattice. [10]
Pruned lattice is the name given to a Galois sub-hierarchy by [10] which is, to our best knowledge, the first paper defining this structure. [10] considers a specific

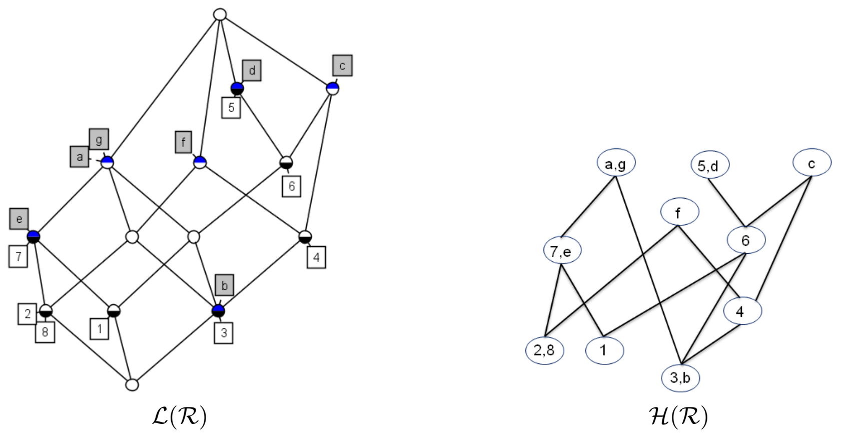


Fig. 1. Lattice $\mathcal{L}(\mathcal{R})$ and Galois sub-hierarchy $\mathcal{H}(\mathcal{R})$, both with the simplified labeling, for our running example.

case where each object owns a specific attribute (not owned by the others). A class inheritance hierarchy is flattened in a table that associates the classes with their members (class attributes and methods). The hierarchy is then rebuilt in a better factorized way, eliminating redundancy.

Pluton. [1]
This algorithm is composed of three successive processes: TomThumb, ToLinext, and ToGSH. TomThumb [3] produces an ordered list of the simplified labels of extents and intents, which maps to a linear extension of the GSH. ToLinext then searches this list to merge consecutive pairs consisting of a simplified extent and a simplified intent belonging to the same concept. Finally, ToGSH computes the edges of the Hasse diagram of the GSH.

Ceres. [14]
This algorithm computes at the same time the elements of the GSH and its Hasse diagram. The elements are computed in an order which maps to a linear extension of the Galois sub-hierarchy. In a first stage, the columns of the relation are sorted by decreasing number of crosses to generate the introducers by decreasing extent size. In the second stage, the strategy is twofold: compute the attribute-concepts by groups sharing the same extent, and add object-concepts when their intent is covered by the intents of the attribute-concepts already computed. The edges of the hierarchy are determined on-the-fly.

Ares. [7]
This algorithm is incremental: given a Galois sub-hierarchy and a new object with its attribute set S, the hierarchy is modified to include this new object. For this, the initial GSH is traversed using a linear extension. If I denotes the intent of the current (explored) concept, then four main cases may occur and the GSH will be updated accordingly: $I = S$, $I \subset S$, $I \supset S$, or I and S are not

comparable by set inclusion. If during exploration, the algorithm did not find an initial concept whose intent is S, a new concept is created. For every modification of the Hasse diagram, the algorithm removes newly created transitivity edges. At the same time, for each modified intent, the algorithm checks if some concept has a simplified label which is empty and removes such concept.

4 Preliminary results and algorithmic tools

4.1 Some preliminary results

The following theorem will help order the introducers:

Theorem 1.
Let C_x be the introducer of $x \in \mathcal{O}$ and C_y be the introducer of $y \in \mathcal{A}$.
– $C_x \leq_{\mathcal{H}(\mathcal{R})} C_y$ iff $(x, y) \in \mathcal{R}$.
– $C_x \geq_{\mathcal{H}(\mathcal{R})} C_y$ iff Intent(C_x)$\subseteq$Intent(C_y) iff Extent(C_x)$\supseteq$Extent(C_y).
In this case, $(x, y) \notin \mathcal{R}$ except if $C_x = C_y$.
– Otherwise, C_x and C_y are not comparable.

[22] introduced the notion of **domination** which originates from graph theory. Domination in a relation stemmed from the concept of domination in the co-bipartite graph which is the complement of the bipartite graph induced by the relation.

An attribute $y \in \mathcal{A}$ is said to **dominate** an attribute $z \in \mathcal{A}$ in $\mathcal{R}$ if the antecedent set of y is included in the antecedent set of z: $\mathcal{R}^{-1}(y) \subseteq \mathcal{R}^{-1}(z)$; the corresponding relation is denoted $Dom_{\mathcal{A}}$. When the inclusion is strict, the domination is said to be strict. For $y \in \mathcal{A}$, $Dom_{\mathcal{A}}(y) = \{z \in \mathcal{A} \,|\, \mathcal{R}^{-1}(y) \subseteq \mathcal{R}^{-1}(z)\}$. This preorder defines the way attributes label the concepts of $\mathcal{H}(\mathcal{R})$ from top (the dominat**ing** attributes) to bottom (the dominat**ed** attributes).

A domination relation, $Dom_{\mathcal{O}}$, can also be defined between objects by inclusion of their image sets: $\forall w \in \mathcal{O}$, $Dom_{\mathcal{O}}(w) = \{x \in \mathcal{O} \,|\, \mathcal{R}(w) \subseteq \mathcal{R}(x)\}$. The label of $\mathcal{H}(\mathcal{R})$ will be set from bottom (the dominating objects) to top (the dominated objects), according to the dual behavior of objects and attributes in concepts.

Theorem 2. *[4]*
Endowed with domination relation $Dom_{\mathcal{A}}$, the set of attribute-concepts of $\mathcal{R}$ forms a sub-order of $\mathcal{H}(\mathcal{R})$: for $y, z \in \mathcal{A}$, the introducer of y is smaller than or equal to the introducer of z iff $(y, z) \in Dom_{\mathcal{A}}$.
Endowed with domination relation $Dom_{\mathcal{O}}$, the set of object-concepts of $\mathcal{R}$ forms a sub-order of $\mathcal{H}(\mathcal{R})$ and $\mathcal{L}(\mathcal{R})$: for $w, x \in \mathcal{O}$, the introducer of w is greater than or equal to the introducer of x iff $(w, x) \in Dom_{\mathcal{O}}$.

Example. In our running example, $R^{-1}(b) = \{3\} \subset R^{-1}(a) = \{1, 2, 3, 7, 8\}$; attribute b dominates attribute a and the introducer of b is smaller than the introducer of a, as shown in Figure 1. $Dom_{\mathcal{A}}(b) = \{a, b, c, d, f, g\}$. $R(6) = \{c, d\} \subset R(1) = \{a, c, d, e, g\}$; object 6 dominates object 1 and the introducer of 6 is greater than the introducer of 1. $Dom_{\mathcal{O}}(6) = \{1, 3\}$.

4.2 Algorithmic tools

We will need two processes for our complexity results.

The first process is to rapidly recognize lines of a relation which are equal, which corresponds to the clarification of context $(\mathcal{O}, \mathcal{A}, \mathcal{R})$. This can be done in linear time $O(|\mathcal{R}|)$ by a process of partition refinement, as proved by [13] for undirected graphs, and detailed as applied to relations [2]. Thus, in linear time, one can merge all sets of lines which are equal. Note that after this process, the domination on attributes (*resp.* objects) will be a strict order.

The second tool we use extensively enables us to decide which lines (rows or columns) are properly included in another, or in other words determines a domination order. This can be done using the tripartite directed graph introduced by Bordat [5]. Computing the transitive edges of this graph will result into the domination order on objects or attributes, depending on how the graph is initially defined [2]. Computing the transitive closure of a graph can be performed in the same time as Matrix Multiplication, with a time complexity of $O(n^\alpha)$, where α is currently 2.376 [6]. However, the $O(n^{2.376})$ algorithm [6] for Matrix Multiplication is not often used, as it is difficult to implement. It is easy to compute the domination order in $O(nm)$ time, as each line can be compared to all the other lines in linear time.

The last step of our algorithm requires a transitive reduction which consists in removing all the transitivity edges of a partial order. This problem has the same time complexity as the equivalent problem of transitivity closure and can also be performed in the same time as matrix multiplication.

5 Algorithmic process

Our algorithm works in five simple steps:

1. Clarify the input relation $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$ into a relation $\mathcal{R}_c$ where no two lines (rows or columns) are identical in order to avoid redundancy.
2. Compute the domination relation $Dom_{\mathcal{A}}$ between attributes (*i.e.* decide which columns of $\mathcal{R}_c$ are included into which other columns).
3. Compute a new relation $\mathcal{R}_{ce}$, obtained by appending $Dom_{\mathcal{A}}$ to $\mathcal{R}_c$, and simplify $\mathcal{R}_{ce}$ into $\mathcal{R}_{ces}$ where no two rows are identical. (This simplification merges an attribute and an object whenever they are introduced by the same concept.)
4. Extract from $\mathcal{R}_{ces}$ the elements of $\mathcal{H}(\mathcal{R})$, whose intents are in fact the rows of $\mathcal{R}_{ces}$ and whose simplified labels are the labels of these rows in $\mathcal{R}_{ces}$.
5. Construct the Hasse diagram of $\mathcal{H}(\mathcal{R})$ from these intents.

Note that as objects and attributes play symmetric roles, the algorithm can dually use domination on objects instead of attributes. The choice may result from an unbalanced number of objects with respect to the number of attributes.

5.1 Clarifying $\mathcal{R}$ into $\mathcal{R}_c$

Some objects (resp. attributes) may have the same image set (resp. antecedent set) and will then appear in the same concepts and share the same introducer. To simplify this redundancy, we will then merge identical lines of $\mathcal{R}$ to obtain clarified relation $\mathcal{R}_c$. This can be done in linear $O(|\mathcal{R}|)$ time, as discussed in Subsection 4.2.

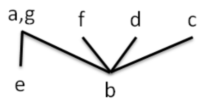
Example. In relation $\mathcal{R}$ of our running example, attributes a and g have the same antecedent set $\{1, 2, 3, 7\}$, objects 2 and 8 have the same image set $\{a, e, f\}$. The corresponding clarified relation $\mathcal{R}_c$ is presented. $\mathcal{R}_c$ and $\mathcal{R}$ have the same lattice and the same Galois sub-hierarchy.

$\mathcal{R}_c$	a,g	b	c	d	e	f
1	×		×	×	×	
2,8	×				×	×
3	×	×	×	×		×
4			×			×
5				×		
6			×	×		
7	×				×	

5.2 Computing $Dom_{\mathcal{A}}$ from $\mathcal{R}_c$

The domination relation on attributes $Dom_{\mathcal{A}}$ is computed using clarified relation $\mathcal{R}_c$ as input. $Dom_{\mathcal{A}}$ has been proven to be a sub-order of the Galois sub-hierarchy where only the elements having an attribute in their simplified label have been preserved [4]. As discussed in Subsection 4.2, this can be done in $O(|Attr|^{\alpha})$ or in $O(|\mathcal{A}|.|\mathcal{R}_c|)$ time.

Example. The domination order $Dom_{\mathcal{A}}$ of $\mathcal{R}_c$ is represented here as a sub-order of $\mathcal{H}(\mathcal{R})$. a and g have been grouped by the clarification process. Then b strictly dominates ag, f, d, and c: $Dom_{\mathcal{A}}(b) = \{ag, f, d, c, b\}$, and e strictly dominates ag: $Dom_{\mathcal{A}}(e) = \{ag, e\}$.



5.3 Constructing relation $\mathcal{R}_{ce}$ and its simplification $\mathcal{R}_{ces}$

We now compute relation $\mathcal{R}_{ce}$, which is the juxtaposition of $\mathcal{R}_c$ with $Dom_{\mathcal{A}}$. The formal definition of $\mathcal{R}_{ce} \subseteq (\mathcal{O} \cup \mathcal{A}) \times \mathcal{A}$ is as follows: $\forall x \in \mathcal{O}$, $\forall y \in \mathcal{A}$, $(x, y) \in \mathcal{R}_{ce}$ iff $(x, y) \in \mathcal{R}_c$, and $\forall y, z \in \mathcal{A}$, $(y, z) \in \mathcal{R}_{ce}$ iff $(y, z) \in Dom_{\mathcal{A}}$.

Now relation $\mathcal{R}_{ce}$ may have identical rows. As the input relation has already been clarified, this can only occur when an object has the same image set (in $\mathcal{R}_c$) as an attribute (in $Dom_{\mathcal{A}}$). We will merge these lines of $\mathcal{R}_{ce}$ to obtain a new relation $\mathcal{R}_{ces}$. We will show in the next section that this last process will associate the rows of $\mathcal{R}_{ces}$ with the elements of $\mathcal{H}(\mathcal{R})$.

This simplification, as the clarification of Step 1, can be obtained in linear time; however, the process now only compares objects with attributes. Note that the initial clarification into $\mathcal{R}_c$ could have been delayed and integrated into this step, but the more redundancies the initial relation contains, the more time the computation of $Dom_{\mathcal{A}}$ will require; a better average time complexity is thus obtained by separating these steps.

Example. $\mathcal{R}_c(3) = Dom_{\mathcal{A}}(b)$, so 3 and b are merged in $\mathcal{R}_{ces}$, as 5 with d, and 7 with e.

$\mathcal{R}_c$	a,g	b	c	d	e	f
1	×		×	×	×	
2,8	×				×	×
3	×	×	×	×		×
4			×			×
5				×		
6			×	×		
7	×				×	

$+$

$Dom_{\mathcal{A}}$	a,g	b	c	d	e	f
a,g	×					
b	×	×	×	×		×
c			×			
d				×		
e	×				×	
f						×

$=$

$\mathcal{R}_c + Dom_{\mathcal{A}} = \mathcal{R}_{ce}$

$\mathcal{R}_{ce}$	a,g	b	c	d	e	f
1	×		×	×	×	
2,8	×				×	×
3	×	×	×	×		×
4			×			×
5				×		
6			×	×		
7	×				×	
a,g	×					
b	×	×	×	×		×
c			×			
d				×		
e	×				×	
f						×

$\rightarrow$

$\mathcal{R}_{ce} \rightarrow \mathcal{R}_{ces}$

$\mathcal{R}_{ces}$	a,g	b	c	d	e	f
1	×		×	×	×	
2,8	×				×	×
4			×			×
6			×	×		
a,g	×					
3,b	×	×	×	×		×
c			×			
5,d				×		
7,e	×				×	
f						×

5.4 Extracting the elements of $\mathcal{H}(\mathcal{R})$ from $\mathcal{R}_{ces}$

We will now prove that the starting set of $\mathcal{R}_{ces}$ yields exactly the elements of $\mathcal{H}(\mathcal{R})$, because of our two-step merging process. Step 1 grouped together separately equivalent objects or equivalent attributes which trivially correspond to objects or attributes having the same introducer. Step 3 grouped together an object and an attribute whenever they have the same introducer, as proved in Theorem 3. Thus the labels of the rows of $\mathcal{R}_{ces}$ are the simplified labels of $\mathcal{H}(\mathcal{R})$, and for each row, its elements yield the intent of the corresponding concept, as proved in Theorem 4. No extra computation is thus needed for this step.

Example. The starting set of $\mathcal{R}_{ces}$ is: $\{\{1\}, \{2{,}8\}, \{4\}, \{6\}, \{a{,}g\}, \{3{,}b\}, \{c\}, \{5{,}d\}, \{7{,}e\}, \{f\}\}$. Its elements correspond exactly to the simplified label of the elements of $\mathcal{H}(\mathcal{R})$ presented in Figure 1. The rows represent the intents of these elements: for example, the complete labeling of the introducer of 2 would be $(\{2{,}8\},\{a{,}g{,}e{,}f\})$.

Theorem 3. *Given a relation $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$, the introducer of $x \in \mathcal{O}$ and the introducer of $y \in \mathcal{A}$ are the same if and only if $\mathcal{R}_{ce}(x) = \mathcal{R}_{ce}(y)$.*

Proof.
– Suppose concept C_{xy} is the introducer of both $x \in \mathcal{O}$ and $y \in \mathcal{A}$. The intent of C_{xy} is by definition $\mathcal{R}(x)$, which includes y. Let z be an attribute of $\mathcal{R}(x) = \mathcal{R}_{ce}(x)$ and C_z its introducer; as $(x, z) \in \mathcal{R}$, $C_{xy} \leq C_z$ and then Extent(C_{xy})⊆Extent(C_z) (Theorem 1); therefore, $(y, z) \in Dom_{\mathcal{A}}$ and so $z \in \mathcal{R}_{ce}(y)$; thus $\mathcal{R}_{ce}(x) \subseteq \mathcal{R}_{ce}(y)$. Let t be an attribute in $\mathcal{R}_{ce}(y)$ and C_t its introducer; $(y, t) \in Dom_{\mathcal{A}}$, which implies $C_t \geq C_{xy}$, and so the extent of C_t contains x, which implies $(x, t) \in \mathcal{R}$, *i.e.* $t \in \mathcal{R}_{ce}(x)$; thus $\mathcal{R}_{ce}(y) \subseteq \mathcal{R}_{ce}(x)$.
– Suppose $\mathcal{R}_{ce}(x) = \mathcal{R}_{ce}(y)$, *i.e.* $\mathcal{R}(x) = Dom_{\mathcal{A}}(y)$. For $z \in Dom_{\mathcal{A}}(y)$, $(x, z) \in$

$\mathcal{R}$, so the introducers of z and x are comparable: $C_z \geq C_x$ (Theorem 1); in particular, $C_y \geq C_x$. By definition of the domination relation, $\mathcal{R}^{-1}(y) \subseteq \mathcal{R}^{-1}(z)$ for all $z \in Dom_{\mathcal{A}}(y) = \mathcal{R}(x)$ and then, by definition of the ordering of $\mathcal{L}(\mathcal{R})$, $C_z \geq C_y$; the extent of C_y is included in the intersection of the extents of these C_z. As we are dealing with elements of a lattice, the extent of C_x is the intersection of the extents of all C_z, $z \in \mathcal{R}(x)$, which means C_x is the infimum (greatest lower bound) of these C_z. Finally, C_y is the minimum concept of a set of concepts of which C_x is the infimum, clearly $C_x = C_y$.
◇

Consequently, the final relation $\mathcal{R}_{ces}$ yields the elements of $\mathcal{H}(\mathcal{R})$:

Theorem 4. *The rows of $\mathcal{R}_{ces}$ are in a one-to-one correspondence with the elements of $\mathcal{H}(\mathcal{R})$. More precisely, each element of the starting set is the simplified label of the corresponding element of $\mathcal{H}(\mathcal{R})$ and its image set is the intent of this concept.*

Proof.
Each object or attribute has an associated introducer. Two objects (resp. attributes) have the same introducer if and only if their image sets (resp. antecedent sets) are equal; the corresponding lines of $\mathcal{R}$ have been merged in $\mathcal{R}_c$. In the other hand, by Theorem 3, an object and an attribute have the same introducer if and only if $\mathcal{R}_{ce}(x) = \mathcal{R}_{ce}(y)$; the corresponding lines of $\mathcal{R}_{ce}$ have been merged in $\mathcal{R}_{ces}$. As a consequence, the starting sets of $\mathcal{R}_{ces}$ are the simplified labels of the elements of $\mathcal{H}(\mathcal{R})$. Their image sets are the corresponding intents: for $x \in \mathcal{O}$, $\mathcal{R}_{ces}(x) = \mathcal{R}(x)$ is the intent of the introducer of x; for $y \in \mathcal{A}$, $\mathcal{R}_{ces}(y) = \{z \in \mathcal{A} \,|\, (y, z) \in Dom_{\mathcal{A}}\}$ which corresponds to the intent of the introducer of y.
◇

Note that the use of $Dom_{\mathcal{A}}$ gives the intent sets of the elements of $\mathcal{H}(\mathcal{R})$. The use of $Dom_{\mathcal{O}}$ instead would have given the extent sets. The use of both $Dom_{\mathcal{A}}$ and $Dom_{\mathcal{O}}$, as proposed in [4], is less efficient for computing the elements of $\mathcal{H}(\mathcal{R})$.

5.5 Constructing the Hasse diagram of $\mathcal{H}(\mathcal{R})$

Now all we have left to do is construct the Hasse diagram of $\mathcal{H}(\mathcal{R})$ by constructing the ordering by inclusion on the intents. This can be done in $O((|\mathcal{O}| + |\mathcal{A}|)^{\alpha})$ time by removing all transitivity edges, as discussed in Subsection 4.2.

6 The algorithm

```
Algorithm HERMES
Input: binary relation $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$.
Output: $\mathcal{H}(\mathcal{R})$
    Compute clarified relation $\mathcal{R}_c$;      //merge all identical lines
    Compute relation $Dom_{\mathcal{A}}$;             //determine column inclusions in $\mathcal{R}_c$
    $\mathcal{R}_{ce}$=$\mathcal{R}_c$+$Dom_{\mathcal{A}}$;            //simple juxtaposition
    Simplify $\mathcal{R}_{ce}$ into $\mathcal{R}_{ces}$;          //merge all identical rows
    Extract the elements of $\mathcal{H}(\mathcal{R})$;   //select the starting set of $\mathcal{R}_{ces}$
    Order the elements of $\mathcal{H}(\mathcal{R})$;     //by inclusion on their image sets.
```

The complexity of the algorithm is bounded by Steps 2 and 5 with a time in $O((|\mathcal{O}|+|\mathcal{A}|).|\mathcal{R}|)$ or $O((|\mathcal{O}|+|\mathcal{A}|)^{\alpha})$, depending on the chosen implementation.

7 Specialized input: chordal-bipartite relations

A special class of relations should be mentioned in this context: relations which correspond to 'chordal-bipartite graphs', which are bipartite graphs containing no chordless cycle of length six or more. This is a superclass of the relations which have a planar lattice, but the lattice of chordal-bipartite relations remains of polynomial size [8].

Relations whose corresponding bipartite graph is chordal-bipartite can be re-ordered so that their matrix becomes 'Γ-free'. A Γ in a matrix is a sub-matrix on 4 elements, with a unique zero in the right-hand lower corner (*i.e.* in matrix M, there is a pair h,i of rows, $h < i$, and a pair j,k of columns, $j < k$, such that M(h,j)=M(h,k)=M(i,j)=1 and M(i,k)=0).

This Γ-free form is obtained by computing a 'Double Lexical Ordering' (DLO) [17]. A DLO is an ordering of the matrix such that the binary 'words' read from bottom to top for columns are in increasing lexical order, and likewise for rows, the binary words read from right to left are in increasing order from bottom to top. In the example below, column a has word 1001**0** which is smaller than the word of b, which is 0000**1**, and likewise the word of object 2, 00**0***01* is smaller than the word of object 3, 00**1***01*.

Any matrix can be re-ordered to be DLO, and this is re-ordering can be done in time $O(min\{m\, logn,\, n^2\})$ [19, 23]. The DLO matrix is Γ-free if and only if the relation is chordal bipartite [17].

When a relation is in such a DLO and Γ-free form, it is easy to compute $Dom_{\mathcal{A}}$: take each attribute from left to right; for each attribute y, let x be the first object (from top to bottom) in the column of y (*i.e.* the first x such that $(x, y) \in \mathcal{R}$); then y dominates exactly the attributes z which are to its right and that are on row x (*i.e.* $(x, z) \in \mathcal{R}$).

This a consequence of the DLO and Γ-free form: in a DLO matrix, a given column can not be included in any column to its left; and in a Γ-free matrix, if w is the first row with a one in column y, for any column z at the right of y

which has a one in the row of w, if column y is not included in column z, as the rows of y above w all have zeros, this might only be because of a row x after w with a one in column y and a zero in column z, *i.e.* because of a Γ in the matrix formed by rows w and x, and columns y and z.

Example. The following matrix is ordered in a double lexical fashion and is Γ-free. Attribute a is processed first; its first one is on row 1, so a dominates all the attributes to its right which has a one on row 1: a dominates d. Attribute b is processed next; its first one is on row 5, which has ones at the right of b for c, d and e, b dominates c, d and e. Attribute c: highest one in row 3, c dominates e. Attribute d: highest one in row 1, no one at the right, no domination. Attribute e is last and therefore can dominate no other attribute.

$\mathcal{R}$	a	b	c	d	e
1	1			1	
2					1
3			1		1
4	1			1	1
5		1	1	1	1

When relation $\mathcal{R}$ is chordal-bipartite, $\mathcal{R}_{ces}$ can then be constructed in $O(n^2)$. We conjecture that the Hasse diagram can be extracted at no extra cost.

8 Conclusion

We have presented a new, simple, and more efficient algorithm, Hermes, for building the Galois sub-hierarchy of a relation. It would be interesting to compare its running time in practice to that of the other known algorithms; we conjecture that Hermes will run faster in most cases.

Algorithm Hermes could be remodeled into an incremental algorithm, which may prove interesting for on-line applications such as updating hierarchies in object-oriented languages.

Acknowledgement

The authors thank Lhouari Nourine for fruitful discussions on the construction of Galois sub-hierarchies.

References

1. Arévalo G., Berry A., Huchard M., Perrot G., and Sigayret A.: *Comparison of performances of Galois sub-hierarchy-building algorithms.* Proc. of ICFCA'07, LNCS 4390, p. 166-180. 2007.
2. Berry A., Bordat J.-P., and Sigayret A.: *A local approach to concept generation.* Ann. Math. Artificial Intelligence 49, p.117-136. 2007.
3. Berry A., Huchard M., McConnell R.M., Spinrad J.P.: *Efficiently Computing a Linear Extension of the Sub-hierarchy of a Concept Lattice.* Proc. of ICFCA'05. 2005

4. Berry A., and Sigayret A.: *Maintaining Class Membership Information.* Workshop MASPEGHI, proc. OOIS'02 (Conference on Object-Oriented Information Systems). 2002.
5. Bordat J-P.: *Calcul pratique du treillis de Galois d'une correspondance.* Mathématiques, Informatique et Sciences Humaines, 96, p. 31-47. 1986.
6. Coppersmith D., and Winograd S.: *Matrix multiplication via arithmetic progressions.* Proc. 9th Annual ACM Symposium on Theory of Computing, p.1-6. 1987.
7. Dicky H., Dony C., Huchard M., and Libourel T.: *Ares, adding a class and restructuring inheritance hierarchies.* Proc. BDA'95, p. 25-42. 1995.
8. Eschen E., Pinet N., Sigayret A.: *Consecutive-ones: handling lattice planarity efficiently.* Proc. CLA'07. 2007.
9. Ganter B., Wille R.: *Formal Concept Analysis: Mathematical Foundations.* Springer. 1999.
10. Godin R., Mili H.: *Building and Maintaining Analysis-Level Class Hierarchies Using Galois Lattices.* Proc. OOPSLA'93, p.394-410. 1993.
11. Godin R., Chau T-T.: *Comparaison d'algorithmes de construction de hiérarchies de classes*, Journal L'OBJET, 5:3-4. 1999.
12. Hitzer P.: *Default reasoning over Domains and Concepts Hierarchies.* Proc. KI'04, LNCS 3238, p.351-365. 2004.
13. Hsu W.-L., and Ma T.-H.: *Substitution decomposition on chordal graphs and its applications.* SIAM Journal on Computing, 28, p.1004-1020. 1999.
14. Huchard M., Dicky H., and Leblanc H.: *Galois Lattice as a Framework to specify Algorithms Building Class Hierarchies.* Theoretical Informatics and Applications, 34, p. 521-548. 2000.
15. Huchard M., Leblanc H.: *Computing Interfaces in Java.* ASE, p.317-320. 2000.
16. Leblanc H. *Sous-hiérarchies de Galois: un modèle pour la construction et l'évolution des hiérarchies d'objets* (in French). PHD thesis, Université Montpellier 2. 2000.
17. Lubiw A.: *Doubly lexical orderings of matrices.* SIAM Journal on Computing, 16(5), p.854-879. 1987.
18. Osswald R., and Petersen W.: *Introduction of Classification from Linguistic Data.* Proc. ECAI'02, Workshop FCAKDD, p.75-84. 2002.
19. Paige R., Tarjan R.E.: *Three partition refinement algorithms.* SIAM J. Comput. 16(6), p.973-989. 1987.
20. Petersen W.: *A Set-Theoretical Approach for the Induction of Inheritance Hierarchies.* Electronic Notes in Theoretical in Computer Science, 51. 2001.
21. Ryssel U., Ploennigs J., Kabitzsch K.: *Extraction of feature models from formal contexts.* SPLC Workshops (15th Int. Software Product Line Conference). 2011.
22. Sigayret A.: *Data mining: une approche par les graphes* (in French). PHD thesis, Université Blaise Pascal (Clermont-Ferrand, France). 2002.
23. Spinrad J.-P.: *Doubly lexical ordering of dense 0-1 matrices.* Information Processing Letters, 45(5), p.229–235. 1993.
24. http://conexp.sourceforge.net/download.html, release 1.3. © S.A. Yevtushenko & al. 2000-2006.

Mining Concepts from Incomplete Datasets Utilizing Matrix Factorization

Lenka Pisková[1], Štefan Pero[1], Tomáš Horváth[2], Stanislav Krajči[1]

[1] ICS, Pavol Jozef Šafárik University, Košice, Slovakia
{Lenka.Piskova, Stefan.Pero}@student.upjs.sk
Stanislav.Krajci@upjs.sk
[2] ISMLL, University of Hildesheim, Germany
horvath@ismll.de

Abstract. Formal Concept Analysis aims at finding clusters (concepts) with given properties in data. Most techniques of concept analysis require a dense matrix with no missing values on the input. However, real data are often incomplete or inaccurate due to the noise or other unforeseen reasons. This paper focuses on using matrix factorization methods to complete the missing values in the input data such that it can be used with arbitrary concept analysis technique. The used matrix factorization model approximates the sparse object-item data matrix by a product of two dense factor matrices, thus, mapping objects and items to a common latent space. The mentioned object-factor and item-factor matrices are obtained by a simple stochastic gradient optimization method. We also investigate how the amount of missing values influences the output of the concept analysis. Two measures, well-known in the information retrieval community, have been used for the evaluation of the proposed framework. Real datasets from the UCI Machine Learning Repository were used in our experiments.

Keywords: Formal Concept Analysis, matrix factorization, missing values completion, clustering

1 Introduction

Formal Concept Analysis (FCA) deals with data in the form of a table which rows represent objects and columns represent attributes of objects. A table entry corresponding to an object x and an attribute y indicates whether or not the object x has the attribute y. The clusters on the output of FCA are called concepts, each of which consists of a set of formal objects and a set of formal attributes, called the extent and the intent, respectively, of the given concept. The set of all concepts ordered by $\leq$ forms a concept lattice [1]. The data table required on the input of FCA has to contain all information, i.e. should not contain missing values.

However, real datasets are often incomplete or inaccurate due to the damage or inconsistency in the data collection process. Incomplete context was first introduced by Burmeister and Holzer [2] and applied to attribute implications and attribute exploration. The analysis of incomplete data in the field of FCA can be accomplished in one of the following two ways: The first one is to adapt concept techniques to cope with missing values. Some research was conducted, recently, to generate a concept lattice of an incomplete context [7]. The approach presented in [8] is based on many-valued contexts and conceptual scaling. The second way of handling incomplete data in FCA is to complete the missing values in a pre-processing step and use a concept analysis on the pre-processed data.

The contribution of this paper follows the latter case described above, namely, to predict the missing values using a matrix factorization method (MF) in the pre-processing step such that FCA can be used on the completed data. MF, one of the useful techniques in data mining, allows the decomposition of a sparse[1] matrix into two dense matrices such that the product of these two dense matrices results in a dense matrix which is an approximation[2] of the original, sparse matrix. We use stochastic gradient descent matrix factorization method (SGD MF) in this paper, which is fast and effective. It is important to note that we need to adjust the predicted values since these should be from the input values. The main reason of it is that the resulting dense matrix is just an approximation of the original one with specific values, e.g. 0 and 1 or 1, 2, 3, 4, 5, etc. The resulting dense matrix is then scaled [1] and feed to FCA.

We also address the problem of the robustness of formal concepts, i.e. the ability to remain unaffected by small variations in an input table. As far as we know, there is no related work investigating the issue that how the concepts computed from a full table differs from those computed from a pre-processed table, i.e. what happens when we remove some values from the original table, complete them with a pre-processing method and compute the concepts on the pre-processed table? We think that this question is quite important for a real-life application of FCA.

2 Related Work

One direction of estimating missing values in an input matrix is by the use of association rule mining techniques consisting of two approaches: The first one discards instances which contain missing data and generate association rules from the remaining complete data. However, excluding instances with missing data can bias the result. The second approach takes into account the presence of missing values in the rule mining process. Note that two or more association rules may result to different predicted values, thus the so called conflict problem have to be tackled here. The large number of association rules and the effort to reduce the conflict problem have led to the usage of generic bases of association rules. We refer to [3], [4] for details and further references. The percentage of correctly completed missing values is affected by the number of missing values.

[1] We will call a matrix with missing values sparse.

[2] The difference of the values in the non-empty cells of the original matrix and the predicted values for these cells is minimal.

A "hidden" problem is the percentage of missing values that these approaches permits to complete.

[5] and [6] show how to use extracted knowledge represented by formal concepts for completing a relation or at least augmenting it in the case when we do not get each missing value. They generate concepts from the sparse matrix such that they remove rows (objects) containing missing values. However, these approaches have two drawbacks, namely, that the large number of concepts (also for a small relation) makes more difficult to predict missing values, and, more missing values leads to more biased completion.

Combinations of MF and FCA are introduced in [9], [10] and [11], where singular value decomposition, semi-discrete decomposition and non-negative matrix factorization are used for reducing the number of formal concepts. These works, however, use matrix factorization for a different purpose (reduction of the number of concepts) and consider full input matrix.

A novel approach to combine matrix decomposition and factor analysis is presented in [12], [13]. There are two main differences from ordinary factor analysis. First, a table entry represents a degree to which an object has an attribute (table entries are from a bounded scale L). Second, the matrix composition operator is a so-called t-norm-based product. The aim of this approach is not the completion of missing values but finding a set of factors that correspond to formal concepts.

3 Preliminaries

We will just briefly describe FCA, and focus instead on a more detailed description of the used MF technique in this section since we think it could be more interesting to the FCA community.

3.1 Formal Concept Analysis

A formal context is a triple (X, Y, I) consisting of two non-empty sets X and Y and a binary relation I between them. The elements of X are called the objects and the elements of Y are called the attributes. $(x, y) \in I$ means that the object x has the attribute y.

For a set $A \subseteq X$ and a set $B \subseteq Y$, define $A' = \{y \in Y : (\forall x \in A)(x, y) \in I\}$ and $B' = \{x \in X : (\forall y \in B)(x, y) \in I\}$. A' is the set of attributes common to the objects in A and B' is the set of object which have all attributes in B.

A formal concept of the context (X, Y, I) is a pair (A, B) of a set $A \subseteq X$ of objects and a set $B \subseteq Y$ of attributes such that $A' = B$ and $B' = A$. A and B are called the extent and the intent of the concept (A, B), respectively. Denote the set of all concepts in (X, Y, I) by $\mathcal{B}(X, Y, I)$.

Introduce a relation $\leq$ on $\mathcal{B}(X, Y, I)$ by $(A_1, B_1) \leq (A_2, B_2)) \Leftrightarrow A_1 \subseteq A_2 \Leftrightarrow B_2 \subseteq B_1$, (A_1, B_1) is called the subconcept of (A_2, B_2) and (A_2, B_2) is called the superconcept of (A_1, B_1). The set of concepts $\mathcal{B}(X, Y, I)$ ordered by $\leq$ constitutes the concept lattice (or Galois lattice) of the context (X, Y, I). The so-called main theorem of concept lattices characterizes the structure of concept lattices. For further details we refer to [1].

3.2 Stochastic Gradient Descent Matrix Factorization

Stochastic gradient descent matrix factorization (SGD MF) is one of the most popular factorization techniques [14] because of its scalability and good accuracy.

The goal of this approach is to approximate a sparse matrix $X \in \mathbb{R}^{|\mathcal{O}|\times|\mathcal{I}|}$ with a product of two (much smaller) matrices $W \in \mathbb{R}^{|\mathcal{O}|\times K}$ and $H \in \mathbb{R}^{|\mathcal{I}|\times K}$ such that

$$X \approx \hat{X} = WH^T, \tag{1}$$

where $|\mathcal{O}|, |\mathcal{I}|$ are the number of objects and items, respectively, and K is number of (latent) factors. The ith row of the matrix W is a vector containing K latent factors describing the object i, and the jth row in matrix H is a vector containing K latent factors describing the item j [15].

The estimate $\hat{x}_{ij}$ of a missing value at the row i and column j of the sparse matrix X is computed as

$$\hat{x}_{ij} = (WH^T)_{ij} = \sum_{k=1}^{K} w_{ik}h_{jk} \tag{2}$$

where w_{ik} (h_{jk}) is the value of W (H) at the ith (jth) row and kth column.

We are interested in such $\hat{X}$ which estimates the missing values of X well. Since we do not know the missing values, we use the following trick, well-known in the machine learning community [18]: We split X into two complementary parts, called the train set X^{train} and the test set X^{test}. The model $\hat{X}$ is then approximated from X^{train} and it's quality is assessed by the root mean squared error (RMSE) loss computed on X^{test}, defined as

$$RMSE = \sqrt{\frac{\sum\limits_{x_{ij}\in X^{test}} (x_{ij} - \hat{x}_{ij})^2}{|X^{test}|}}, \tag{3}$$

where x_{ij} are the values of the non-empty cells of X belonging to X^{test}. In this way, we simulate the case when we have a sparse matrix (X^{train}) which missing values (X^{test}) are known, however.

Our goal is to find those parameters W and H of the model $\hat{X}$ for which the RMSE is minimal. However, we have to keep in mind that X^{test} is “hidden”, i.e. it represents the “unknown“ missing values, thus, we have to use[3] the data we know, i.e. the train set X^{train}. Since $|X^{train}|$ is constant, minimizing RMSE (3) on X^{train} is equivalent to minimizing the sum of errors $\sum_{x_{ij}\in X^{train}}(x_{ij} - \hat{x}_{ij})^2$. Since RMSE is a (loss) function, we use a stochastic gradient descent (SGD)

[3] Here we also expect that X^{test} has similar statistical characteristics as X^{train}.

optimization method [16] for searching those parameters W and H of RMSE such that the following objective function is minimal:

$$\sum_{x_{ij} \in X^{train}} (x_{ij} - \hat{x}_{ij})^2 + \lambda(\|W\|^2 + \|H\|^2) \tag{4}$$

where $\hat{x}_{ij}$ is defined as in (2) and λ is a regularization term to prevent the so-called overfitting (i.e. when a model estimates very well the training data but poorly the test data [18]). λ controls the magnitudes of the factor vectors such that W and H would give a good approximation of the whole original input matrix X (containing both X^{train} and X^{test}). $\|W\|^2$ means square of the one vector in matrix W.

In the training phase we first initialize two matrices W and H with some random values (for example from the normal distribution $N(0, \sigma^2)$ with mean 0 and standard deviation 0.01) and compute the estimation error

$$err = \sum_{x_{ij} \in X^{train}} e_{ij}^2 \tag{5}$$

where

$$e_{ij}^2 = (x_{ij} - \hat{x}_{ij})^2 = (x_{ij} - \sum_{k=1}^{K} w_{ik} h_{jk})^2 \tag{6}$$

We minimize this error by updating the values of W and H in the following way [16]: we randomly choose a value x_{ij} from X^{train} and compute the gradient of the objective function (6) in this value w.r.t. the parameters[4] w_i and h_j, i.e.

$$\frac{\delta}{\delta_{w_{ik}}} e_{ij}^2 = -2e_{ij} h_{jk} = -2(x_{ij} - \hat{x}_{ij})^2 h_{jk} \tag{7}$$

$$\frac{\delta}{\delta_{h_{jk}}} e_{ij}^2 = -2e_{ij} w_{ik} = -2(x_{ij} - \hat{x}_{ij})^2 w_{ik} \tag{8}$$

In the next step we use the computed gradient to update the values of w_{ik} and h_{jk}:

$$w_{ik}^{(new)} = w_{ik} - \beta \frac{\delta}{\delta_{w_{ik}}} e_{ij}^2 = w_{ik} + 2\beta e_{ij} h_{jk} \tag{9}$$

$$h_{jk}^{(new)} = h_{jk} - \beta \frac{\delta}{\delta_{h_{jk}}} e_{ij}^2 = h_{jk} + 2\beta e_{ij} w_{ik} \tag{10}$$

where β is the learning rate controlling the step sizes.

We update W and H for each value of X^{train} in one iteration. The number of iterations, i.e. the number of passes over the full X^{train} is a hyper-parameter of the factorization technique as well as the regularization term λ, the learn rate β and the number of factors K.

[4] Only the vectors w_i and h_j contribute to e_{ij}^2, thus we treat the other vectors (rows of W and H) as constants (which derivatives are zero).

As we see, the quality of the model $\hat{X}$ depends on the hyper-parameters used in the learning phase. Also, it can happen that the split of the data into the train and the test part was done in a way that these two parts have no similar characteristics, e.g. one part contains large values while the other the low ones. To prevent this phenomenon to happen we usually use n-fold cross validation [18] in which we split X into n equal parts $X_1, \ldots, X_n$ and for each n and hyper-parameter combination $(K, \lambda, \beta, iterations)$ do the following: train a model $\hat{X}^n$ with the given hyper-parameter combination on $\cup_{i \neq n} X_i$ and compute its RMSE on X_n. The resulting RMSE of $\hat{X}$ with the hyper-parameters $(K, \lambda, \beta, iterations)$ is then the average RMSE over all $\hat{X}^n$. After trying some hyper-parameter combinations with n-fold cross validation, we choose the best combination $(K, \lambda, \beta, iterations)_{best}$ which has the smallest average RMSE over all folds. Finally, we train the model $\hat{X}$ again using the whole input matrix X.

It can happen, that the input matrix X contains an empty line or column with no values. For those objects (rows) or items (columns) we will not update the parameters W and H (it follows from the algorithm of SGD MF). In such cases, estimated missing values could be the global average of non-missing values. More advanced and sophisticated methods that can improve the estimation in such cases are listed in [17].

4 Illustrative example

To illustrate our approach we present a small toy example. Our purpose is to cluster students into groups with respect to their success in solving the tasks. The key problem is that not every student have solved all of the tasks. We will proceed as follows. We predict the success of students in tasks that they have not solved yet (empty values in the input table). After the process of conceptual scaling is completed, we find formal concepts.

The table in the left side of the figure 1 contains five students who have performed several tasks. The total marks that can be scored are 9; 3 marks for each task. Table values represent students' marks for tasks. The matrix represents a relation between objects and items (which are students and tasks in our case).

Figure 1 shows an example of how we can factorize the students and tasks. After the training phase with $K = 2$ latent factors (F1 and F2), we get the student-factor matrix W and the factor-task matrix H. Suppose we would like to predict Tom's (3rd row) performance for the task 2 (2nd column). The prediction is performed using equation 2

$$\hat{x}_{32} = \sum_{k=1}^{K} w_{3k} h_{2k} = 1.80 * 1.20 + 0.13 * 1.08 = 2.30.$$

Since the predicted value should be included in the input matrix, we round this value to one of the values 1, 2 and 3. We have predicted Tom will achieve average results in task 2. Similarly, we can predict the performance of other

X	T1	**T2**	T3
Katie	3	3	
Dean	2	2	1
Tom	2		2
Sam		2	
Adam	2	3	2

W	F1	F2
Katie	1.18	1.52
Dean	1.24	0.31
Tom	1.80	0.13
Sam	0.91	1.57
Adam	1.68	0.49

H	T1	T2	T3
F1	0.98	1.20	0.85
F2	1.35	1.08	0.54

Fig. 1. Factorization of the input matrix.

students in tasks which they have not done yet. The full matrix is depicted in the left side of the figure 2.

Now, we would like to cluster students according to their results in solving the tasks. The table in the left side of the figure 2 represents many-valued context. Since values are ordered and each value implies the weaker one, we use the ordinal scale in the right side of the figure 2. Performing a task at the highest level implies performing the task at the average level, too. The table in the figure 3 is the result of the conceptual scaling process. For details on many-valued contexts and conceptual scaling we refer to [1].

	T1	T2	T3
Katie	3	3	2
Dean	2	2	1
Tom	2	2	2
Sam	3	2	2
Adam	2	3	2

	3	2	1
3	×	×	×
2		×	×
1			×

Fig. 2. The many-valued context and the ordinal scale S for all attributes of the many-valued context.

The obtained concept lattice contains 6 concepts (clusters) and is depicted in the figure 3. A labeling is simplified by putting down each object and attribute only once. The extent of each concept is formed by collecting all objects which can be reached by descending line paths from the concept. The intent of each concept consists of all attributes located along ascending line paths.

We interpret some of the clusters, here: Each student solved the task 1 and task 2 at least at average level (the top element of the lattice). All students except for Dean achieved at least average results in solving all tasks. Nobody performed good results in every task (the bottom element of the lattice).

5 Experiments

Experiments were conducted on computing cluster with 7 computing nodes, each of which has 16 cores, running Red Hat Linux.

	T1			T2			T3		
	1	2	3	1	2	3	1	2	3
Katie	×	×	×	×	×	×	×	×	
Dean	×	×		×	×		×		
Tom	×	×		×	×		×	×	
Sam	×	×	×	×	×		×	×	
Adam	×	×		×	×	×	×	×	

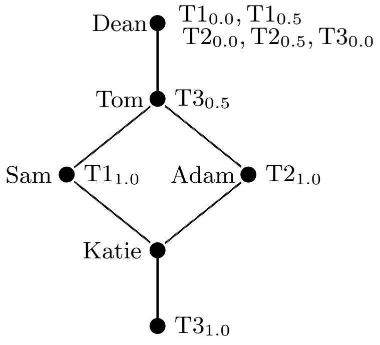


Fig. 3. The one-valued context as the derived context of the many-valued context in figure 2 and the corresponding concept lattice.

5.1 Data

For these experiments, we consider a complete database to act as a reference database, and we randomly introduce missing values with the following different rates : 10%, 20% and 30%. Benchmark datasets used for this experiments are from the UCI Machine Learning Repository[5]. Characteristics of these datasets are depicted in the table 1.

Table 1. Datasets characteristics

Dataset	Objects	Attributes	Attribute types	FCA attributes
Wine	178	13	Integer, Real	68
SPECT Heart	267	22	Categorical	46
Tic-tac-toe	958	9	Categorical	29

The data conversion into FCA format was done as follows. Categorical (many-valued) attributes were converted by creating a formal context attribute for each of the values. Attributes with integer or real types were scaled to create context attributes with ranges of values.

5.2 Matrix factorization model settings

We implemented matrix factorization model in Java described in the section 3.2 on our own. We have used 3-fold cross-validation for testing the proposed model: In each of the 3 iterations, one fold was used for testing. The remaining two folds were used for tuning hyper-parameters of the model. Hyper-parameters (number

[5] http://www.ics.uci.edu/mlearn/MLRepository.html.

of factors, number of iterations, learn rate and regularization term) were tuned using grid search [19] (a systematic search over different combinations). The best hyper-parameter combinations for our model and the root mean squared error (RMSE) was measured on the test fold for each dataset. The best hyper-parameters are reported in the table 2.

Table 2. The best hyper-parameters found by grid search

Dataset	Missing values (%)	Factors	Iterations	Learn rate	Reg. term
Wine	10	12	30	0.055	0.1
	20	12	40	0.055	0.05
	30	12	50	0.055	0.05
SPECT Heart	10	22	50	0.055	0.05
	20	42	40	0.055	0.05
	30	72	40	0.055	0.05
Tic-tac-toe	10	42	70	0.005	0.05
	20	12	60	0.005	0.05
	30	32	90	0.005	0.1

5.3 Results

Concept lattices from complete data sets were computed, the number of concepts is shown in the table 3.

Table 3. Datasets and the number of concepts computed from the full matrix.

	number of concepts
Wine	24423
SPECT Heart	2135549
Tic-tac-toe	59505

Table 4 presents the comparison of the RMSE measured over 3 folds and average of success rates of the prediction for all datasets. Using matrix factorization model we correctly predicted from 30% to 80% missing values. If we did not predict a missing value correctly then the predicted value was close to the original value.

In order to evaluate methods for completing missing values to mine formal concepts, one have to compare concepts generated from the original data table (initial context) with concepts generated from the data table containing estimated values (predicted context).

Table 4. RMSE and the average of success rates of the prediction

Dataset	Missing values (%)	RMSE	Average of the success rates (%)
Wine	10	0.6944168	53.18
	20	0.7345221	66,46
	30	0.6972556	48,84
SPECT Heart	10	0.37255815	78,61
	20	0.3785878	78,09
	30	0.38591003	77,37
Tic-tac-toe	10	0.88052654	31,72
	20	0.9031108	31,56
	30	0.9076978	33,71

We propose to use two measures for evaluating the performance of the proposed approach for the missing values completion with respect to mined concepts. Both are well-known in the information retrieval (IR) community. The first one indicates how many concepts of the initial context occur in the set of concepts of the estimated context. The second one determines how many concepts of the estimated context are in the set of concepts of the initial context.

Let (X, Y, I) be the initial formal context and (X, Z, J) be the estimated formal context. Let O be the set of formal concepts of the initial context and E be the set of formal concepts of the estimated one ($O = \mathcal{B}(X, Y, I)$ and $E = \mathcal{B}(X, Z, J)$). We propose to use the following two measures:

$$precision = \frac{|O \cap E|}{|E|} \text{ and } recall = \frac{|O \cap E|}{|O|}.$$

Precision is the fraction of concepts of the estimated context that are in the set of concepts of the initial context. Recall is the fraction of concepts of the initial context that are in the set of concepts of the estimated context. The higher the precision, the more concepts of the estimated context are in the set of concepts of the initial context. The higher the recall, the more concepts of the initial context are in the set of concepts of the estimated context.

Two formal concepts are equal when any formal object in the extent of one is in the extent of the other and vice versa, the same for formal attributes. In this case, *precision* and *recall* are computed. From the extent-ional point of view: two concepts are equal when they have the same extent. Analogously, from the point of view of intent, two concepts are equal when they have the same intent. In these cases, we calculate $precision_e$, $recall_e$ and $precision_i$, $recall_i$, respectively.

The results of our experiments are shown in the table 4 and 5.

It is noteworthy that there is a huge difference between $precision_e$ and $precision_i$. We assume that the gap between $precision_e$ and $precision_i$ is caused by the fact that the number of objects is greater than the number of attributes.

Moreover, the precision and recall are quite small. The reason is how we have measured the equality of the sets: the sets are not equal if they differ even only in one element. Nevertheless, the proposed measures are useful for experimental

Table 5. Precision and recall.

Dataset	Missing values (%)	*prec*	$prec_e$	$prec_i$	*rec*	rec_e	rec_i
Wine	10	0,0005	0,0277	0,2603	0,0005	0,0311	0,2920
	20	0,0004	0,0246	0,2603	0,0004	0,0272	0,2883
	30	0,0002	0,0194	0,2187	0,0002	0,0198	0,2238
SPECT Heart	10	6×10^{-5}	0,0107	0,3775	4×10^{-6}	0,0007	0,0257
	20	4×10^{-5}	0,0063	0,3667	3×10^{-6}	0,0005	0,0279
	30	2×10^{-5}	0,0039	0,3346	2×10^{-6}	0,0003	0,0267
Tic-tac-toe	10	0,0011	0,0239	0,3969	0,0002	0,0032	0,0524
	20	0,0011	0,0132	0,3899	0,0001	0,0009	0,0258
	30	0,0008	0,0139	0,3477	0,0001	0,0011	0,0282

comparisons of various techniques used in the pre-processing step. We will focus on finding other measures. We suppose that if we modify measures to use the similarity (of extents, intents and concepts) better results could be achieved.

6 Conclusion and future research directions

In this paper we have introduced framework for mining concepts from incomplete datasets. The proposed framework uses stochastic gradient descent matrix factorization (SGD MF) in the pre-processing step and after completing the missing values a formal concept analysis (FCA) is deployed. Using SGD MF we are able to quite precisely predict values from a sparse data matrix without having any background knowledge about the data. Two measures (precision and recall) have been used for evaluating the presented framework.

The experiments on three datasets (with 10, 20 and 30 percent of missing values) showed that the percentage of missing values does not influence the prediction rate, and, that there is a large difference in the precision and recall measures when computed on the extents and the intents of the resulting concepts, respectively.

Our further research will focus on modifying the (precision and recall) measures with using the similarity of extents, intents and concepts instead of their equality. The generation of all concepts from the huge incidence matrix is time consuming and the number of concepts is very large. Using our matrix factorization model we transform input matrix into more smaller latent factor matrices W and H. These matrices describe objects and items according to latent factors. We want to explore these matrices and to use them for reducing the number of concepts.

Even if there are some remaining issues to investigate, experimental results show that the proposed approach is promising and worth of further research.

Acknowledgements: This work was supported by the grants VEGA 1/0832/12 and VVGS-PF-2012-22 at the Pavol Jozef Šafárik University in Košice, Slovakia.

References

1. B. Ganter, R. Wille: Formal concept analysis: Mathematical foundations. Springer, 1999, ISBN 3-540-62771-5.
2. P. Burmeister, R. Holzer: Treating Incomplete Knowledge in Formal Concept Analysis. In: B. Ganter, G. Stumme, R. Wille (Eds.): Formal Concept Analysis: State of the Art, LNAI 3626, Springer, Heidelberg, 2005.
3. L. Ben Othman, S. Ben Yahia: Yet Another Approach for Completing Missing Values. CLA 2006, pages 155-169.
4. L. Ben Othman, S. Ben Yahia: $GBAR_{MVC}$: Generic Basis of Association Rules Based Approach for Missing Values Completion. The International Journal of Computing and Information Sciences (IJCIS), 2008.
5. S. Ben Yahia, Kh. Arour, A. Jaoua: Completing Missing Values Using Discovered Formal Concepts. DEXA 2000, pages 741-751.
6. S. Ben Yahia, Kh. Arour.: Mining fuzzy formal concepts for completing missing values. In Proceedings of the Intl. Symposium on Innovation in Information and Communication Technology, Amman, Jordan, 2002.
7. M. Krupka, J. Lastovicka: Concept lattices of incomplete data. In: F. Domenach, D.I. Ignatov, and J. Poelmans (Eds.): ICFCA 2012, LNAI 7278. Springer, Heidelberg, pages 180-194, 2012.
8. J. Liu, X. Yao: Formal Concept Analysis of Incomplete Information System. In: FSKDIEEE (2010), pages 2016-2020.
9. P. Gajdoš, P. Moravec, V. Snášel: Concept Lattice Generation by Singular Value Decomposition, In: V. Snášel, R. Bělohlávek (Eds.): CLA 2004, pages 102-110.
10. V. Snášel, P. Gajdoš, H. Abdulla, M. Polovinčák: Using Matrix Decompositions in Formal Concept Analysis. ISIM 2007, pages 121-128.
11. V. Snášel, H. Abdulla, M. Polovinčák: Using Nonnegative Matrix Factorization and Concept Lattice Reduction to visualizing data. SYRCoDIS, 2007.
12. R. Bělohlávek, V. Vychodil: Factor Analysis of incidence data via Novel Decomposition of Matrices, In: S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pages 83-97, 2009.
13. R. Bělohlávek, V. Vychodil: Discovery of Optimal Factors in binary data via novel method of matrix decomposition. Journal of Computer and System Sciences 76(1)(2010), pages 3-20.
14. Y. Koren, R. Bell and C. Volinsky. Matrix Factorization Techniques for Recommender Systems. COMPUTER (Vol. 42, 8): IEEE Computer Society Press, Los Alamitos, USA, 2009.
15. G. Takács, I. Pilászy, B. Németh. Investigation of Various Matrix Factorization Methods for Large Recommender Systems. KDD Netflix workshop, 2008.
16. L. Bottou. Stochastic learning. In Bousquet, O. and von Luxburg, U., editors, Advanced Lectures on Machine Learning, Lecture Notes in Artificial Intelligence, LNAI 3176, pages 146-168. Springer Verlag, Berlin.
17. Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, L. Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. In Proceedings of the 10th IEEE International Conference on Data Mining (ICDM 2010). IEEE Computer Society, 2010.
18. T. Hastie, R. Tibshirani, J. Friedman: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Second Edition, Springer Series in Statistics, 2009.
19. J. Bergstra, Y. Bengio: Random Search for Hyper-Parameter Optimization. J. Machine Learning Research 13, pages 281-305, 2012.

Learning Model Transformations from Examples using FCA: One for All or All for One? ⋆

Hajer Saada[1], Xavier Dolques[2], Marianne Huchard[1], Clémentine Nebut[1], and Houari Sahraoui[3]

[1] LIRMM, Université de Montpellier 2 et CNRS, Montpellier, France, first.last@lirmm.fr
[2] INRIA, Centre Inria Rennes - Bretagne Atlantique, Campus universitaire de Beaulieu, 35042 Rennes, France, xavier.dolques@inria.fr
[3] DIRO, Université de Montréal, Canada, sahraouh@iro.umontreal.ca

Abstract. In Model-Driven Engineering (MDE), model transformations are basic and primordial entities. An efficient way to assist the definition of these transformations consists in completely or partially learning them. MTBE (Model Transformation By-Example) is an approach that aims at learning a model transformation from a set of examples, i.e. pairs of transformation source and target models. To implement this approach, we use Formal Concept Analysis as a learning mechanism in order to extract executable rules. In this paper, we investigate two learning strategies. In the first strategy, transformation rules are learned independently from each example. Then we gather these rules into a single set of rules. In the second strategy, we learn the set of rules from all the examples. The comparison of the two strategies on the well-known transformation problem of class diagrams to relational schema showed that the rules obtained from the two strategies are interesting. Besides the first one produces rules which are more proper to their examples and apply well compared to the second one which builds more detailed rules but larger and more difficult to analyze and to apply.

1 Introduction

Model Driven Engineering is a recent paradigm emphasizing the use of models in the development process of an application. The models may be graphical or not, but they are all structured conforming to particular models named meta-models. In Model-Driven Development, several kinds of models are successively handled (e.g. requirements, use cases, classes, etc.) and models may be obtained one from each other, in an automated way, thanks to model transformations. A Model Transformation is a program that takes as input a model conforming to a source meta-model and produces as output another model conforming to a target meta-model. Implementing a model transformation requires a strong knowledge about model driven engineering (meta-modeling and model-transformation environments) and about the specification of the transformation: the input domain,

⋆ This work has been supported by project CUTTER ANR-10-BLAN-0219

the output domain and the transformation rules by themselves. Moreover, the transformation rules are usually defined at the meta-model level, which requires a clear and deep understanding about the abstract syntax and semantic interrelationships between the source and target models [7].

Domain experts generally do not have sufficient skills in model driven engineering. An innovative approach called Model Transformation By Example (MTBE) [12] is proposed to let them design model transformation by giving an initial set of examples. An example consists of an input model, the corresponding transformed model and links explaining which target element(s) one or several source model elements are transformed into. From these examples, transformation rules are deduced using a learning approach.

In this context, we presented a Model Transformation By Example approach that goes from examples to transformation patterns. The proposed learning mechanism is based on Relational Concept Analysis (RCA) [5], one of the extensions of Formal Concept Analysis [3], that considers links between objects in the concept construction. This approach results in a lattice of transformation patterns. Those patterns are filtered to keep the more relevant ones, and are made operational using a rule engine. In this paper, we analyze and compare two strategies for learning the transformation patterns with RCA. In the first one, each example is used alone to learn transformation patterns, and the transformation patterns obtained from all the examples are then gathered. In the second strategy, the examples are first gathered into a single large example, that is then used to learn the transformation patterns. The obtained transformation patterns are inspected and applied to test examples.

The remainder of this paper is structured as follows. Section 2 gives an overview of the approach. Section 3 briefly explains how RCA is used to generate transformation patterns from examples. Section 4 details how the pattern lattices are filtered and refined. Section 5 describes the two learning strategies, and an experimentation to compare them on a case study. Section 6 positions our work w.r.t. related work, and Section 7 concludes the paper and describes future work.

2 Learning and executing model transformation rules

A model transformation is a program handling and transforming models. We focus here on transformations that take a model as input and result in another model as output. Model transformations are usually written in languages dedicated to model handling, or generic languages using adequate frameworks to handle models. Those transformations implement transformation rules, expressing for each kind of elements from an input model which kind of elements of the output model they will be transformed into. Model-Transformation By Example (MTBE) consists in learning those transformation rules from examples: instead of programming the model transformation, the user designs examples illustrating the behavior of the transformation, and the transformation rules are automatically learned from those examples.

Usually, an example is composed of a source model, the corresponding transformed model, and transformation links between those two models. To illustrate MTBE, we consider the well-known case of transforming UML class diagrams into entity relationship. An example for this transformation is thus composed of a UML model (the input model), a Relational model (the output model) and transformation links making explicit from which elements of the UML model, the elements of the entity relationship model stem from. Such an example is given in Figure 1. The input UML model is on the l.h.s., and is composed of two classes named `Text` and `Style`, each one owning an attribute (respectively named `Title` and `Name`) and linked with an association named `Has a`. The output model (on the r.h.s. of Figure 1) has two entities `Text` and `Style`, each one described by an attribute, linked with a relation named `Has a`. The dotted lines show some of the transformation links. For instance, there is a transformation link specifying that the class `Text` is mapped into the entity `Text`.

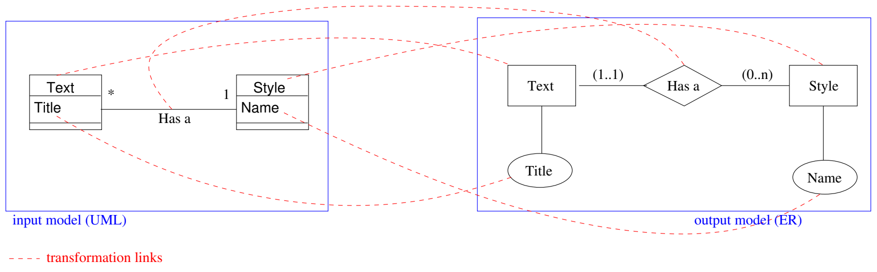


Fig. 1. Example for the UML2ER transformation: input model (l.h.s), output model (r.h.s.) and transformation links (dotted lines)

An MTBE process analyzes the examples and learns from them transformation rules such as *a class is transformed into an entity*, or *a UML property linked to a class (i.e., an attribute and not a role) is transformed into a role of an entity*.

We proposed an MTBE approach in which the learning mechanism relies on Relational Concept Analysis [2]. The abstract learned rules are named *transformation patterns*, they are obtained in a *transformation patterns lattice*, and are then filtered so as to select the more relevant ones. To make those transformation patterns operational so as to be able to execute the learned transformation, we designed a transformation from the transformation patterns to Jess rules [6], Jess being a rule engine [11].

3 RCA and transformation patterns discovery

As stated in Section 2, a key step in our MTBE approach consists in generating transformation patterns. Such patterns describe how a source model element is transformed into a target model element, within a given source context and a given target context. To derive patterns from examples, a data analysis method is used, namely Formal Concept Analysis (FCA) [3] and its extension to relational data, the Relational Concept Analysis (RCA) [5]. Both Formal and Relational Concept Analysis, also used for data mining problems, group entities described by characteristics into concepts, ordered in a lattice structure. While FCA produces a single classification from one formal context, RCA computes several connected classifications from several formal contexts linked by relational contexts.

Definition 1 (Relational Context Family). *A* Relational Context Family $\mathfrak{R}$ *is a couple* (K, R). *K is a set of* Object-Attribute Contexts $K_i = (O_i, A_i, I_i)$ *where O_i is a set of objects, A_i is a set of attributes and $I_i \subseteq O_i \times A_i$. R is a set of* Object-Object contexts $R_j = (O_k, O_l, I_j, S_j)$ *where* $(O_k, A_k, I_k) \in K$, $(O_l, A_l, I_l) \in K$, $I_j \subseteq O_k \times O_l$, *and S_j is a scaling operator,* i.e. *a boolean function taking as parameter an object from O_k, a concept extent $e \subseteq O_l$ and the binary relation I_j.*

RCA considers a Relational Context Family $\mathfrak{R} = (K, R)$ as input of the lattice building process. This process applies iteratively FCA on each Object-Attribute Context from K extended with the Object-Object contexts of R scaled with the lattices of the previous iteration.

Initialization step At the first step, FCA is applied on each Object-Attribute Context $K_i = (O_i, A_i, I_i)$ to produce a lattice L_{0i}. The output of this step is a Concept Lattice Family $\mathfrak{L}_0$.

Step n+1 At step n+1, from each context $R_j = (O_k, O_l, I_j, S_j)$ from R and the lattice L_{nl} we compute an Object-Attribute Context $C_{(n+1)j} = (O_k, \{j\} \times L_{nl}, J_{(n+1)j})$ where $J_{(n+1)j} = \{(o, (j, c)) | o \in O_k \wedge c \in L_{nl} \wedge S_j(o, extent(c), I_j)\}$. Then each context $K_i = (O_i, A_i, I_i)$ from K is extended to obtain $K_{(n+1)i} = (O_i, A_{(n+1)i}, I_{(n+1)i})$ where $A_{(n+1)i} = A_i \bigcup_{\{p,q | \exists R_q = (O_i, O_p, I_q, S_q) \in R\}} \{q\} \times L_{np}$ and $I_{(n+1)i} = I_i \bigcup_{\{q | \exists R_q = (O_i, O_p, I_q, S_q) \in R\}} J_{(n+1)q}$. The lattice $L_{(n+1)i}$ is then obtained by applying FCA on $K_{(n+1)i}$.

The process stops when an iteration does not add any new concept and we consider the last lattice family obtained as the output of the process.

We use RCA to classify: the source model elements, the target model elements and the transformation links. Which means that every one of them will be modelled as an Object-Attribute context in RCA. Those contexts will be linked by Object-Object contexts modelled after the following relations. Source and target model elements are classified using their metaclasses and relations. The transformation link classification relies on model element classifications and groups links that have similarities in their source and target ends: similar elements in similar contexts. From the transformation link classification, we derive a lattice of transformation patterns. Figure 2 shows an excerpt of the obtained pattern lattice for the transformation of UML class diagrams into relational models. The transformation patterns are represented by rectangles, and are named with the prefix TPatt, the number of the transformation pattern, and then the number of the corresponding concept.

In each concept representing a transformation pattern, we have two types in two ellipses connected by a bold edge. The source ellipse of the bold edge represents the type T_s of the element to transform by the pattern. It can be seen as the main type of the premise. For instance, in Concept `TPatt_17-Concept_67`, we see that the pattern aims at transforming *generalizations* (note that in the UML meta-model, there is a meta-class named Generalization, which represents an inheritance relationship, and which is linked to two classes (in fact: classifiers, `Classifier` being a superclass of the meta-class `Class`) : the specific one and the general one). This main type of the premise is linked, with non-bold edges, to the environment that an element of type T_s must have in order to be transformed by the pattern. Those edges are named according to the relation-role names between the type T_s and its environment in the meta-model. Those edges also have a cardinality defining the cardinality of the environment. Such an environment corresponds to the rest of the premise. For instance, in Concept `TPatt_17-Concept_67`, `Generalization` is linked to a specific `Class` and a general `Class` with a cardinality [1..*], meaning that the Generalization must have a specific and a general classes. The target ellipse of the bold edge represents the main type T_t of the conclusion of the pattern, *i.e.*, a T_s will be transformed into a T_t (with a specific environment). For example, in the transformation pattern `TPatt_17-Concept_67`, the conclusion corresponds to a role, connected to a relationship, an entity, and zero or one cardinalities. Note that the conclusion of this pattern is quite long: this will be discussed in the next section.

4 Patterns lattices simplification

The obtained lattice of transformation patterns has to be filtered to keep only the useful/relevant patterns or pattern fragments.

First, the empty concepts are removed. They do not contain information about the transformation. They are present in the lattice to link other concepts (representing patterns). We only keep the Bottom and Top to maintain the order structure. For the same reason, when an empty concept is removed, its children are connected with the Top concept.

Secondly, we noticed that some patterns contain a deep premise or conclusion, *i.e.*, a long chain of linked objects. After observing many patterns of this type for many transformation problems, we found that after a certain depth, the linked elements are not useful. For instance, in the pattern `TPatt_17-Concept_67` in Figure 2, the important information is that a generalization linked to two classes (specific and general) must be transformed into a role linked to a relationship, an entity and a cardinality. The other elements are details specific to some examples, that are not relevant to the transformation. Starting from this observation, we implemented a simplification heuristic that prunes the premises and conclusions

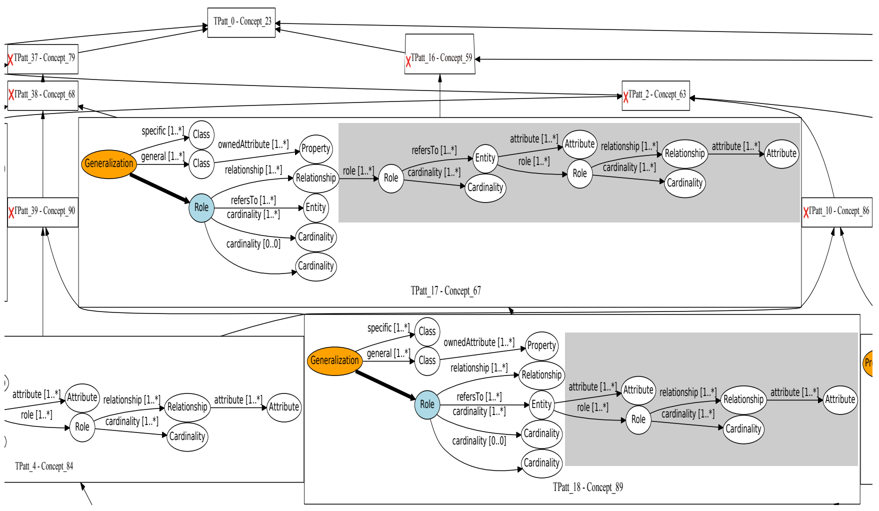


Fig. 2. An excerpt of the obtained hierarchy for the example UML class diagrams to entity-relationship model. Grey boxes indicate the part of the patterns that will be removed during the filtering phase, red crosses indicate the patterns that will be removed.

after the first level (key element and its immediate neighbors). In Figure 2, the grey boxes indicate the part of the patterns that will be removed.

After pruning the patterns according to the depth heuristic, some patterns could become identical. This is the case of patterns `TPatt_17-Concept_67` and `TPatt_18-Concept_89`). In Figure 2, we see the kept part of those two patterns (that is not in grey rectangles) is identical. For those redundant patterns, only the highest in the lattice is preserved, and all others removed. For removed concepts, their children are linked to their parents. Note that by doing so, we may lose the lattice structure.

5 Experimentation

In this section we experimentally compare transformation patterns obtained from distinct examples to transformation patterns obtained by the union of all examples. Our case study concerns the transformation of class diagrams into relational schema. The rule generation is performed starting from a set of 30 examples of class diagrams and their corresponding relational models. Some of them were taken from [7], the others were collected from different sources on the Internet. We ensured by manual inspection that all the examples conform to valid transformations. To take the best from the examples, a 3-fold cross validation was performed. We divide the j ($j \in 1..30$) examples into three groups of 10. For each fold i ($i \in 1..3$), we use two strategies to produce transformation rules:

– In the first one, we use the experimentation of [11] which consists of using two groups (20 examples) separately for generating 20 pattern lattices (denoted l_{ij}). The l_{ij} lattices are analyzed and simplified, as explained in Section 4, to select automatically the relevant transformation patterns. Then, we transform them into operational rules written for Jess. The remaining third group is used for testing them. Testing consists in executing the generated rules on the source models of the testing examples and in comparing the obtained target models with those provided in the examples.
– In the second one, we gather two groups (20 examples) for generating only one lattice of patterns (denoted L_i). L_i is analyzed and simplified to select automatically the relevant patterns. Those patterns are then transformed into operational rules. The remaining third group is used for testing them.

The goal is to compare in each fold i the results obtained from the two strategies. First, we compare the lattices generated from examples (lij), and the lattice generated from the union of those examples (Li). Then, we compare the results of executing the rules obtained from each strategy on the source models provided in the testing examples.

5.1 lij vs L_i

Compared to the first strategy, which produces small size lattices (from each lij we have about 9 patterns before simplification and 4 patterns after simplification), the second one produces large ones (from each L_i we have about 100 patterns before simplification and 50 patterns after simplification). Although the lattices L_i are larger and more difficult to analyze, they have more specific and complete transformation patterns compared to lij which are simple to analyze but contain transformation patterns that are proper to their examples. A single pattern of L_i can combine several patterns that exist in l_{ij}.

Figure 3 shows examples of different patterns obtained from l_{1j}. For instance, in the pattern of Fig. 3(a), a transformation link is given to specify that a *class* linked to an *aggregation* is mapped into a *table* linked to *primary foreign key.* Pattern of Fig. 3(b) shows that a class linked to a *property* is transformed into a table linked to a *column.* In the last pattern of Fig. 3(c), the transformation specifies that a class linked to a property and a *generalization* are transformed into a table linked to a column and a *foreign key.*

If we compare these patterns with the pattern of lattice L_1 in Figure 4, we note that the information contained in the three patterns exist in the pattern of Figure 4. It is more complete. It combines all the informations of transformation existing in Fig. 3(a), Fig. 3(b) and Fig. 3(c).

So, if we combine various examples together, the generated lattice contains patterns which are more specific and combine different information. But, if we test each example separately, the obtained lattice contains less information. In addition, L_i contains all the patterns needed to transform a class diagram to a relational schema. The lattices l_{ij} contain just the transformation pattern

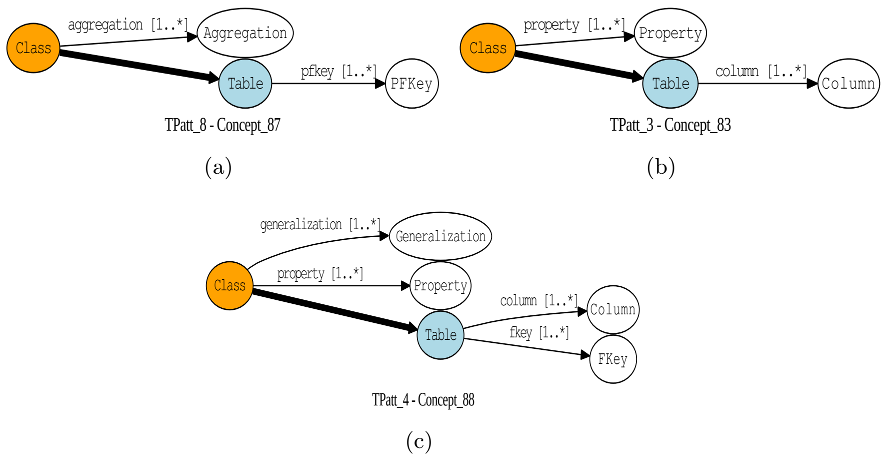


Fig. 3. Examples of transformation patterns extracted from lattices L_{l1}

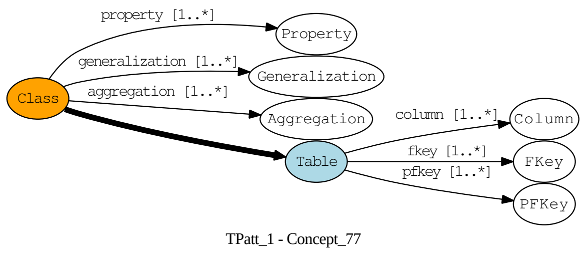


Fig. 4. Example of transformation pattern extracted from lattice L_1

proper to the transformation examples used. So, we need to merge several transformation examples to obtain all transformation rules of a class diagram into a relational schema.

Furthermore, in each fold, a L_i lattice contains about 50 transformation patterns and the union of l_{ij} produces about 40 ones (4 transformation patterns * 20 minus the redundant ones which exist). If we examine the patterns as an expert, we note that L_i contains about 12 relevant transformation patterns which are useful to the transformation. But they are less detailed and not applicable for all examples types. On the other side, the union of l_{ij} contains about 10 relevant transformation patterns. Those patterns are easy to read and to apply because each one contains a piece of information of the transformation compared to L_i'rules which combine several pieces of information in the same pattern.

5.2 *lij*'s rules execution vs L_i's rules execution

In this section, we compare the result of executing the rules obtained from the two strategies, which are transformed into Jess rules, on the source models provided in the testing examples. This comparison allows calculating the recall (Equation 1) and the precision (Equation 2) measures for each T. T represents the type of elements in the target meta-model (table, column, foreign key...)

$$R(T) = \frac{number\ of\ T\ with\ correct\ transformation}{total\ number\ of\ initial\ T} \tag{1}$$

$$P(T) = \frac{number\ of\ T\ with\ correct\ transformation}{total\ number\ of\ generated\ T} \tag{2}$$

Table 1 shows precision and recall averages on all element types of the 10 generated transformations for the 3-folds. As mentioned in [11], the precision and recall averages are higher than 0.7 in the first strategy. Some models were perfectly transformed (precision=1 and recall=1). Precision and recall decrease in the case of elements which have more than one transformation possibility. For example, if we have a generalization between two classes, we can transform it into two tables or into a simple table which contains the attributes of general and specific classes. In this case, two rules are applied on the same example and this affects the performance results.

In the second strategy, precision and recall averages are low (less than 0.5) in the 3-folds. This is due to the fact that the generated rules are very large and contain different informations from different examples. Thus, the premises of the rules can not be matched for most of the examples because the examples are simple and do not contain all the transformation cases that have been learned. So, the Jess rule engine does not apply a part of the rule premise when it is executed on an example, it searches for each example its corresponding rule and this decreases the precision and the recall.

5.3 Discussion

The study presented in this section is a comparison of two strategies for generating transformation rules using RCA. The first consists to generate from each example its rule lattice and the second consists to gather all the examples and generate only one rule lattice. Each one has its advantages and disadvantages:

- The first strategy produces simple and small transformation patterns which are easy to analyze and to manipulate, but they are proper to their examples. On the other side, the second one produces larger patterns but they are more specific and more complete. They combine different information about the transformation. An analysis on those patterns shows that the two strategies have the same number of relevant patterns (about 12 transformation patterns). The relevant ones of the first strategy are simple and applicable for each example. On the contrary, the relevant patterns of the second strategy are larger and mainly applicable for larger examples.

Examples	Fold1			
	Recall Average		Precision Average	
	First Strategy	Second Strategy	First Strategy	Second Strategy
1	1	0.5	1	0.5
2	0.77	0.45	0.75	0.43
3	0.70	0.5	0.75	0.43
4	0.94	0.43	0.75	0.32
5	1	0.45	1	0.43
6	1	0.5	0.77	0.23
7	0.88	0.43	0.77	0.40
8	1	0.6	0.77	0.43
9	0.90	0.5	0.77	0.44
10	0.90	0.5	0.85	0.45

Examples	Fold2			
	Recall Average		Precision Average	
	First Strategy	Second Strategy	First Strategy	Second Strategy
1	0.78	0.5	0.79	0.4
2	0.90	0.45	0.75	0.31
3	0.85	0.45	0.77	0.43
4	0.77	0.43	0.79	0.40
5	1	0.5	0.80	0.34
6	1	0.43	0.77	0.47
7	0.85	0.4	0.77	0.37
8	0.85	0.45	0.80	0.43
9	1	0.5	0.75	0.34
10	1	0.5	0.80	0.33

Examples	Fold3			
	Recall Average		Precision Average	
	First Strategy	Second Strategy	First Strategy	Second Strategy
1	0.80	0.49	0.75	0.33
2	1	0.44	1	0.34
3	1	0.5	0.85	0.44
4	1	0.45	0.80	0.44
5	0.77	0.40	0.75	0.35
6	1	0.5	0.77	0.40
7	1	0.4	1	0.33
8	1	0.33	0.80	0.23
9	0.85	0.35	0.77	0.3
10	0.88	0.4	0.80	0.39

Table 1. Result of 3-fold cross validation

– The execution of the rules of the first strategy gives good results in our experiment. The rule engine searches and finds for each example the set of rules to apply. On the contrary the rules of the second strategy are more large and contain more information. Thus the rule engine does not find a rule to apply for the simple examples. We can work again on the obtained rules of L_i to execute them on all types of examples (for example by separating into smaller pieces), but as we found good results with the union of l_{ij}, it is not a promising track. We obtained a non-intuitive result: before the experimentation, we thought the best rules would be obtained with the second strategy.

Although the example used is a classical one, it is a good example of typical model transformations that we aim to learn. To confirm what is the best strategy to produce transformation rules, additional experiments have to be conducted with other model transformation kinds. Besides, the obtained result depends on the models on which we execute the rules. If we use larger models, the second strategy may have better results.

6 Related Work

Writing model transformations requires time and specific skills: the transformation developer needs to master the transformation language and both transformation source and target meta-models. Model Transformation by Example is a recent field of research that intends to use models as artifacts of development of the transformation.

Most of the research works consider all the examples at once. In [1], the authors use inductive logics programming to derive transformation rules, and although they consider an iterative process where examples are added to complete the derived transformation, they don't consider the examples independently. The same remark applies to [13], whose work uses the constraints explicitly applied by the transformation from the concrete syntax of a language to its abstract syntax and for [4] who proposes an algorithm to produce many to many transformation rules.

Another track in MTBE consists in using the analogy to perform transformations using examples [8,9,10]. The provided examples are decomposed into transformation blocks linking fragments of source models to fragments of target models. When a new source model has to be transformed, its elements are compared to those in the example source fragments to select the similar ones. Blocks corresponding to the selected fragments, coming from different examples, are composed to propose a suitable transformation. Fragment selection and composition are performed through a meta-heuristic algorithm. Compared to the above-mentioned approaches, the analogy-based MTBE does not produce rules. Here the examples are considered differently as they are added separately and not as a whole, influencing incrementally the system.

7 Conclusion

In this paper, we studied an approach for inferring model transformations (composed of transformation rules) from transformation examples. We compare two strategies for applying this approach: inferring the rules from the example taken separately (then gathering the rules), or inferring the rules from the gathering of the examples. Although we thought the second strategy would produce better rules (more detailed), it appeared that the rules (less detailed) produced by the first approach execute better. Future work includes learning rules whose premise and conclusion have several main elements and design heuristics to determine the best rule to apply when several rules are candidate.

References

1. Balogh, Z., Varró, D.: Model transformation by example using inductive logic programming. Software and Systems Modeling 8(3), 347–364 (2009)
2. Dolques, X., Huchard, M., Nebut, C.: From transformation traces to transformation rules: Assisting model driven engineering approach with formal concept analysis. In: Supplementary Proceedings of ICCS'09. pp. 15–29 (2009)
3. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical Foundations. Springer (1999)
4. García-Magariño, I., Gómez-Sanz, J.J., Fuentes-Fernández, R.: Model transformation by-example: An algorithm for generating many-to-many transformation rules in several model transformation languages. In: ICMT. pp. 52–66 (2009)
5. Huchard, M., Hacène, M.R., Roume, C., Valtchev, P.: Relational concept discovery in structured datasets. Ann. Math. Artif. Intell. 49(1-4), 39–76 (2007)
6. Jess rule engine, http://herzberg.ca.sandia.gov/jess
7. Kessentini, M.: Transformation by Example. Ph.D. thesis, University of Montreal (2010)
8. Kessentini, M., Sahraoui, H., Boukadoum, M.: Model Transformation as an Optimization Problem. In: MODELS'08, LNCS 5301. pp. 159–173. Springer (2008)
9. Kessentini, M., Sahraoui, H., Boukadoum, M.: Méta-modélisation de la transformation de modèles par l'exemple : approche méta-heuristiques. In: Carré, B., Zendra, O. (eds.) LMO'09: Langages et Modèles à Objets. pp. 75–90. Cepaduès, Nancy (mars 2009)
10. Kessentini, M., Sahraoui, H., Boukadoum, M., Omar, B.O.: Model transformation by example : a search-based approach. Software and Systems Modeling Journal (2010), (To appear)
11. Saada, H., Dolques, X., Huchard, M., Nebut, C., Sahraoui, H.: Generation of operational transformation rules from examples of model transformations. In: to appear in proc. of MODELS'12 (Sept 2012)
12. Varró, D.: Model transformation by example. In: Proc. MODELS 2006, LNCS 4199. pp. 410–424. Springer (2006)
13. Wimmer, M., Strommer, M., Kargl, H., Kramler, G.: Towards model transformation generation by-example. In: HICSS. p. 285 (2007)

Annotating Lattice Orbifolds with Minimal Acting Automorphisms

Tobias Schlemmer

Technische Universität Dresden,
Fachrichtung Mathematik, 01069 Dresden, Germany
`Tobias.Schlemmer@mailbox.tu-dresden.de`
`http://www.math.tu-dresden.de/~schlemme/`

Abstract Context and lattice orbifolds have been discussed by M. Zickwolff [1,2], B. Ganter and D. Borchmann[3,4]. Preordering the folding automorphisms by set inclusion of their orbits gives rise to further development. The minimal elements of this preorder have a prime group order and any group element can be dissolved into the product of group elements whose group order is a prime power. This contribution describes a way to compress an orbifold annotation to sets of such minimal automorphisms. This way a hierarchical annotation is described together with an interpretation of the annotation. Based on this annotation an example is given that illustrates the construction of an automaton for certain pattern matching problems in music processing.

Key words: formal concept lattice, lattice orbifold, annotation, automorphism group

1 Introduction

Lattice orbifolds have been described by Monika Zickwolff [1,2] as a useful tool for the compression of formal concept lattices. Daniel Borchmann has extended this theory in his diploma thesis to context orbifolds [3,4]. A binary relation orbifold can be considered as a mathematical structure on the sets of orbits of a given group of automorphisms of a binary relation structure that allows to reconstruct the original relation. Thus, it can provide deeper insight into the structure of such a relation. On the other hand it provides the means for compressing a relational structure in a way that preserves the possibility for certain algorithms to act on it.

In the work of Zickwolff, Ganter and Borchmann together with the theory also a method of data compression by means of the stabilisers in the automorphism group has been provided. This abridged annotation contains the automorphisms that violate a certain kind of symmetry which can be described by the stabilisers of the equivalence classes. Thus, it provides an insight how the lattice violates the symmetry reflected by the folding group.

The latter approach starts from a global view at the orbifold and cuts out redundant information treating all nodes in the Hasse diagram equally. Properties like direction are not used for this kind of annotation.

The work presented here, starts from a local view (a pair of neighbours) and uses both direction and transitivity of the relation to minimise the annotation. In this way it spares the necessity to save the stabilisers increasing the information provided by the edge annotation in comparison with the classical abridged annotation.

After some theoretical section an example is given that provides further insight into applications of the theory provided in this article.

2 Preliminaries

If not stated otherwise algebraic structures are be denoted by double stroke letters, the base set of an algebraic structure $\mathbb{A}$ by the same letter A in normal font. $\mathfrak{Aut}\,\mathbb{A}$ is its automorphism group and $\mathbb{1} := \big(\{(1)\}, \cdot, (1)\big)$ the trivial group. For any permutation group $\mathbb{G}$ on a set A we denote the set of its orbits by $A \mathbin{\backslash\!\backslash} \mathbb{G} := \{x^{\mathbb{G}} \mid x \in A\}$. Obviously, for any group element $g \in G$ and any orbit $U \in A \mathbin{\backslash\!\backslash} \mathbb{G}$ also its adjoint gUg^{-1} is an orbit. Throughout this paper we refer to the set of volatile points of an automorphism $g \in \mathfrak{Aut}\,\mathbb{A}$ as $\mathfrak{Var}\,g := \{x \in M \mid x^g \neq x\}$ and to its set of fixed points using the notation $\mathfrak{Fix}\,g := \{x \in M \mid x^g = x\}$. Obviously, for any element $g \in \mathfrak{Aut}\,\mathbb{A}$ the equation $A = \mathfrak{Fix}\,g \cup \mathfrak{Var}\,g$ holds. We say that a permutation $g \in G$ acts semiregular on a set $M \subseteq A$, iff $M \setminus \mathfrak{Var}\,g \in \{\emptyset, M\}$ is true. Note that $\mathfrak{Var}\,g$ is not restricted to be a subset of M.

If $(M, \leq)$ is an ordered set and $N \subseteq M$ then the corresponding order ideal is defined by the set $\downarrow_{\leq} N := \{1 \in M \mid \exists y \in N : x \leq y\}$ and we write $\downarrow_{\leq} x$ for $\downarrow_{\leq}\{x\}$ if the context is clear. The neighbourhood relation is denoted by the symbol $\prec$.

As defined in [3] an orbifold of an ordered set $(M, \leq_M)$ will be denoted by a triple $(M \mathbin{\backslash\!\backslash} \mathbb{G}, \leq, \lambda)$ where $x^{\mathbb{G}} \leq y^{\mathbb{G}} :\Leftrightarrow \exists g \in G : x \leq_M y^g$ and λ is an annotational function which carries additional information that allows to reconstruct the relation $\leq_M$. The most generic choice of λ is defined in the same article as a mapping $\lambda : (M \mathbin{\backslash\!\backslash} \mathbb{G}) \times (M \mathbin{\backslash\!\backslash} \mathbb{G}) \to \mathfrak{P}\,G$ which fulfils the condition $\lambda(x^{\mathbb{G}}, y^{\mathbb{G}}) = \{g \in G \mid x \leq_M y^g\}$ for any $x, y \in M$. In this setting the stabiliser $\mathbb{G}_x$ of an Element x can be retrieved from its annotation $\lambda(x^{\mathbb{G}}, x^{\mathbb{G}})$.[1]

A simplified description of order orbifolds is defined utilizing the fact

$$\lambda(x^{\mathbb{G}}, y^{\mathbb{G}}) \setminus \bigcup_{x^{\mathbb{G}} < z^{\mathbb{G}} < y^{\mathbb{G}}} \lambda(x^{\mathbb{G}}, z^{\mathbb{G}}) \cdot \lambda(z^{\mathbb{G}}, y^{\mathbb{G}}) = \bigcup_{\substack{g \in \lambda(x,y) \\ g \notin \lambda(x^{\mathbb{G}}, z^{\mathbb{G}}) \cdot \lambda(z^{\mathbb{G}}, y^{\mathbb{G}}) \\ x^{\mathbb{G}} < z^{\mathbb{G}} < y^{\mathbb{G}}}} \mathbb{G}_x \cdot g \cdot \mathbb{G}y.$$

This consists of a transversal T of $M \mathbin{\backslash\!\backslash} \mathbb{G}$ and an abridged annotaton function $\lambda_{abr} : T \times T \to \mathfrak{P}\,G$, where $\lambda_{abr}(x, y)$ is a set of double coset representatives, i. e. it is a minimal set such that $\mathbb{G}_x \cdot \lambda_{abr}(x, y) \cdot \mathbb{G}_y = \lambda(x, y)$.

It is well known that orbits of automorphisms of a finite ordered set are always antichains in this set.

[1] For other settings see [1,2,4].

3 Minimal Acting Automorphisms

Let $\mathbb{A}$ be an algebraic structure. The orbits of the cyclic subgroups of $\mathfrak{Aut}\,\mathbb{A}$ can be used to preorder the automorphism group. If we refer to some group $\mathbb{G}$ or its base set G without further notice it is always meant to be $\mathbb{G} \leq \mathfrak{Aut}\,\mathbb{A}$.

Lemma 1. *Let $\mathfrak{Aut}\,\mathbb{A}$ the automorphism group of a finite algebraic structure $\mathbb{A}$. Then the binary relation $\sqsubseteq\, \subseteq \mathfrak{Aut}\,\mathbb{A} \times \mathfrak{Aut}\,\mathbb{A}$ defined by*

$$g \sqsubseteq h :\Leftrightarrow \forall U \in (A \,\backslash\!\backslash\, \langle g\rangle)\exists U' \in (A \,\backslash\!\backslash\, \langle h\rangle) : U \subseteq U'$$

is a preorder.

Proof. This follows directly from the definition: Reflexivity is obivious as the equation $A \,\backslash\!\backslash\, \langle g\rangle = A \,\backslash\!\backslash\, \langle g\rangle$ holds. Given three automorphisms $f, g, h \in \mathfrak{Aut}\,\mathbb{A}$ such that for each orbit $U \in (A \,\backslash\!\backslash\, \langle f\rangle)$ there exists an orbit $U' \in (A \,\backslash\!\backslash\, \langle g\rangle)$ with $U \subseteq U'$. If the same condition is true for the pair (g, h) we can find an orbit $U'' \in (A \,\backslash\!\backslash\, \langle h\rangle)$ such that $U \subseteq U' \subseteq U''$. Thus transitivity holds, too. □

As in any cyclic subgroup the implication $\langle g^n\rangle \subseteq \langle g\rangle \Rightarrow x^{\langle g^n\rangle} \subseteq x^{\langle g\rangle}$ holds, we can fix the following corollary:

Corollary 1. *For any group element $g \in \mathfrak{Aut}\,\mathbb{A}$ and any natural number we get: $g^n \sqsubseteq g$*

In particular, this means that $g \in \mathfrak{Aut}\,\mathbb{A}$ and $n \in \mathbb{N}$ imply $\mathfrak{Var}\, g^n \subseteq \mathfrak{Var}\, g$.

On the other hand, the so defined relation $\sqsubseteq$ is usually no order relation as for any $g \in \mathfrak{Aut}\,\mathbb{A}$ we have $A \,\backslash\!\backslash \langle g\rangle = A \,\backslash\!\backslash \langle g^{-1}\rangle$, but in general the equation $g = g^{-1}$ does not hold.

If A is finite, then the preordered set $(\mathfrak{Aut}\,\mathbb{A}, \sqsubseteq)$ has minimal elements.

Definition 1. *Let $\mathbb{A}$ be a finite algebraic structure. The minimal elements of the preordered set $(\mathfrak{Aut}\,\mathbb{A}, \sqsubseteq)$ are called* automorphisms with minimal action*.*

Corollary 2. *Let g be minimal in $(G, \sqsubseteq)$, then the cyclic group $\langle g\rangle$ acts semiregular on $\mathfrak{Var}\, g$.*

Proof. Suppose $\langle g\rangle$ does not act semiregular on $\mathfrak{Var}\, g$. Then there exist elements $\exists x, y \in \mathfrak{Var}\, g$ and a positive integer $n \in \mathbb{N} \setminus \{0\}$ such that $x^{g^n} = x, y^{g^n} \neq y$. This implies $x^{\langle g^n\rangle} = \{x\} \neq x^{\langle g\rangle}$. Thus, $x^{\langle g^n\rangle} \notin A \,\backslash\!\backslash\, \langle g\rangle$. Consequently, $g^n \sqsubseteq g$ and $g \not\sqsubseteq g^n$. Thus, g is not minimal. □

Corollary 3. *Let $g \in G$ be minimal in $(G, \sqsubseteq)$. Then for any element $h \in G$: huh^{-1} is minimal, too.*

Proof. Suppose the existence of an element $v \in G$ such that the set of its orbits $A \,\backslash\!\backslash \langle v\rangle$ is a refinement of $A \,\backslash\!\backslash \langle gug^{-1}\rangle$. Then, $A \,\backslash\!\backslash \langle g^{-1}vg\rangle = (A \,\backslash\!\backslash \langle v\rangle)g$ is a refinement of $A \,\backslash\!\backslash \langle u\rangle$, as $A = A^{g^{-1}}$. This means that u is not minimal. □

The set containing the minimal nontrivial automorphisms of $(G, \sqsubseteq)$ will be denoted by $\mathfrak{Min}(G, \sqsubseteq)$, in particular we define

$$(1) \in \mathfrak{Min}(G, \sqsubseteq) \text{ iff } \mathfrak{Min}(G, \sqsubseteq) \setminus \{(1)\} = \emptyset \text{ and } G \neq \emptyset.$$

Corollary 4. *The subgroup $\langle \mathfrak{Min}(G, \sqsubseteq) \rangle$ is a normal subgroup in $\mathbb{G}$.*

Hall's theorem [5] tells us that we can dissolve each cyclic subgroup of $\mathbb{G}$ into a product of cyclic groups whose orders are prime powers. Thus, we can generate each cyclic group by a set of elements with pairwise coprime orders. This leads us to the following corollary:

Corollary 5. *Let $\mathbb{G} \leq \mathfrak{Aut}\,\mathbb{A}$. Then the set*

$$P := \bigl\{ g \in G \bigm| |\langle g \rangle| \textit{ is a prime power} \bigr\}$$

is a generating set of $\mathbb{G}$.

For the construction of the Sylow groups, we can use the following lemma:

Lemma 2. *Let $g \in \mathfrak{Aut}\,\mathbb{A}$ an automorphism of finite order $n \in \mathbb{N} \setminus \{0\}$ and $g_1, g_2 \in \langle g \rangle$ with $|\langle g_1 \rangle| = m_1$ and $|\langle g_2 \rangle| = m_2$. Then $\langle g^{\gcd(n/m_1, n/m_2)} \rangle \leq \langle g_1, g_2 \rangle$.*

Proof. As cyclic groups are Abelian, there exist integers $a, b \in \mathbb{Z}$ such that $\gcd(\frac{n}{m_1}, \frac{n}{m_2}) = a\frac{n}{m_1} + b\frac{n}{m_2}$. Since $g_1 \in \langle g^{n/m_1} \rangle$ and $g_2 \in \langle g^{n/m_2} \rangle$, w. l. o.g. we can assume $g_1 = g^{n/m_1}$ and $g_2 = g^{n/m_2}$. Thus, we get $g^{\gcd(n/m_1, n/m_2)} = g_1^a g_2^b$. □

Consequently, the generating set of Corollary 5 contains all minimal elements $\mathfrak{Min}(G, \sqsubseteq)$.

Lemma 3. *Let $\mathbb{G} \leq \mathfrak{Aut}\,\mathbb{A}$ be a finite automorphism group. Then the elements of $\mathfrak{Min}(\mathbb{G}, \sqsubseteq)$ have prime order.*

Proof. Let $|\langle g \rangle| = p^n$ where p is prime. Then $\bigl|\bigl\langle g^{p^{n-1}} \bigr\rangle\bigr| = p$. As $\langle g \rangle$ is cyclic, its order is the least common multiple of the sizes of its orbits. Thus, it has at least one orbit of size p^n, while the orbits in $\langle g^{n-1} \rangle$ have either one or p elements. Corollary 1 tells us, that the minimal Elements of $(G, \sqsubseteq)$ are those of prime orders. □

4 Automorphisms of ordered sets

Let $\mathbb{G} \leq \mathfrak{Aut}\,\mathbb{A}$ and $U_1, U_2 \leq \mathbb{G}$ such that $U_1 \cdot U_2 = \mathbb{G}$. Considering the implied action of $\mathbb{G}$ on $\mathfrak{P}\,A$, a straight forward calculation shows for all $X \subseteq \mathfrak{P}\,A$ that $X \in (A \setminus\!\!\setminus U_1) \setminus\!\!\setminus U_2$ iff $\bigcup X \in A \setminus\!\!\setminus \mathbb{G}$.
Let us consider some additional properties of automorphisms of finite lattices.

Definition 2. *Let $(M, \leq)$ be an ordered set, $\mathbb{G} \leq \mathfrak{Aut}(M, \leq)$, and $x \in M$. The set*

$$\mathbb{G}_{\downarrow_{\leq} x} := \{g \in G \mid \mathfrak{Var}\, g \cap \downarrow_{\leq} x = \emptyset\} \tag{1}$$

is called downwards stabiliser of x in $\mathbb{G}$.

Obviously the downwards stabiliser of a maximal element in a complete lattice is $\mathbb{1}$. In fact $\mathbb{G}_{\downarrow_{\leq} x}$ is a subgroup of the stabiliser $\mathbb{G}_x$ of x.

Corollary 6. *Let $\mathbb{V} = (V, \leq)$ be a finite lattice ordered set. And let $x \in V$ while $y, z \in V$ are upper neighbours of x. Furthermore, if there exist two automorphisms $g, h \in \mathbb{G}_{\downarrow_{\leq} x}$ with the property $z^g = y = z^h$, then the equation $(\downarrow_{\leq} y)^{h^{-1}g} = \downarrow_{\leq} y$ holds.*

Proof. We know that $y^{h^{-1}g} = y$. So for any $a \in \downarrow_{\leq} y$ we know $a^{h^{-1}g} \leq y$ as g and h are automorphisms. $\square$

Thus, if two elements are in the same orbit their downwards stabilisers are related by conjugation. This proves the following lemma:

Lemma 4. *Let $\mathbb{V} = (V, \leq)$ a finite lattice ordered set, $\mathbb{G} \leq \mathfrak{Aut}\,\mathbb{V}$, and let $x, y, z \in V$ while y and z are upper neighbours of x. Any automorphism g with $z^g = y$ is an automorphism mapping $V \setminus \downarrow_{\leq} z$ to $V \setminus \downarrow_{\leq} y$, while the equation $\mathbb{G}_{\downarrow_{\leq} y} = g^{-1}\mathbb{G}_{\downarrow_{\leq} z}g$ holds.*

Proof. Let $g \in \mathbb{G}_{\downarrow_{\leq} x}$ with $z^g = y$ and let $h \in \mathbb{G}_{\downarrow_{\leq} z}$. Then for any $a \in V\setminus\downarrow_{\leq} z$ and any $b \in V\setminus\downarrow_{\leq} y$ we get $a^g \in V\setminus\downarrow_{\leq} y$ and $b^{g^{-1}} \in V\setminus\downarrow_{\leq} z$ as g is an automorphism. As $V \setminus \downarrow_{\leq} y$ and $V \setminus \downarrow_{\leq} z$ are isomorphic by g, for any automorphism $h \in \mathbb{G}_{\downarrow_{\leq} z}$ the mapping $f := g^{-1}hg$ is an automorphism on $V \setminus \downarrow_{\leq} y$ and even $f \in \mathbb{G}_{\downarrow_{\leq} y}$, as it is constant for any $c \in \downarrow_{\leq} y$. Finally, we get $gfg^{-1} = h$. Thus, $\mathbb{G}_{\downarrow_{\leq} z}$ is a conjugate of $\mathbb{G}_{\downarrow_{\leq} y}$. The other direction of the implication is obvious. $\square$

As an immediate conclusion, we get that the downwards stabiliser of an element is contained in the union of the downwards stabilisers of its upper neighbours:

Corollary 7. *Let $\mathbb{V} = (V, \leq)$ a finite lattice ordered set, $\mathbb{G} \leq \mathfrak{Aut}\,\mathbb{V}$, and let $x \in V$ and $N = \{y \in V \mid x \prec y\}$ the set of upper neighbours of x. Let further T a transversal of $N \mathbin{\backslash\!\backslash} \mathbb{G}$ and $S \subseteq G$ such that $T^S = N$. Then $S \cdot \bigcup_{t \in T} \mathbb{G}_{\downarrow_{\leq} t} \subseteq \mathbb{G}_{\downarrow_{\leq} x}$.*

In other words: At any point in the lattice we can restrict ourselves to a local view. These considerations can be easily extended to finite ordered sets.

5 Orbifolds

In this section we define an orbifold representation using minimisations according to the preorder discussed in Section 3.

Definition 3. *Let $\lambda : T \times T \to \mathfrak{P}\, G$ be a mapping that assigns to each pair of elements from a finite set T to a subset of another set G. A* contiguous chain of λ from $x \in T$ to $y \in T$ *is defined as a subset $C \subseteq T$ such that the relation $\rho := \{(z, z') \in T \times T \mid \lambda(z, z') \neq \emptyset\}$ forms the neighbourhood relation of a linear order with minimal element x and maximal element y. The set of all such contiguous chains of λ between two elements x and y will be denoted by $\mathfrak{C}_{T,\lambda}(x, y)$.*

Further let the relation $\prec' \subseteq T \times T$ be defined by $x \prec' y :\Leftrightarrow \exists g \in G : x \prec y^g$. Using the non-commutative complex product $\prod$ in largest-left order, the operator $\Lambda_\lambda : T \times T \to \mathfrak{P}\, G$ is defined by $\Lambda(x, x) = \mathbb{1}$, and for $x \neq y$ by

$$\Lambda_\lambda(x, y) := \left\langle \bigcup \left\{ \prod_{i=2}^{|C|} \lambda(z_{i-1}, z_i) \;\middle|\; \begin{array}{c} C \in \mathfrak{C}_{T,\lambda}(x, y), z_{i-1} \prec z_i, \\ \{z_1, z_2, \ldots, z_{|C|}\} = C \end{array} \right\} \right\rangle \cap \mathbb{G}_{x,y}. \qquad (2)$$

Theorem 7 will provide us with another kind of annotation of an orbifold:

Definition 4. *Let $\mathbb{V} = (V, \leq)$ be a finite lattice ordered set, $\mathbb{G} \leq \mathfrak{Aut}\, \mathbb{V}$, T a transversal of $V \mathbin{\backslash\!\backslash} \mathbb{G}$, and the relation $\prec'$ defined as above.*

A mapping $\lambda_{hier} : T \times T \to \mathfrak{P}\, G$ is called hierarchical annotation *(of V, T and $\mathbb{G}$), if it fulfils the following conditions for all $x, y \in T$:*

1. *$\lambda_{hier}(x, y) \subseteq \mathbb{G}_{\downarrow_\leq x}$,*
2. *$\lambda_{hier}(x, y) = \emptyset$ if $\forall y' \in y^{\mathbb{G}} : x \not\prec' y'$, and*
3. *$x \prec' y$ implies $y^{\lambda_{hier}(x,y)} = y^{\mathbb{G}_{\downarrow_\leq x}}$*
4. *$y^{\lambda_{hier}(x,y) \Lambda_{\lambda_{hier}}(0,x)} = y^{\mathbb{G}_x}$.*

Corollary 8. *Let $\mathbb{V} = (V, \leq)$ be a finite lattice ordered set, $\mathbb{G} \leq \mathfrak{Aut}\, \mathbb{V}$, and λ a hierarchical annotation. Then the equation $\left\langle \bigcup_{x,y \in T} \lambda(x, y) \right\rangle \leq \mathbb{G}$ holds.*

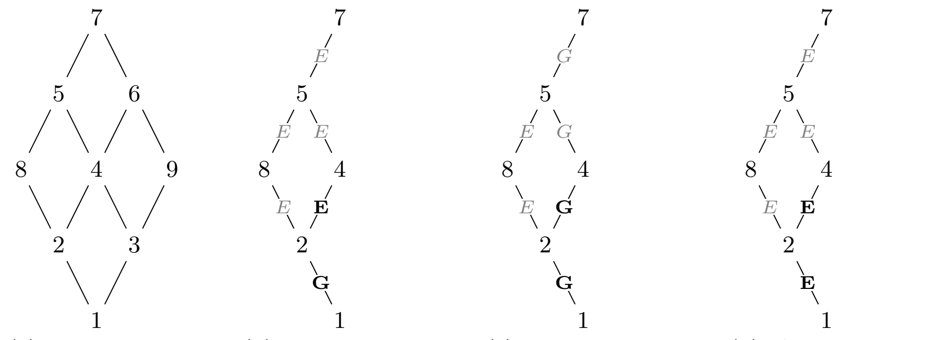
Example 1. Figure 1 shows a simple example of a lattice and its orbifold annotated with three different annotations. Besides the hierarchical annotation the annotations from Borchmann, Ganter and Zickwolff [1,2,3,4] have been included.

Before we explore some basic properties of hierarchical annotations we must assure their existence:

Lemma 5. *Let $\mathbb{V} = (V, \leq)$ be a finite lattice ordered set, $\mathbb{G} \leq \mathfrak{Aut}\, \mathbb{V}$ a group of automorphisms, and T a transversal of $V \mathbin{\backslash\!\backslash} \mathbb{G}$. Then there exists a hierarchical annotation $\lambda : T \times T \to \mathfrak{P}\, G$.*

Proof. For any $x, y \in T$ and any $z \in V$ with $x \prec y$, $x \prec z$, if $z \in y^{\mathbb{G}_{\downarrow_\leq x}}$ we fix an arbitrary $g_{x,y,z} \in \mathbb{G}_{\downarrow_\leq x}$ such that $z = y^{g_{x,y,z}}$. In that case we define $\lambda_1(x, y) := \{g_{x,y,z} \mid z \in y^{\mathbb{G}_{\downarrow_\leq x}}\}$. If $z \in y^{\mathbb{G}_x} \setminus y^{\mathbb{G}_{\downarrow_\leq x}}$ and $\mathbb{G}_{\downarrow_\leq x} = \emptyset$ then the orbits of y are predefined by all automorphisms that act below x thus, we define $\lambda_1(x, y) := \mathbb{1}$.

In any case where $z \in y^{\mathbb{G}_x} \setminus y^{\mathbb{G}_{\downarrow_\leq x}}$, there exists an automorphism $h_{x,y,z} \in \mathbb{G}_x$ such that $z \in y^{\lambda_1(x,y) h_{x,y,z}}$. Then there exist two elements $\hat{x}_z, \hat{y}_z \in {\downarrow_{\leq'}} x$ such



(a) Original order (b) Hierarchical annotation. (c) Full annotation. (d) Abridged annotation.

Figure 1. The lattice is folded by the group $G = \{(1), (2\,3)(5\,6)(8\,9)\}$. As you can see in 1(b), $\mathbb{G}_4 = G$, but $\lambda_{hier}(4,5) = \{(1)\}$. The singleton E is defined by $E := \{(1)\}$.

that $\hat{x}_z \prec' \hat{y}_z$ and there is an automorphism $h_{x,y,z} \in \mathbb{G}_{\downarrow_\leq \hat{x}_z} \setminus \mathbb{G}_{\hat{y}_z}$ and another automorphism $\hat{h}_{x,y,z} \in \Lambda(\hat{x}, x)$. Using this we define a second preliminary annotation $\lambda_2(\hat{x}_z, \hat{y}_z, x, y) := \{h_{x,y,z}\hat{h}^{-1}_{x,y,z} \mid z \in y^G \setminus y^{\mathbb{G}_{\downarrow_\leq x}}\}$.

For all other combinations of elements $\hat{x}, \hat{y}, x, y$ we set $\lambda_1(x,y) := \emptyset$ and $\lambda_2(\hat{x}, \hat{y}, x, y) := \emptyset$. Finally we define:

$$\lambda_{hier}(x,y) := \lambda_1(x,y) \cup \bigcup_{\tilde{x},\tilde{y} \in T, \tilde{z} \in \tilde{y}^{\mathbb{G}_x}} \lambda_2(x, y, \tilde{x}, \tilde{y}, \tilde{z}).$$

Obviously such a function exists and fulfils the conditions of Definition 4. □

Now, as we know how to describe the automorphism group $\mathbb{G} \leq \mathfrak{Aut}\,\mathbb{V}$ by means of automorphisms acting locally, we will use automorphisms that are minimal under certain restrictions. The corresponding operator is defined as follows:

Definition 5. *Let $\mathbb{V} = (V, \leq)$ be a finite lattice ordered set, and $\mathbb{G} \leq \mathfrak{Aut}\,\mathbb{V}$ an automorphism group of $\mathbb{V}$, while $U \subseteq G$ is one of its subsets. For any $x, y, z \in V$ the elements of the set*

$$\mathfrak{Min}_{x,y\mapsto z}\, U := \mathfrak{Min}_{\sqsubseteq}\{g \in U \mid x^g = x, y^g = z\} \tag{3}$$

are called minimal annotating automorphisms (fixing x and mapping y to z). *The elements of the set*

$$\mathfrak{Min}_{\downarrow x,y\mapsto z}\, U := \begin{cases} \mathfrak{Min}_{x,y\mapsto z}\, U \cap \mathbb{G}_{\downarrow_\leq x} & \mathfrak{Min}_{x,y\mapsto z}\, U \cap \mathbb{G}_{\downarrow_\leq x} \neq \emptyset \\ \mathbb{1} & \mathfrak{Min}_{x,y\mapsto z}\, U \cap \mathbb{G}_{\downarrow_\leq x} = \emptyset \textit{ and} \\ & \mathfrak{Min}_{x,y\mapsto z}\, U \neq \emptyset \\ \emptyset & \textit{else} \end{cases} \tag{4}$$

are called upper minimal automorphisms.

For applications it would be interesting to have an annotation that consists only of minimal acting automorphisms. Unfortunately, that is not generally possible. Nevertheless when $\mathfrak{Irred}^{\vee}_{\leq} : V \to \mathfrak{P}\, V$ maps each element to the set of supremum irreducible elements less or equal to it, we can proof the following lemma:

Lemma 6. *For any finite lattice ordered set $\mathbb{V} = (V, \leq)$, any automorphism group $\mathbb{G} \leq \mathfrak{Aut}(V, \leq)$, and any transversal $T \subseteq V$ of $V \mathbin{\backslash\!\backslash} \mathbb{G}$ there exists a hierarchical annotation that consists of upper minimal automorphisms, if for all elements $x, y \in T$ with $x \prec y$ and $z \in \mathfrak{Irred}^{\vee}_{\leq} y \setminus \mathfrak{Irred}^{\vee}_{\leq} x$ the following condition holds:*

$$(\mathfrak{Irred}^{\vee}_{\leq} y \setminus \mathfrak{Irred}^{\vee}_{\leq} x)^{\mathbb{G}} = z^{\mathbb{G}} \tag{5}$$

Proof. It is a well-known fact, that for each automorphism $g \in G$ and every element $x \in V$ the equation $x^g = \bigvee\big((\mathfrak{Irred}^{\vee}_{\leq} x)^g\big)$.

Let $x \prec' y$ a pair of neighbours in $(T, \leq')$ and $x \prec z$ a pair of neighbours in $(V, \leq)$ such that $z \in y^{\mathbb{G}}$. Then we can modify the proof of Lemma 5 with the following refinements:

1. If $y \in \mathfrak{Irred}^{\vee}_{\leq} y$ then choose for any $z \in y^{\mathbb{G}_{\downarrow_{\leq} x}}$ a minimal automorphism $g_{x,y,z} \in \mathfrak{Min}_{x,y\mapsto z}\, \mathbb{G}$ and define λ_1 as in Lemma 5.
2. For $y \in \mathfrak{Irred}^{\vee}_{\leq} y$ and $z \notin y^{\mathbb{G}_{\downarrow_{\leq} x}}$ there exist an automorphism $g \in \mathbb{G}_x \setminus \mathbb{G}_{\downarrow_{\leq} x}$ and an automorphism $h \in \lambda_1(x, y)$ such that $z = y^{hg}$ where $g \neq (1)$. In that case the action of $\langle g\rangle$ on z depends on the action on $\downarrow_{\leq} x$. As $(\downarrow_{\leq} x)^{\langle g\rangle} \subseteq \downarrow_{\leq} x$ and the action of $\langle g\rangle$ on $\downarrow_{\leq} x$ is defined by the irreducibles also the action on z depends on the irreducibles below it (everything else we have already collected in $\lambda_1(x, y)$. Let $0 = \hat{x}_0 \prec' \hat{x}_1 \prec' \ldots \prec' \hat{x}_l = x$ be a maximal chain from 0 to x of elements of V. Then there exists a chain $x_0 \prec x_1 \prec \ldots \prec x_l$ such that $x_i \in \hat{x}_i^{\mathbb{G}}$. For each pair (x_i, x_{i+1}) we define $I(x_i, x_{i+1}) := \mathfrak{Irred}^{\vee}_{\leq} x_{i+1} \setminus \mathfrak{Irred}^{\vee}_{\leq} x_i$. Let $g_0 = (1)$. Given $\hat{x}_i^{g_i} = x_i$ chose a minimal automorphism m_i from $\lambda(\hat{x}_i, \hat{x}_{i+1})$ that maps $\hat{x}_i$ to $x_{i+1}^{g g_i^{-1}}$. This is always possible as $|I(x_i, x_{i+1})^{\mathbb{G}} \cap T| = 1$. Then define $g_{i+1} := m_i g_i$. Finally we get an automorphism g_l that acts on $\downarrow_{\leq} x$, implying $g_l g^{-1} \in \mathbb{G}_{\downarrow_{\leq} x}$.
3. If $y \notin \mathfrak{Irred}^{\vee}_{\leq} y$ there exist a unique irreducible $\hat{y} \in T$ and an element $\hat{x} \in T$ such that $\hat{y} \in (\mathfrak{Irred}^{\vee}_{\leq} y \setminus \mathfrak{Irred}^{\vee}_{\leq} x)^{\mathbb{G}}$, $\hat{x} \prec' \hat{y}$ and $\hat{x} < x$ hold. Obviously $\hat{y} \not\leq x$. In that case we define $\lambda_1(x, y) := \lambda_1(\hat{x}, \hat{y})$.

Induction over the height (the size of the longest chain) leads to the desired annotation. Obviously we don't need to define any λ_2 to something different than the empty set. Thus we can define $\lambda := \lambda_1$ which fulfils the conditions of Definition 4 and provides a labelling using upper minimal automorphism. □

Definition 6. *Let $\mathbb{V} = (V, \leq)$ a finite lattice and $\mathbb{G} \leq \mathfrak{Aut}\, \mathbb{V}$ a group of automorphisms. Let further $T \subseteq V$ a transversal of the orbit partition $V \mathbin{\backslash\!\backslash} \mathbb{G}$ and $\lambda : T \times T \to \mathfrak{P}\, G$ a minimal acting annotation. Let further $\leq'$ defined by $x \leq' y$ iff there exists an automorphism $g \in G$ such that $x \leq y^g$. Then the triplet $(T, \leq', \lambda)$ is called* minimal acting orbifold of $\mathbb{V}$ by $\mathbb{G}$.

Theorem 1 (Unfolding). *Let $\mathbb{V} = (V, \leq)$ a finite lattice and $\mathbb{G} \leq \mathfrak{Aut}\,\mathbb{V}$ a group of automorphisms and $(T, \leq', \lambda)$ a hierarchical (minimal acting) orbifold of $\mathbb{V}$ by $\mathbb{G}$. Let further for $\mathfrak{C} : T \times T \to \mathfrak{P}\,T$ the mapping that assigns a pair $x \leq' y$ to the set of all contiguous chains of λ from x to y, and to the empty set otherwise (i. e. if $x \not\leq' y$).*

Then the ordered set $(L, \preccurlyeq)$ with the relation $\preccurlyeq \subseteq L \times L$ defined by

$$L := \bigcup \{x^{\Lambda(0,x)} \mid x \in T\}, \qquad \textit{and} \tag{6}$$

$$x \preccurlyeq y :\Leftrightarrow \exists z, \hat{z} \in T, g \in \Lambda(0, \hat{z}) : z^g = x, \hat{z}^g = y, z \leq' \hat{z} \tag{7}$$

equals $(V, \leq)$.

Proof. We prove this theorem by induction. As any finite lattice is also a complete lattice the orbit of the minimal element 0 of $(V, \leq)$ is a singleton. That implies that $0 \in T$.

Let us start with the set $L_0 := \{0\}$ containing the infimum of the lattice and the relation $\preccurlyeq_0 := \{(0, 0)\}$.

Let $y \in V$ and suppose that for any $x < y$ we have already proved that $\downarrow_{\preccurlyeq} x = \downarrow_{\leq} x \subseteq V$. Thus, for each $x \in \{x' \in V \mid x' \prec y\}$ there exists an automorphism $g_x \in G$ such that $x^{g_x} \in T$. W. l. o. g. $g_x \in \Lambda(0, x^{g_x})$ (otherwise there exists $g' \in \Lambda(0, x^{g_x})$ with $x^{g'} = y^{g_x}$). As T is a transversal of V, there is also an automorphism $g_y \in G$ such that $y^{g_y} \in T$. From $x \leq y$ we know $x^{g_x} \leq' y^{g_y}$ and thus, if $g_y \in \lambda(x^{g_x}, y^{g_y}) \cdot \Lambda(0, x^{g_x})$ then also $x \preccurlyeq y$.

Note that for any z the equation $\Lambda(0, z) = \bigcup_{\hat{z} \prec' z} \big(\lambda(\hat{z}, z)\Lambda(0, \hat{z})\big)$ holds. If there exists an automorphism $h \in \lambda(x^{g_x}, y^{g_y})$ such that $y^{g_x} = (y^{g_y})^h$ then the condition $h \cdot g_x^{-1} \in h \cdot \Lambda(0, x^{g_x}) \subseteq \lambda(x^{g_x}, y^{g_y}) \cdot \Lambda(0, x^{g_x})$ holds. Thus, $y \in L$ and $x \preccurlyeq y$. If there is no such element in $\lambda(x^{g_x}, y^{g_y})$, then by Definition 4 we can find an automorphism $h \in \lambda(x, y) \cdot \Lambda(0, x)$ which maps y^{g_y} to y^{g_x}. Thus, $y \in L$ and $x \preccurlyeq y$ hold in this case, too. As we had chosen x arbitrarily below y we have proved $\downarrow_{\leq} y \subseteq \downarrow_{\preccurlyeq} y$.

Since L is constructed by automorphisms of $\mathbb{V}$ which map certain elements of V to other elements of V, we know $L \subseteq V$. Suppose that for any two elements $x, y \in L$ the inequality $x \preccurlyeq y$ holds. Then we know that in equation (7) the condition $\lambda(z, \hat{z}) \cdot \Lambda(0, z) \subseteq \lambda_{full}(z, \hat{z})$ holds if λ_{full} is the full annotation as discussed in [1,2,3,4]. Thus, we know that for any automorphism $g \in \lambda(z, \hat{z}) \cdot \Lambda(0, z)$ the inequality $x = z^g \leq \hat{z}^g = y$ holds. Thus also $x \leq y$.

As we have proved the condition $\downarrow_{\preccurlyeq} y = \downarrow_{\leq} y \subseteq V$, induction proves the equation $(L, \preccurlyeq) = (V, \leq)$ for $y = 1 \in T$. □

6 An Example with Musical Background

In many parts of computational music theory pitches and notes are represented by integers. This has been proved to be useful especially in technical applications. As there are well-documented mathematical models available (see e. g., [6,7,8,9]) and an applied description is available in [10], here only the technically necessary parts are described. Let $\mathbb{Z}$ be considered as tone system. Then each subset $C \subseteq \mathbb{Z}$ can be considered as a chord. In music theory it is not very common to talk about tones. It is more common to talk about scales that consist of chromas. Let $o \in \mathbb{Z}$ be an interval which we will call octave. Two tones which are an octave apart are considered to have the same chroma. The transitive continuation of this procedure leads to a structure of chromas that is isomorphic to $\mathbb{Z}_o$. Each of its subsets is called harmony. For certain applications (e. g. in the software "MUTABOR" [11]) the form of harmonies of incoming streams of music (e. g. a MIDI stream [12]) are of special interest. Two harmonies have the same form if there exists a transposition that transforms one into the other.

Let $H \subseteq \mathbb{Z}_o$ a harmony. Then for some chromatic interval $i \in \mathbb{Z}_o$ the mapping $t_i : \mathfrak{P}\,\mathbb{Z}_o \to \mathfrak{P}\,\mathbb{Z}_o : H \mapsto \{p + i \mid p \in H\}$ is called a transposition. The harmonic form $F(H)$ of some harmony H is defined as the mapping $F : \mathfrak{P}\,\mathbb{Z}_o \to \mathfrak{P}(\mathfrak{P}\,\mathbb{Z}_o) : H \mapsto \{t_i(H) \mid i \in \mathbb{Z}_o\}$.

As for each interval $i \in \mathbb{Z}_o$ there exists a transposition t_i. These transpositions can be considered as automorphisms of the ordered set $(\mathfrak{P}\,\mathbb{Z}_o, \subseteq)$, the transposition group will be denoted by $\mathbb{T}$. In fact this ordered set is a complete lattice which is invariant under transposition. The harmonic forms can be considered as the set of the orbits of the transpositions $\mathfrak{P}\,\mathbb{Z}_o \setminus\!\!\setminus \mathbb{T}$.

If we want to recognise a certain set of harmonies $\mathfrak{H}$ we can build an automaton that can be described by a concept lattice. Let $\mathbb{H} = \mathbb{K}(G, M, I)$ be the context defined by

$$G := \bigcup_{H \in \mathfrak{H}} \mathfrak{P}\,H, \qquad M := \mathbb{Z}_o, \text{ and } \qquad I := \{(H, p) \in G \times M \mid p \in H\}. \tag{8}$$

Then $\mathfrak{B}\mathbb{H}$ can be considered as automaton that recognises all finite words that consist of letters which are included in one of the harmonies of $\mathfrak{H}$. Starting in the concept $(G, \emptyset)$, with each pitch $p \in \mathbb{Z}_o$ the automaton switches state (A, B) to state $\big((B \cup \{p\})^I, B \cup \{p\}\big)$. The latter is a state as with every Harmony H the set of objects G includes each of its subsets $H' \subseteq H$. If such a state doesn't exist the automaton won't recognise the word.

The naive approach to recognise harmonic forms uses the same idea. Let $\mathfrak{F} := \{F(H) \mid H \in \mathfrak{H}\}$ a set of harmonic forms. Then we define the lattice as follows: $\mathbb{F} = \mathbb{K}(G', M', I')$ with

$$G := \{t_i(H), H \in \mathfrak{H}, i \in \mathbb{Z}_o\}, \quad M := \mathbb{Z}_o, \text{ and } \quad I := \{(H, p) \in G \times M \mid p \in H\}. \tag{9}$$

The concept lattice $\mathfrak{B}(\mathbb{F})$ has all transpositions as automorphisms. Figure 6 shows a concept lattice that can be used to recognise the major seventh chord $F(\{0, 4, 7, 10\})$, the minor triad $F(\{0, 3, 7\})$ and all of their harmonic subforms. The nodes are arranged orbit-wise. That means, each cluster is an orbit of $\mathfrak{B}(\mathbb{F}) \setminus\!\!\setminus \mathbb{T}$. Thus, the automorphisms can be seen as cyclic permutations of the endpoints of the edges. In comparison with the number of orbits the lattice is large: 14 orbits are formed by 140 concepts.

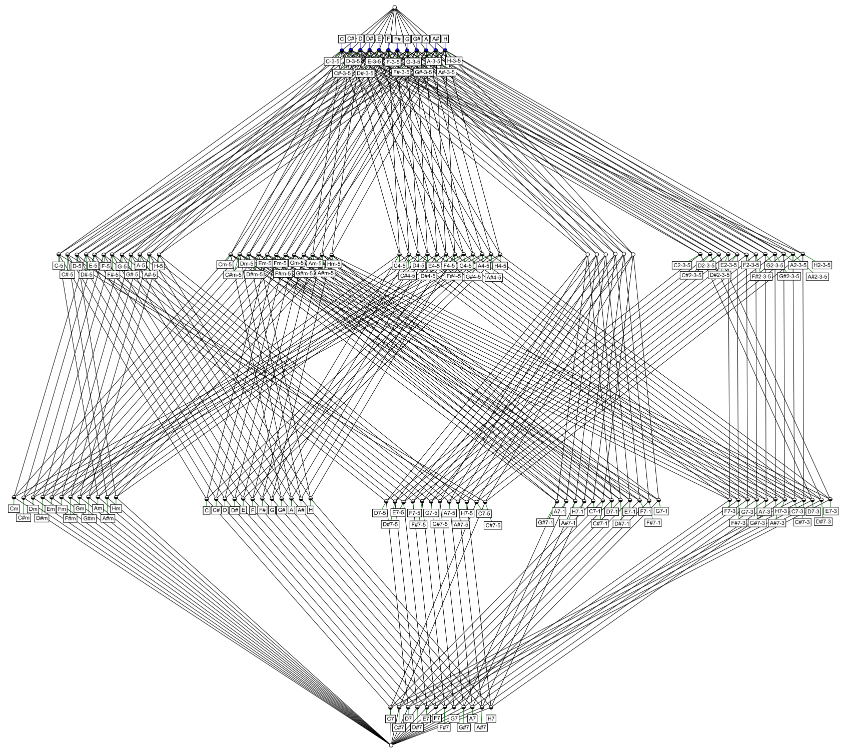


Figure 2. Concept lattice describing patterns to be matched

For an automaton that recognises harmonic forms it would be interesting to compress the data as the generation of the lattice can be very time and space consuming if the chroma system contains more chromas. E. g. considering the pitch bend parameter as part of a pitch in standard MIDI environments the number of pitches increases from 12 to $12 \cdot 2^{14}$. In such a case an orbifold based representation of the lattice does not necessarily increase in size. Starting by a hierarchical annotation of minimal acting automorphisms we can enhance the annotation by replacing each automorphism by a pair consisting of the automorphism and the character (pitch) that triggers its action. To avoid unnecessary operations the automaton could save the automorphism that must be applied to the pattern rather than applying it. In many cases (e. g., classification) it does not need to be applied at all.

This approach provides two additional advantages: As we know the context automorphisms, we can use a folded context to compute the order relation of the concept orbifolds as described in [4]. On the other hand changing the size of the chroma system can be done in several ways. The orbifold based approach provides a promising base for analysing such operations in order to provide fast algorithms that can be used in real time.

7 Outlook

We have seen that orbifolds of certain lattices can be described using hierarchical annotations, and that it is possible to minimise the action of the annotating automorphisms without losing the possibility of unfolding such hierarchical orbifolds.

Nevertheless there are open topics that can improve the theory. In Lemma 6 Restriction (5) has technical reasons. At the moment it is an open question how to deal with arbitrary lattices. It might be helpful to use systems of generators for the annotation λ. That should be straight forward if care is taken on conjugated subgroups.

Another easy extension would be to elaborate the idea for arbitrary ordered sets.

References

1. Zickwolff, M.: Darstellung symmetrischer Strukturen durch Transversale. Contributions to General Algebra **7** (1991) 391–403
2. Zickwolff, M.: Rule Exploration: First Order Logic in Formal Concept Analysis. Dissertation, Technische Hochschule Darmstadt (1991)
3. Borchmann, D., Ganter, B.: Concept Lattice Orbifolds – First Steps. In: Formal Concept Analysis. Volume 5548 of Lecture Notes in Computer Science., Springer (2009) 22–37
4. Borchmann, D.: Context Orbifolds. Diplomarbeit, Technische Universität Dresden (2009)
5. Hall, P.: A note on soluble groups. Journal of the London Mathematical Society **1**(2) (1928) 98–105
6. Neumaier, W., Wille, R.: Extensionale Standardsprache in der Musiktheorie – eine Schnittstelle zwischen Musik und Informatik. In Hesse, H.P., ed.: Mikrotöne III: Bericht über das 3. internationale Symposium „Mikrotonforschung, Musik mit Mikrotönen, Ekmelische Musik“, 28.-30. April 1989 in Salzburg. Volume 6 of Veröffentlichungen der Gesellschaft für Ekmelische Musik., Innsbruck, Ed. Helbling (1990) 149–167
7. Neumaier, W.: Was ist ein Tonsystem? : Eine historisch-systematische Theorie der abendländischen Tonsysteme, gegründet auf die antiken Theoretiker Aristoxenos, Eukleides und Ptolemaios, dargestellt mit Mitteln der modernen Algebra. Volume 9 of Quellen und Studien zur Musikgeschichte von der Antike bis in die Gegenwart. Lang, Frankfurt am Main (1986)
8. Wille, R.: Mathematische Sprache in der Musiktheorie. In: Jahrbuch Überblicke Mathematik 1980. Bibliographisches Institut, Mannheim (1980) 167–184
9. Winkler, J.T.: Algebraische Modellierung von Tonsystemen. Beiträge zur begrifflichen Wissensverarbeitung. Verl. Allg. Wiss. – HRW e.K., Mühltal (2009)
10. Schlemmer, T., Schmidt, S.: A formal concept analysis of harmonic forms and interval structures. Annals of Mathematics and Artificial Intelligence **59**(2) (2010) 241–256 10.1007/s10472-010-9198-6.
11. Mutabor team: Mutabor – the dynamic tempered piano. Website (2012) URL: `http://www.math.tu-dresden.de/~mutabor/` (Archived by WebCite® at `http://www.webcitation.org/6ASACEUxB`) Accessed: 2012-09-05.
12. MIDI Manufacturers Association: The complete midi 1.0 detailed sepecification (1996)

Fuzzy formal concept analysis via multilattices: first prospects and results*

J. Medina[1] and J. Ruiz-Calviño[2]

[1] Department of Mathematics. University of Cádiz
Email: jesus.medina@uca.es
[2] Department of Mathematics. University of Córdoba
Email: jrcalvino@uco.es

Abstract. The most general algebraic structure of truth-values considered in the theory of fuzzy concept analysis to evaluate the attributes and objects has been a lattice. However, in some examples arises the necessity of a more general structure. In this paper we investigate the use of multilattices as underlying set of truth-values for these attributes and objects.

1 Introduction

The study of reasoning methods to work out with uncertainty, imprecise data or incomplete information has been a trending topic in the recent years in order to explain, in a better way, observed facts, specify statements, reasoning and/or execute programs.

One important and powerful mathematical tool that has been used for this purpose at theoretical level is fuzzy logic. From the applicative side, neural networks have a massively parallel architecture-based dynamics which are inspired by the structure of human brain, adaptation capabilities, and fault tolerance. The recent paradigm of soft computing promotes the use and integration of different approaches for the problem solving.

Formal concept analysis, introduced by Wille in [23], is a useful tool for qualitative data analysis and has become an appealing major research topic, from both the theoretical and applied perspectives. What we pretend in this paper is to present the multilattices as a basis on the area of formal concept analysis and, specifically, what will lead us to what we have called fuzzy formal concept multilattice.

There has been many approaches in order to generalise the classical concept lattices given by Ganter and Wille [12] allowing some uncertainty in data. The first *fuzzy* approach was proposed by Burusco and Fuentes-González [4] where fuzzy concept lattices were first presented, and later further developed by Pollandt [22] and Bělohlávek [1]. Other approaches emerge trying to work with non-commutative fuzzy logic and similarity as we can see in the work of

* Partially supported by the Spanish Science Ministry projects TIN2009-14562-C05-01 and TIN2009-14562-C05-03, and by Junta de Andalucía project P09-FQM-5233.

Georgescu and Popescu [13]. This approach, consisting in generalizing the equality relation and considering an alternative similarity relation, underlies in the works of Bělohlávek [2], which considered L-equalities to extend the fuzzy concept lattice. This approach was extended in an asymmetric way, although only for the case of classical equality ($L = \{0, 1\}$) by Krajči, who introduced the so-called generalized concepts lattices in [16,15].

Recently, a new approach has been proposed by Medina et al in [20,19] who introduced the multi-adjoint concept lattices, joining the multi-adjoint philosophy with concept lattices. To do this the authors needed to generalize the adjoint pairs into what they called adjoint triples [6]. This new structure directly generalizes almost all the approaches previously cited.

On the other hand, the theory of multilattices arose trying to weaken the restrictions imposed on a (complete) lattice, namely, the "existence of least upper bounds and greatest lower bounds" is relaxed to "existence of *minimal* upper bounds and *maximal* lower bounds". In this direction, several definitions have been proposed in the mathematical literature of the structure so-called multilattice [3,14].

Later on, an alternative notion of multilattice, with better properties regarding substructures than the previous definitions, has been introduced [5,17]. Moreover, this structure has proved to be an important tool in order to obtain some advances in the theory of mechanized deduction in temporal logics.

Multilattices, in the sense of the paragraph above, also arise in a natural manner in the research area concerning fuzzy extensions of logic programming [18]. For instance, one of the hypotheses of the main termination result for sorted multi-adjoint logic programs [7] has been weakened only when the underlying set of truth-values is a multilattice [8]; as far as we know, the question of providing a counter-example on a lattice remains open.

The main result introduced here is the presentation of multilattices as underlying set where to evaluate the attributes, the objects and the relation in a fuzzy environment to formal concept analysis, indeed, this leads us to see that the set of "multilattice concepts" is a multilattice. We can see as well that if we evaluate the objects or the attributes in a lattice and the other in a multilattice what we have again a lattice not a multilattice as we could think at first.

The idea of using multilattices as underlying set of truth values arises us since in real life many thigns are ordered in a way that we we know that some objects are better than others but sometimes we can not choose which is the best of them because they can have different properties. This idea can be seen in the example we give in the last section of this paper where we consider a group of hotels. It is logical to think that four-star hotels are better than three star hotel, but sometimes we can not decide if a four-star hotel is better than another four-star hotel since these can have different properties to be considered like acommodation, location, price, etc., and one can be better in one property and worse in other. This, as we have already said, is what have lead up to consider multilattices because this structure deals better with objects which are uncomparable.

The plan of this paper is the following: in Section 2 we present the main definitions and results to understand the paper. Section 3 presents the formal concept multilattice; we also provide an example where this new structure can be used in Section 4; the paper ends with some conclusions and prospects for future work.

2 Preliminaries

In this first section we will set the basic notions required to the complete understanding of the paper. We will start with a bit of lattice and multilattice theory and finished with concept lattice analysis.

One of the most structures used when hanging with fuzziness is a lattice, this has been a very suitable in order to develop many theories. Although the following definitions are well known, we will recall them here in order to make this paper as selfcontained as possible. The definition of a lattice is given bellow.

Definition 1. *By a* complete lattice *is understood a poset,* $(L, \preceq)$*, where every subset of* L *has supremum and infimum.*

When instead of the existence of both supremum and infimum for every subset we only ask for the existence of one of them the notion of semilattice arised.

Definition 2. *By a* complete lower semilattice *is understood a poset,* $(L, \preceq)$*, where every non-empty subset of* L *has infimum.*

Definition 3. *By a* complete upper semilattice *is a poset,* $(L, \preceq)$*, where every non-empty subset of* L *has supremum.*

Nevertheless, there is a closed relationship between lattices and semilattices, indeed, if we have the existence of a top element in lower semilattices or a bottom element in upper semilattices we have that they become lattices as the following theorem states.

Theorem 1. *A complete upper (lower) semilattice* $(L, \preceq)$ *with a minimum element (maximum element) is a complete lattice.*

Once we have reminded the notions of lattice and semilattice we will pass to define what a multilattice is. To get this we will start with some preliminaries notions.

Definition 4. *Let* $(P, \leq)$ *be a poset and* $K \subseteq P$*, we say that:*

- K *is called a* chain *if for every two elements* $x, y \in K$ *we have that* $x \leq y$ *or* $y \leq x$*.*
- K *is called an* antichain *if none of its elements are comparable, i.e., for every* $x, y \in K$ *we have that* $x \nleq y$ *and* $y \nleq x$*.*

Definition 5. *A poset $(P, \leq)$ is called* coherent *if every chain has supremum and infimum.*

Once we have introduced these notions we can give the definition of a complete multilattice.

Definition 6. *A* complete multilattice *is a coherent poset without infinite antichains, $(M, \leq)$, where for each subset, the set of its upper (lower) bounds has minimal (maximal) elements.*

Each minimal(maximal) element of the upper (lower) bounds of a subset is called *multisupremum(multinfimum)*. The set of all multisuprema(multinfima) will be denoted by multisup(multinf).

Example 1. An example of a multilattice is given in figure 1.

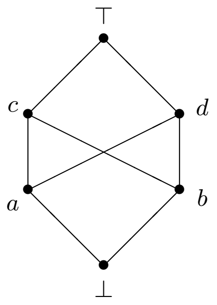


Fig. 1. Multilattice M6

In this multilattice we have that if we consider the subset $\{a, b\}$ we have that multinf$\{a, b\} = \bot$ and multisup$\{a, b\} = \{c, d\}$ while if we consider as subset $\{c, d\}$ we have that multinf$\{c, d\} = \{a, b\}$ and multisup$\{a, b\} = \top$

We will remind now the notion of adjoint pair we will use later [11,21].

Definition 7. *Let $(P_1, \leq_1)$, $(P_2, \leq_2)$, $(P_3, \leq_3)$ be posets and $\&\colon P_1 \times P_2 \to P_3$, $\leftarrow\colon P_3 \times P_2 \to P_1$, be mappings, then $(\&, \leftarrow)$, is called an* adjoint pair *with respect to P_1, P_2, P_3 if:*

- *$\&$ is increasing in both arguments;*
- *$\leftarrow$ are increasing in the first argument and decreasing in the second;*
- *$x \leq_1 z \leftarrow y$ iff $x \,\&\, y \leq_3 z$ for all $x \in P_1$, $y \in P_2$ and $z \in P_3$;*

Now, we will pass to introduce a bit of fuzzy concept analysis. We will remind first the notions of Galois connection and concept [9,10].

Definition 8. *Let $^{\downarrow}\colon P \to Q$ and $^{\uparrow}\colon Q \to P$ be two maps between the posets $(P, \leq)$ and $(Q, \leq)$. The pair $(^{\uparrow}, ^{\downarrow})$ is called a* Galois connection *if:*

- *$p_1 \leq p_2$ implies ${p_2}^{\downarrow} \leq {p_1}^{\downarrow}$ for every $p_1, p_2 \in P$;*
- *$q_1 \leq q_2$ implies ${q_2}^{\uparrow} \leq {q_1}^{\uparrow}$ for every $q_1, q_2 \in Q$;*

– *$p \leq p^{\uparrow\downarrow}$ and $q \leq q^{\downarrow\uparrow}$ for all $p \in P$ and $q \in Q$;*

An interesting property of a Galois connection $(^{\uparrow}, ^{\downarrow})$ is that $^{\downarrow} = {}^{\downarrow\uparrow\downarrow}$ and $^{\uparrow} = {}^{\uparrow\downarrow\uparrow}$.

Definition 9. *A pair (p, q) is called a* concept *if $p^{\downarrow} = q$ and $q^{\uparrow} = p$.*

If P and Q are lattices we have as well the following result:

Theorem 2. *[9] Let $(L_1, \preceq_1)$ and $(L_2, \preceq_2)$ be two complete lattices and $(^{\uparrow}, ^{\downarrow})$ a Galois connection between them, then we have that the set $\mathcal{C} = \{(x, y) \mid x \in L_1, y \in L_2, x^{\downarrow} = y, y^{\uparrow} = x\}$ is a complete lattice with the following ordering $(x_1, y_1) \preceq (x_2, y_2)$ if and only if $x_1 \preceq_1 x_2$ (or equivalently $y_2 \preceq_2 y_1$), where the supremum and the infimum are given bellow:*

$$\bigwedge_{i\in I}(x_i, y_i) = \left(\bigwedge_{i\in I} x_i, (\bigvee_{i\in I} y_i)^{\uparrow\downarrow}\right)$$
$$\bigvee_{i\in I}(x_i, y_i) = \left((\bigvee_{i\in I} x_i)^{\downarrow\uparrow}, \bigwedge_{i\in I} y_i\right)$$

With all this notions now we can pass to the following section where we will present concept multilattices.

3 Fuzzy formal concept multilattices

When working in concept analysis theory we always have two sets A and B representing the attributes and the objects together with a relation between them. In order to reach the concept multilattices these ones will be evaluated in complete multilattices.

The first result we obtain concerning concept analysis in multilattices will be crucial for our purpose. We will denote by M_1^A and M_2^B the sets of all mappings from A to M_1 and from B to M_2 respectively.

Theorem 3. *Let $(M_1, \leq_1)$ and $(M_2, \leq_2)$ be two complete multilattices, A and B two sets and $(^{\uparrow}, ^{\downarrow})$ a Galois connection between M_1^A and M_2^B. If $\{(g_i, f_i)\}_{i\in I}$ is a set of concepts we have that*

$$\textit{multinf}\{{f_i}^{\downarrow} \mid i \in I\} \subseteq (\textit{multisup}\{f_i \mid i \in I\})^{\downarrow} \qquad (1)$$

$$\textit{multinf}\{{g_i}^{\uparrow} \mid i \in I\} \subseteq (\textit{multisup}\{g_i \mid i \in I\})^{\uparrow} \qquad (2)$$

where $(\textit{multisup}\{f_i \mid i \in I\})^{\downarrow} = \{f_{\textit{mult}}^{\downarrow} \mid f_{\textit{mult}} \in \textit{multisup}\{f_i \mid i \in I\}\}$ and $(\textit{multisup}\{g_i \mid i \in I\})^{\uparrow}$ is given similarly.

Proof. We will prove (1). Item (2) is proved in a similar way.

Let $g \in \text{multinf}\{f_i{}^{\downarrow} \mid i \in I\}$ we have that $g \leq_1 f_i{}^{\downarrow}$ for every $i \in I$. As $^{\uparrow}$ is decreasing we have that $f_i^{\downarrow\uparrow} \leq_2 g^{\uparrow}$, but the pair $(^{\uparrow}, {}^{\downarrow})$ is a Galois connection so we have that:

$$f_i \leq_2 (f_i)^{\downarrow\uparrow} \leq_2 g^{\uparrow}$$

Then there is $f_{\text{mult}} \in \text{multisup}\{f_i \mid i \in I\}$ such that $f_{\text{mult}} \leq_2 g^{\uparrow}$. As $^{\downarrow}$ is decreasing we have that $g^{\uparrow\downarrow} \leq_1 f^{\downarrow}_{\text{mult}}$ using again that $(^{\uparrow}, {}^{\downarrow})$ is a Galois connection we obtain that:

$$g \leq_1 g^{\uparrow\downarrow} \leq_1 f^{\downarrow}_{\text{mult}} \tag{3}$$

On the other hand, we have that $f_{\text{mult}} \in \text{multisup}\{f_i \mid i \in I\}$, so $f_i \leq_2 f_{\text{mult}}$ for every $i \in I$ then, as $^{\downarrow}$ is decreasing $f^{\downarrow}_{\text{mult}} \leq_1 f_i^{\downarrow}$, for every $i \in I$, then $f^{\downarrow}_{\text{mult}}$ is a lower bound of the set $\{f_i{}^{\downarrow} \mid i \in I\}$, but $g \in \text{multinf}\{f_i{}^{\downarrow} \mid i \in I\}$ and $g \leq_1 f^{\downarrow}_{\text{mult}}$, by (3). Therefore, by maximality of g we have that $g = f^{\downarrow}_{\text{mult}}$.

Thus, we have proved that for every $g \in \text{multinf}\{f_i{}^{\downarrow} \mid i \in I\}$ there is $f_{\text{mult}} \in \text{multisup}\{f_i \mid i \in I\}$ such that $g = f^{\downarrow}_{\text{mult}}$, which leads us to the result. $\square$

We cannot get always the equality in this theorem as we can see in the next example:

Example 2. If we consider the multilattice of Fig. 1 and the folowing Galois connection, $^{\uparrow} = {}^{\downarrow}\colon M6 \to M6$ defined as:

$$\bot^{\uparrow} = \top \ ; \ a^{\uparrow} = b^{\uparrow} = c^{\uparrow} = c \ ; \ d^{\uparrow} = \bot \ ; \ \top^{\uparrow} = \bot$$

It is routine to prove that the pair $(^{\uparrow}, {}^{\downarrow})$ is a Galois connection.

On one hand, we obtain that

$$\text{multinf}\{a^{\uparrow}, b^{\uparrow}\} = \text{multinf}\{c\} = c$$

However, on the other hand:

$$(\text{multisup}\{a, b\})^{\uparrow} = (\{c, d\})^{\uparrow} = \{c^{\uparrow}, d^{\uparrow}\} = \{c, \bot\}$$

which proves that we cannot get the equality always.

As a consequence of the previous theorem, we have that, given the set of all concepts $\mathcal{C} = \{(g, f) \mid f \in M_1^A, g \in M_2^B, g^{\uparrow} = f, f^{\downarrow} = g\}$, and the ordering defined as $(g_1, f_1) \leq (g_2, f_2)$ if and only if $g_1 \leq_1 g_2$ (if and only if $f_2 \leq_2 f_1$), then $(\mathcal{C}, \leq)$ is a complete multilattice which is a result similar to Theorem 2, but now with respect to multilattices.

Theorem 4. *If $(M_1, \leq_1)$ and $(M_2, \leq_2)$ be two complete multilattices, A and B two sets and $(^{\uparrow}, {}^{\downarrow})$ a Galois connection between M_1^A and M_2^B, then we have that $(\mathcal{C}, \leq)$ is a complete multilattice where for every set of concepts $\{(g_i, f_i)\}_{i \in I}$:*

$$\mathit{multinf}\{(g_i, f_i)\} = (\mathit{multinf}\{g_i\}, (\mathit{multinf}\{g_i\})^{\uparrow}) \tag{4}$$

$$\mathit{multisup}\{(g_i, f_i)\} = ((\mathit{multinf}\{f_i\})^{\downarrow}, \mathit{multinf}\{f_i\}) \tag{5}$$

Proof. If we prove that they are concepts, then it is obvious that they are the multisuprema and the multinfima due to the definition of the ordering in $\mathcal{C}$.

By Theorem 3, we have that

$$\text{multinf}\{g_i\} \subseteq (\text{multisup}\{f_i\})^{\downarrow} \tag{6}$$

Hence, given $g \in \text{multinf}\{g_i\}$, there exist $f \in \text{multisup}\{f_i\}$, such that $g = f^{\downarrow}$. Therefore, since $(^{\uparrow}, ^{\downarrow})$ is a Galois connection, $g^{\uparrow\downarrow} = f^{\downarrow\uparrow\downarrow} = f^{\downarrow} = g$.

Consequently, it is trivial that they are concepts. For the multisuprema the proof is similar. The proof of coherence and the non-existence of anti-chains comes directly from the definition of the ordering consider. $\square$

At this point we could think what would happen whether the set of objects or the set of attributes are evaluated in a lattice while the other in a multilattice. The answer to this is given by the following corollary.

Proposition 1. *Considering the framework of the previous theorem, if M_1 or M_2 is a lattice, then we have that $\mathcal{C}$ is a lattice.*

Proof. If M_1 is a lattice in the first equality of (4) the multinfimum becomes a singleton so it is indeed an infimum. Hence, every set has an infimum and so $(\mathcal{C}, \leq)$ is a complete lower semilattice. Therefore, we only have to prove that there is a maximum element $\top_{\mathcal{C}}$ in $\mathcal{C}$.

Let $g_\top \in M_1^A$ the map which sends every element of A to the maximum element $\top$ of M_1 and consider the pair $(g_\top, g_\top^{\uparrow})$. If we prove that it is a concept then we have finished, since it is obvious that this element would be the maximum element in $\mathcal{C}$.

We only have to prove that $g_\top = g_\top^{\uparrow\downarrow}$. As $(^{\uparrow}, ^{\downarrow})$ a Galois connection we have that $g_\top \leq_1 g_\top^{\uparrow\downarrow}$ and, as $g_\top(a) = \top$ for every element $a \in A$, we have that the equality holds, i.e., $g_\top = g_\top^{\uparrow\downarrow}$.

The proof for M_2 being a lattice is similar. $\square$

The following section introduces a simple and particular context where we can get a Galois connection from an adjoint pair what allows us to obtain a concept multilattice.

4 A worked out example

The multilattice considered for the calculation in this example is the one given in Fig. 2 together with the following adjoint pair $(\&, \leftarrow)$.

$$x \,\&\, y = \begin{cases} x & \text{if } y = \top \\ y & \text{if } x = \top \\ \bot & \text{if } x \in \{\bot, b\} \text{ or } y \in \{\bot, b\} \\ a & \text{otherwise} \end{cases}$$

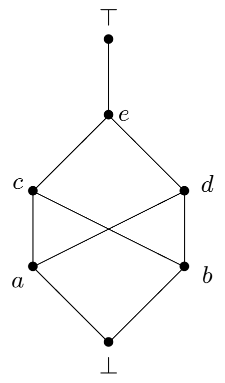


Fig. 2. Multirretículo M6*

$$z \leftarrow y = \begin{cases} \top & \text{if } y \leq z \\ z & \text{if } y = \top \\ b & \text{if } y \notin \{\bot, b, \top\} \text{ and } z \in \{\bot, b\} \\ e & \text{otherwise} \end{cases}$$

It is routine calculation that $(\&, \leftarrow)$ is, indeed, an adjoint pair, in which $\&$ is commutative.

Imagine that we are going to travel to a city and we have to decide which hotel is the best for us. In this example, in order to no complicate the calculation we will taking into account seven different hotels, as objects, and two attributes, which will be *price* and *situation*. Hence, we have as set of objects $B = \{H1, H2, H3, H4, H5, H6, H7\}$ and as set of attributes $A = \{\text{price}, \text{situation}\}$, both evaluated in $M6^*$ and the $M6$-fuzzy relation, $R\colon A \times B \to M6^*$, between them, defined in Table 1

Table 1. Relation R

R	price	situation
H1	d	$\bot$
H2	c	a
H3	$\top$	b
H4	a	d
H5	b	e
H6	a	b
H7	d	c

Evaluating the hotels in a multilattice comes from the idea that the hotels are ordered thinking of the number of stars they have. We can state, for example that any four-star hotel is better than any three-star hotel, but if both hotels are four-star ones we cannot distinguish between them at the beginning.

In the case of the situation, we have that we can say one situations are better than other but we cannot compare a situations that are for example one kilometer from the downtown but in different directions.

In the case of prizes, happens more or less the same because we cannot distinguish between prizes which are very similar.

If we see the relationship we have that $R(H5, \text{price}) = b$, $R(H6, \text{price}) = a$ means that the fifth and the sixth hotels have more or less the same prices but we cannot decide which is best taking into account only their prizes.

For these reasons we have chosen multilattices for their evaluation.

We are trying to choose a hotel to stay in according to our preferences in prizes and situation.

It is easy to check that for the adjoint pair $(\&, \leftarrow)$ and for any mapping $f\colon A \to M6^*$ or $g\colon B \to M6^*$ the following sets has infimum.

$$\{R(a,b) \leftarrow g(b) \mid b \in B\}$$
$$\{R(a,b) \leftarrow f(a) \mid a \in A\}$$

Hence, we can define the next Galois connection

$$g^{\uparrow}(a) = \inf\{R(a,b) \leftarrow g(b) \mid b \in B\}$$
$$f^{\downarrow}(b) = \inf\{R(a,b) \leftarrow f(a) \mid a \in A\}$$

The proof of $(\downarrow, \uparrow)$ being a Galois connection follows directly from the existence of the infimum of these sets, that $\&$ is commutative and that the implication are decreasing in the second argument.

Therefore, from Theorem 4 we have an fuzzy concept multilattice and if our preferences are the following $g(\text{price}) = a$ and $g(\text{situation}) = d$ we have that for $H1$.

$$g^{\uparrow}(H1) = \inf\{d \leftarrow a, \bot \leftarrow d\} = \inf\{\top, b\} = b$$

And for the others:

$$g^{\uparrow}(H2) = e \ , \ g^{\uparrow}(H3) = b \ , \ g^{\uparrow}(H4) = \top \ , \ g^{\uparrow}(H5) = b \ , \ g^{\uparrow}(H6) = b \ , \ g^{\uparrow}(H7) = e$$

On the other hand we have that

$$\begin{aligned} f^{\downarrow\uparrow}(\text{price}) &= \inf\{d \leftarrow b, c \leftarrow e, \top \leftarrow b \ , a \leftarrow \top \, b \leftarrow b, a \leftarrow b, a \leftarrow e\} \\ &= \inf\{\top, \top, a, \top, e, e\} = a \end{aligned}$$

In a similar way we obtain that

$$f^{\downarrow\uparrow}(\text{situation}) = d$$

Thus, according to out preference stablished by f, we have that our best choice is $H4$, although $H2$ and $H7$ are really good ones too.

5 Conclusions and future work

A first approach to fuzzy formal concept multilattices has been presented. This paradigm arises as a more flexible setting than formal concept analysis framework, as the introduced motivating example shows.

Moreover, several properties have been proved. For example, we have checked that the concepts in the new framework form a complete multilattice and that if we impose that one of the set of attributes or objects are evaluated in a lattice and the other in a multilattice, then we obtain a complete lattice.

In the future, we will study general Galois connections which allows us to get more concept multilattices. We will focus as well, when we have these Galois connections, on getting a representation theorem for them to be able to get which multilattices are isomorphic to concept multilattices.

References

1. R. Bělohlávek. Lattice generated by binary fuzzy relations (extended abstract). In *4th Intl Conf on Fuzzy Sets Theory and Applications*, page 11, 1998.
2. R. Bělohlávek. Concept lattices and order in fuzzy logic. *Annals of Pure and Applied Logic*, 128:277–298, 2004.
3. M. Benado. Les ensembles partiellement ordonnés et le théorème de raffinement de Schreier, II. Théorie des multistructures. *Czechoslovak Mathematical Journal*, 5(80):308–344, 1955.
4. A. Burusco and R. Fuentes-González. The study of L-fuzzy concept lattice. *Mathware & Soft Computing*, 3:209–218, 1994.
5. P. Cordero, G. Gutiérrez, J. Martínez, and I. P. de Guzmán. A new algebraic tool for automatic theorem provers. *Annals of Mathematics and Artificial Intelligence*, 42(4):369–398, 2004.
6. M. Cornejo, J. Medina, and E. Ramírez. A comparative study of adjoint triples. *Fuzzy Sets and Systems*, 2012.
7. C. Damásio, J. Medina, and M. Ojeda-Aciego. Sorted multi-adjoint logic programs: Termination results and aplications. In *Logics in Artificial Intelligence, JELIA'04*, pages 252–265. Lecture Notes in Artificial Intelligence, 3229, 2004.
8. C. Damásio, J. Medina, and M. Ojeda-Aciego. Termination of logic programs with imperfect information: applications and query procedure. *Journal of Applied Logic*, 5:435–458, 2007.
9. B. Davey and H. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, second edition, 2002.
10. E. Diday and R. Emilion. Maximal and stochastic galois lattices. *Discrete Applied Mathematics*, 127(2):271–284, 2003.
11. R. P. Dilworth and M. Ward. Residuated lattices. *Transactions of the American Mathematical Society*, 45:335–354, 1939.
12. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundation*. Springer Verlag, 1999.
13. G. Georgescu and A. Popescu. Concept lattices and similarity in non-commutative fuzzy logic. *Fundamenta Informaticae*, 53(1):23–54, 2002.
14. D. Hansen. An axiomatic characterization of multilattices. *Discrete Mathematics*, 1:99–101, 1981.

15. S. Krajči. The basic theorem on generalized concept lattice. In V. Snásel and R. Bělohlávek, editors, *International Workshop on Concept Lattices and their Applications, CLA 2004*, pages 25–33, 2004.
16. S. Krajči. A generalized concept lattice. *Logic Journal of IGPL*, 13(5):543–550, 2005.
17. J. Martínez, G. Gutiérrez, I. de Guzmán, and P. Cordero. Generalizations of lattices looking at computation. *Discrete Mathematics*, 295:107–141, 2005.
18. J. Medina, M. Ojeda-Aciego, and J. R.-C. no. Fuzzy logic programming via multilattices. *Fuzzy Sets and Systems*, 158:674–688, 2007.
19. J. Medina, M. Ojeda-Aciego, and J. Ruiz-Calviño. Relating generalized concept lattices with concept lattices for non-commutative conjunctors. *Applied Mathematics Letters*, 21(12):1296–1300, 2008.
20. J. Medina, M. Ojeda-Aciego, and J. Ruiz-Calviño. Formal concept analysis via multi-adjoint concept lattices. *Fuzzy Sets and Systems*, 160(2):130–144, 2009.
21. J. Medina, M. Ojeda-Aciego, and P. Vojtáš. Similarity-based unification: a multi-adjoint approach. *Fuzzy Sets and Systems*, 146:43–62, 2004.
22. S. Pollandt. *Fuzzy Begriffe*. Springer, Berlin, 1997.
23. R. Wille. Restructuring lattice theory: an approach based on hierarchies of concepts. In I. Rival, editor, *Ordered Sets*, pages 445–470. Reidel, 1982.

An interpretation of the L-Fuzzy Concept Analysis as a tool for the Morphological Image and Signal Processing

Cristina Alcalde[1], Ana Burusco[2], and Ramón Fuentes-González[2]

[1] Dpt. Matemática Aplicada. Escuela Univ. Politécnica
Univ. del País Vasco. Plaza de Europa, 1
20018 - San Sebastián (Spain)
c.alcalde@ehu.es
[2] Dpt. Automática y Computación. Univ. Pública de Navarra
Campus de Arrosadía
31006 - Pamplona (Spain)
{burusco,rfuentes}@unavarra.es

Abstract. In this work we are going to set up a new relationship between the L-fuzzy Concept Analysis and the Fuzzy Mathematical Morphology. Specifically we prove that the problem of finding fuzzy images or signals that remain invariant under a fuzzy morphological opening or under a fuzzy morphological closing, is equal to the problem of finding the L-fuzzy concepts of some L-fuzzy context. Moreover, since the Formal Concept Analysis and the Mathematical Morphology are the particular cases of the fuzzy ones, the showed result has also an interpretation for binary images or signals.

Keywords: L-fuzzy Concept Analysis, Fuzzy Mathematical Morphology, Morphological Image Processing

1 Introduction

The L-fuzzy Concept Analysis and the Fuzzy Mathematical Morphology were developed in different contexts but both use the lattice theory as algebraic framework.

In the case of the L-fuzzy Concept Analysis, we define the L-fuzzy concepts using a fuzzy implication and a composition operator associated with it. In the Fuzzy Mathematical Morphology, a fuzzy implication is also used to define the erosion but a t-norm also appears to introduce the dilation.

On the other hand, both theories have been used in knowledge extraction processes in data bases [14–16].

Next, we will show a brief description of them.

2 Antecedents

2.1 *L*-fuzzy Concept Analysis

The Formal Concept Analysis of R. Wille [28, 17] extracts information from a binary table that represents a formal context (X, Y, R) with X and Y finite sets of objects and attributes respectively and $R \subseteq X \times Y$. The hidden information is obtained by means of the formal concepts that are pairs (A, B) with $A \subseteq X$, $B \subseteq Y$ verifying $A^* = B$ and $B^* = A$, where $*$ is the derivation operator that associates the attributes related to the elements of A to every object set A, and the objects related to the attributes of B to every attribute set B. These formal concepts can be interpreted as a group of objects A that shares the attributes of B.

In previous works [11, 12] we have defined the L-fuzzy contexts (L, X, Y, R), with L a complete lattice, X and Y sets of objects and attributes respectively and $R \in L^{X \times Y}$ a fuzzy relation between the objects and the attributes. This is an extension of the Wille's formal contexts to the fuzzy case when we want to study the relationship between the objects and the attributes with values in a complete lattice L, instead of binary values.

In our case, to work with these L-fuzzy contexts, we have defined the derivation operators 1 and 2 given by means of these expressions:

$$\forall A \in L^X, \forall B \in L^Y \quad \begin{aligned} A_1(y) &= \inf_{x \in X}\{I(A(x), R(x, y))\} \\ B_2(x) &= \inf_{y \in Y}\{I(B(y), R(x, y))\} \end{aligned}$$

with I a fuzzy implication operator defined in the lattice $(L, \leq)$ and where A_1 represents the attributes related to the objects of A in a fuzzy way, and B_2, the objects related to all the attributes of B.

In this work, we are going to use the following notation for these derivation operators to stand out their dependence to relation R:

$\forall A \in L^X$, $\forall B \in L^Y$, we define $\mathcal{D}_R : L^X \to L^Y$, $\mathcal{D}_{R^{op}} : L^Y \to L^X$

$$\begin{aligned} \mathcal{D}_R(A)(y) &= A_1(y) = \inf_{x \in X}\{I(A(x), R(x, y))\} \\ \mathcal{D}_{R^{op}}(B)(x) &= B_2(x) = \inf_{y \in Y}\{I(B(y), R^{op}(y, x))\} \end{aligned}$$

where we denote by R^{op} the opposite relation of R, that is, $\forall (x, y) \in X \times Y$, $R^{op}(y, x) = R(x, y)$.

The information stored in the context is visualized by means of the L-fuzzy concepts that are some pairs $(A, A_1) \in (L^X, L^Y)$ with $A \in \text{fix}(\varphi)$, set of fixed points of the operator φ, being defined from the derivation operators 1 and 2 as $\varphi(A) = (A_1)_2 = A_{12}$. These pairs, whose first and second components are said to be the fuzzy extension and intension respectively, represent a set of objects that share a set of attributes in a fuzzy way.

The set $\mathcal{L} = \{(A, A_1)/A \in \text{fix}(\varphi)\}$ with the order relation $\leq$ defined as:

$$\forall (A, A_1), (C, C_1) \in \mathcal{L}, \quad (A, A_1) \leq (C, C_1) \text{ if } A \leq C(\text{ or } A_1 \geq C_1)$$

is a complete lattice that is said to be the L-fuzzy concept lattice [11, 12].

On the other hand, given $A \in L^X$, (or $B \in L^Y$) we can obtain the associated L-fuzzy concept. In the case of using a residuated implication, as we do in this work, the associated L-fuzzy concept is (A_{12}, A_1) (or (B_2, B_{21})).

Other important results about this theory are in [1, 10, 25, 13, 24, 5].

A very interesting particular case of L-fuzzy contexts appears trying to analyze situations where the objects and the attribute sets are coincident [2, 3], that is, L-fuzzy contexts (L, X, X, R) with $R \in L^{X \times X}$, (this relation can be reflexive, symmetrical ...). In these situations, the L-fuzzy concepts are pairs (A, B) such that $A, B \in L^X$.

These are the L-fuzzy contexts that we are going to use to obtain the main results of this work. Specifically, we are going to take a complete chain $(L, \leq)$ as the valuation set, and L-fuzzy contexts as $(L, \mathbb{R}^n, \mathbb{R}^n, R)$ or $(L, \mathbb{Z}^n, \mathbb{Z}^n, R)$. In the first case, the L-fuzzy concepts (A, B) are interpreted as signal or image pairs related by means of R. In the second case, A and B are digital versions of these signals or images.

2.2 Mathematical Morphology

The Mathematical Morphology is a theory concerned with the processing and analysis of images or signals using filters and operators that modify them. The fundamentals of this theory (initiated by G. Matheron [22, 23] and J. Serra [26]), are in the set theory, the integral geometry and the lattice algebra. Actually this methodology is used in general contexts related to activities as the information extraction in digital images, the noise elimination or the pattern recognition.

Mathematical Morphology in binary images and grey levels images In this theory images A from $X = \mathbb{R}^n$ or $X = \mathbb{Z}^n$ (digital images or signals when n=1) are analyzed.

The *morphological filters* are defined as operators $F : \wp(X) \to \wp(X)$ that transform, symplify, clean or extract relevant information from these images $A \subseteq X$, information that is encapsulated by the filtered image $F(A) \subseteq X$.

These morphological filters are obtained by means of two basic operators, the *dilation* δ_S and the *erosion* ε_S, that are defined in the case of binary images with the *sum* and *difference of Minkowski* [26], respectively.

$$\delta_S(A) = A \oplus S = \bigcup_{s \in S} A_s \quad \varepsilon_S(A) = A \ominus \breve{S} = \bigcap_{s \in \breve{S}} A_s$$

where A is an image that is treated with another $S \subseteq X$, that is said to be *structuring element*, or with its opposite $\breve{S} = \{-x/x \in S\}$ and where A_s represents a translation of A: $A_s = \{a + s/a \in A\}$.

The structuring image S represents the effect that we want to produce over the initial image A.

These operators are not independent since they are dual transformations with respect to the complementation [27], that is, if A^c represents the complementary set of A, then:

$$\varepsilon_S(A) = (\delta_S(A^c))^c, \forall A, S \in \wp(X)$$

We can compose these operators dilation and erosion associated with the structuring element S and obtain the basic filters *morphological opening* $\gamma_S : \wp(X) \to \wp(X)$ and *morphological closing* $\phi_S : \wp(X) \to \wp(X)$ defined by:

$$\gamma_S = \delta_S \circ \varepsilon_S \quad \phi_S = \varepsilon_S \circ \delta_S$$

The opening γ_S and the closing ϕ_S over these binary images verifies the two conditions that characterize the morphological filters: They are isotone and idempotent operators, and moreover it is verified, $\forall A, S \in \wp(X)$:

a) $\gamma_S(A) \subseteq A \subseteq \phi_S(A)$

b) $\gamma_S(A) = (\phi_S(A^c))^c$

These operators will characterize some special images (the S-open and the S-closed ones) that will play an important role in this work.

This theory is generalized introducing some tools to treat images with grey levels [26]. The images and the structuring elements are now maps defined in $X = \mathbb{R}^n$ and with values in $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$ or defined in $X = \mathbb{Z}^n$ and with values in finite chains as, for instance, $\{0, 1, \ldots, 255\}$.

The previous definitions can be immersed in a more general framework that considers each image as a point $x \in L$ of a partially ordered structure $(L, \leq)$ (complete lattice), and the filters as operators $F : L \to L$ with properties related to the order in these lattices [26, 19].

Now, the erosions $\varepsilon : L \to L$ are operators that preserve the infimum $\varepsilon(\inf M) = \inf \varepsilon(M), \forall M \subseteq L$ and the dilations $\delta : L \to L$, the supremum: $\delta(\sup M) = \sup \delta(M), \forall M \subseteq L$. The opening $\gamma : L \longrightarrow L$ and the closing $\phi : L \longrightarrow L$ are isotone and idempotent operators verifying $\gamma(A) \leq A \leq \phi(A)$.

Fuzzy Mathematical Morphology In this new framework and associated with lattices, a new *fuzzy morphological image processing* has been developed [6, 7, 4, 8, 9, 21, 20] using L-fuzzy sets A and S (with $X = \mathbb{R}^2$ or $X = \mathbb{Z}^2$) as images and structuring elements.

In this interpretation, the filters are operators $F_S : L^X \to L^X$, where L is the chain $[0, 1]$ or a finite chain $L_n = \{0 = \alpha_1, \alpha_2, ..., \alpha_{k-1}, 1\}$ with $0 < \alpha_1 < ... < \alpha_{k-1} < 1$.

In all these cases, *fuzzy morphological dilations* $\delta_S : L^X \to L^X$ and *fuzzy morphological erosions* $\varepsilon_S : L^X \to L^X$ are defined using some operators of the fuzzy logic [4, 6, 9, 21].

In general, there are two types of relevant operators in the Fuzzy Mathematical Morphology. One of them is formed by those obtained by using some pairs $(*, I)$ of adjunct operators related by:

$$(\alpha * \beta \leq \psi) \Longleftrightarrow (\beta \leq I(\alpha, \psi))$$

The other type are the morphological operators obtained by pairs $(*, I)$ related by a strong negation $' : L \to L$:

$$\alpha * \beta = (I(\alpha, \beta'))', \forall(\alpha, \beta) \in L \times L$$

An example of one of these pairs that belongs to both types is the formed by the t-norm and the implication of Lukasiewicz.

In this paper, we work taking as $(X, +)$ the commutative group $(\mathbb{R}^n, +)$ or the commutative group $(\mathbb{Z}^n, +)$, and as $(L, \leq, ', I, *)$, the complete chain $L = [0, 1]$ or a finite chain as $L = L_n = \{0 = \alpha_1, \alpha_2, ..., \alpha_{k-1}, 1\}$ with the Zadeh negation and $(*, I)$ the Lukasiewicz t-norm and implication.

We interpret the L-fuzzy sets $A : X \to L$ and $S : X \to L$ as n-dimensional images in the space $X = \mathbb{R}^n$ (or n-dimensional digital images in the case of $X = \mathbb{Z}^n$).

In the literature, (see [4, 6, 18, 21]), erosion and dilation operators are introduced associated with the residuated pair $(*, I)$ as follows:

If $S : X \to L$ is an image that we take as *structuring element*, then we consider the following definitions associated with (L, X, S)

Definition 1. *[6] The fuzzy erosion of the image $A \in L^X$ by the structuring element S is the L-fuzzy set $\varepsilon_S(A) \in L^X$ defined as:*

$$\varepsilon_S(A)(x) = \inf\{I(S(y - x), A(y))/y \in X\} \qquad \forall x \in X$$

The fuzzy dilation of the image A by the structuring element S is the L-fuzzy set $\delta_S(A)$ defined as:

$$\delta_S(A)(x) = \sup\{S(x - y) * A(y)/y \in X\} \qquad \forall x \in X$$

Then we obtain fuzzy erosion and dilation operators $\varepsilon_S, \delta_S : L^X \to L^X$. Moreover, it is verified:

Proposition 1. *(1) If $\leq$ represents now the usual order in L^X obtained by the order extension in the chain L, then the pair $(\varepsilon_S, \delta_S)$ is an adjunction in the lattice $(L^X, \leq)$, that is:*

$$\delta_S(A_1) \leq A_2 \Longleftrightarrow A_1 \leq \varepsilon_S(A_2)$$

(2) If A' is the negation of A defined by $A'(x) = (A(x))', \forall x \in X$ and if $\breve{S}$ represents the image associated with S such that $\breve{S}(x) = S(-x), \forall x \in X$, then it is verified:

$\varepsilon_S(A') = (\delta_{\breve{S}}(A))', \quad \delta_S(A') = (\varepsilon_{\breve{S}}(A))', \qquad \forall A, S \in L^X$

Proof. (1) Suppose that $\delta_S(A_1) \leq A_2$. Then $\delta_S(A_1)(x) \leq A_2(x)\ \forall x \in X$. That is, $S(x-y) * A_1(y) \leq A_2(x)\ \forall (x,y) \in X \times X$.

From these inequalities and from the equivalence $\alpha * \beta \leq \gamma \Leftrightarrow \beta \leq I(\alpha, \gamma)$:

$$A_1(y) \leq I(S(x-y), A_2(x)), \forall (x,y) \in X \times X$$

and interchanging x and y, we have:

$$A_1(x) \leq I(S(y-x), A_2(y)), \forall (x,y) \in X \times X$$

and consequently

$$A_1(x) \leq \inf\{I(S(y-x), A_2(y))/y \in X\}, \forall x \in X$$

That is: $A_1(x) \leq \varepsilon_S(A_2)(x), \forall x \in X$ that shows that $A_1 \leq \varepsilon_S(A_2)$.

We can prove the other implication in a similar way.

(2) Let be $x \in X$.

$$\begin{aligned}\varepsilon_S(A')(x) &= \inf\{I(S(y-x), A'(y))/y \in X\} = \inf\{I(\check{S}(x-y), A'(y))/y \in X\} \\ &= \inf\{(\check{S}(x-y) * A(y))'/y \in X\} = (\sup\{(\check{S}(x-y) * (A(y))/y \in X\})' \\ &= (\delta_{\check{S}}(A)(x))' = (\delta_{\check{S}}(A))'(x)\end{aligned}$$

The second equality is proved analogously. □

3 Relation between both theories

The erosion and dilation operators given in Definition 1 are used to construct the basic morphological filters: the opening and the closing (see [4, 6, 18, 21]).

Definition 2. *The fuzzy opening of the image $A \in L^X$ by the structuring element $S \in L^X$ is the fuzzy subset $\gamma_S(A)$ that results from the composition of the erosion $\varepsilon_S(A)$ of A by S followed by its dilation:*

$$\gamma_S(A) = \delta_S(\varepsilon_S(A)) = (\delta_S \circ \varepsilon_S)(A)$$

The fuzzy closing of the image $A \in L^X$ by the structuring element $S \in L^X$ is the fuzzy subset $\phi_S(A)$ that results from the composition of the dilation $\delta_S(A)$ of A by S followed by its erosion:

$$\phi_S(A) = \varepsilon_S(\delta_S(A)) = (\varepsilon_S \circ \delta_S)(A)$$

It can be proved that the operators γ_S and ϕ_S are morphological filters, that is, they preserve the order and they are idempotent:

$$A_1 \leq A_2 \Longrightarrow (\gamma_S(A_1) \leq \gamma_S(A_2)) \text{ and } (\phi_S(A_1) \leq \phi_S(A_2))$$

$$\gamma_S(\gamma_S(A)) = \gamma_S(A), \phi_S(\phi_S(A)) = \phi_S(A), \forall A \in L^X, \forall S \in L^X$$

Moreover, these filters verify that:

$$\gamma_S(A) \leq A \leq \phi_S(A) \quad \forall A \in L^X, \forall S \in L^X$$

Analogous results that those obtained for the erosion and dilation operators can be proved for the opening and closing:

Proposition 2. *If A' is the negation of A defined by $A'(x) = (A(x))' \quad \forall x \in X$, then:*

$$\gamma_S(A') = (\phi_{\breve{S}}(A))', \quad \phi_S(A') = (\gamma_{\breve{S}}(A))' \qquad \forall A, S \in L^X$$

Proof. $\gamma_S(A') = \delta_S(\varepsilon_S(A')) = \delta_S((\delta_{\breve{S}}(A))') = (\varepsilon_{\breve{S}}(\delta_{\breve{S}}(A)))' = (\phi_{\breve{S}}(A))'$.
The other equality can be proved in an analogous way. □

Since the operators γ_S and ϕ_S are increasing in the complete lattice $(L^X, \leq)$, by Tarski's theorem, the respective fixed points sets are not empty. These fixed points will be used in the following definition:

Definition 3. *An image $A \in L^X$ is said to be S-open if $\gamma_S(A) = A$ and it is said to be S-closed if $\phi_S(A) = A$.*

These S-open and S-closed sets provide a connection between the Fuzzy Mathematical Morphology and the Fuzzy Concept Theory, as we will see next.

For that purpose, given the complete chain L that we are using, and a commutative group $(X, +)$, we will associate with any fuzzy image $S \in L^X$, the fuzzy relation $R_S \in L^{X \times X}$ such that:

$$R_S(x, y) = S(x - y), \forall (x, y) \in X \times X$$

It is evident that $R_{S'} = R_S^{'}$ and, if R_S^{op} represents the opposite relation of R_S, then $R_S^{op} = R_{\breve{S}}$.

In agreement with this last point, we can redefine the erosion and dilation as follows:

$$\begin{aligned}
\varepsilon_S(A)(x) &= \inf\{I(R_S(y, x), A(y))/y \in X\} \\
&= \inf\{I(R_S^{op}(x, y), A(y))/y \in X\}, \forall x \in X \\
\delta_S(A)(x) &= \sup\{R_S(x, y) * A(y)/y \in X\}, \forall x \in X
\end{aligned}$$

With this rewriting, given the structuring element $S \in L^X$, we can interpret the triple (L, X, S) as an L-fuzzy context (L, X, X, R'_S) where the sets of objects and attributes are coincident. The incidence relation $R'_S \in L^{X \times X}$ is at the same time the negation of an interpretation of the fuzzy image by the structuring element S.

We will use this representation as L-fuzzy context to prove the most important results that connect both theories:

Theorem 1. *Let (L, X, S) be the triple associated with the structuring element $S \in L^X$. Let $(L, X, X, R_S^{'})$ be the L-fuzzy context whose incidence relation $R_S^{'} \in L^{X\times X}$ is the negation of the relation R_S associated with S. Then the operators erosion ε_S and dilation δ_S en (L, X, S) are related to the derivation operators $\mathcal{D}_{R'_S}$ and $\mathcal{D}_{R'^{op}_S}$ in the L-Fuzzy context $(L, X, X, R_S^{'})$ by:*

$$\varepsilon_S(A) = \mathcal{D}_{R'_S}(A') \quad \forall A \in L^X$$
$$\delta_S(A) = (\mathcal{D}_{R'^{op}_S}(A))' \quad \forall A \in L^X$$

Proof. Taking into account the properties of the Lukasiewicz implication, for any $x \in X$, it is verified that:

$\varepsilon_S(A)(x) = \inf\{I(R_S(y, x), A(y))/y \in X\} = \inf\{I(A'(y), R_S^{'}(y, x))/y \in X\} = \mathcal{D}_{R'_S}(A')(x)$

Analogously,

$\delta_S(A)(x) = \sup\{R_S(x, y) * A(y)/y \in X\} = \sup\{(I(R_S(x, y), A'(y)))'/y \in X\} =$
$(\inf\{I(R_S(x, y), A'(y))/y \in X\})' = (\inf\{I(A(y), R_S^{'}(x, y))/y \in X\})' =$
$(\inf\{I(A(y), R_S^{'op}(y, x))/y \in X\})' = ((\mathcal{D}_{R'^{op}_S}(A))(x))' = (\mathcal{D}_{R'^{op}_S}(A))'(x)$. □

As a consequence, we obtain the following result which proves the connection between the outstanding morphological elements and the L-fuzzy concepts:

Theorem 2. *Let be $S \in L^X$ and let be $R_S \in L^{X\times X}$ its associated relation. The following propositions are equivalent:*

1. *The pair $(A, B) \in L^X \times L^X$ is an L-fuzzy concept of the context $(L, X, X, R_S^{'})$, where $R_S^{'}(x, y) = S'(x - y)$ $\forall (x, y) \in X \times X$.*
2. *The pair $(A, B) \in L^X \times L^X$ is such that the negation A' of A is S-open ($\gamma_S(A') = A'$) and B is the S-erosion of A' (that is, $B = \varepsilon_S(A')$).*
3. *The pair $(A, B) \in L^X \times L^X$ is such that B is S-closed ($\phi_S(B) = B$) and A is the negation of the S-dilation of B (that is, $A = (\delta_S(B))'$).*

Proof.

1 $\Longrightarrow$ 2) Let be $S \in L^X$ and $R_S \in L^{X\times X}$ its associated relation. Let us consider an L-fuzzy concept (A, B) of the L-fuzzy context $(L, X, X, R_S^{'})$ in which $R_S^{'}$ is the negation of R_S. Then, it is verified that $B = \mathcal{D}_{R'_S}(A)$ and $A = \mathcal{D}_{R'^{op}_S}(B)$, and, by the previous proposition, $\varepsilon_S(A') = \mathcal{D}_{R'_S}(A) = B$. Moreover, it is fulfilled that $\gamma_S(A') = \delta_S(\varepsilon_S(A')) = \delta_S(B) = (\mathcal{D}_{R'^{op}_S}(B))' = A'$ which proves that A' is S-open.

2 $\Longrightarrow$ 3) Let us suppose that the hypothesis of *2* are fulfilled. Then, $\phi_S(B) = \varepsilon_S(\delta_S(B)) = \varepsilon_S(\delta_S(\varepsilon_S(A'))) = \varepsilon_S(\gamma_S(A')) = \varepsilon_S(A') = B$, which proves that B is S-closed. On the other hand, from the hypothesis $B = \varepsilon_S(A')$ can be deduced that $\delta_S(B) = \delta_S(\varepsilon_S(A')) = \gamma_S(A')$, and consequently, taking into account that A' is S-open, that $\delta_S(B) = A'$, and finally, $A = (\delta_S(B))'$.

3 $\Longrightarrow$ 1) Let (A, B) be a pair fulfilling the hypothesis of *3*. Let us consider the L-fuzzy context $(L, X, X, R_S^{'})$. Then, by the previous theorem we can deduce that $(\mathcal{D}_{R_S^{'op}}(B)) = (\delta_S(B))' = A$. On the other hand, applying the previous theorem and the hypothesis, $\mathcal{D}_{R_S'}(A) = \varepsilon_S(A') = \varepsilon_S(\delta_S(B)) = \phi_S(B) = B$, which finishes the proof. □

Let us see now some examples.

Example 1. Interpretation of some binary images as formal concepts.

In the referential set $X = \mathbb{R}^2$, if $\overline{\boldsymbol{x}} = (x_1, x_2) \in \mathbb{R}^2$, w is a positive number and if S is the structuring binary image

$$S = \{(x_1, x_2) \in \mathbb{R}^2 / x_1^2 + x_2^2 \leq w^2\}$$

then the associated incidence relation $R_S^c \subset \mathbb{R}^2 \times \mathbb{R}^2$ is such that:

$$\overline{\boldsymbol{x}} R_S^c \overline{\boldsymbol{y}} \Longleftrightarrow ((x_1 - y_1)^2 + (x_2 - y_2)^2 > w^2)$$

which is irreflexive and transitive. The pair (A, B) showed in Figure 1 is a concept of the context $(\mathbb{R}^2, \mathbb{R}^2, R_S^c)$, because $\gamma_S(A^c) = A^c$ and $B = \varepsilon_S(A^c)$.

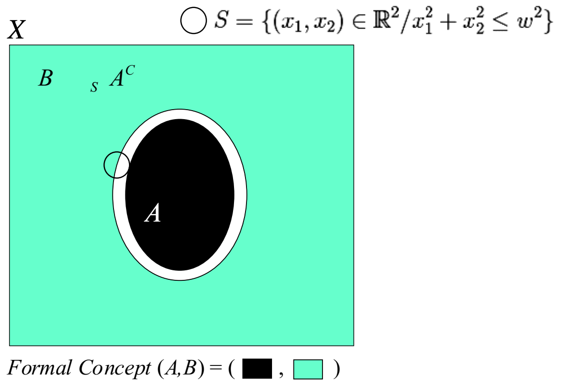

$S = \{(x_1, x_2) \in \mathbb{R}^2 / x_1^2 + x_2^2 \leq w^2\}$
X
B
s
A^C
A
Formal Concept $(A,B) = ($ ■ , 🟩 $)$

Fig. 1. A formal concept of the context $(\mathbb{R}^2, \mathbb{R}^2, R_S^c)$.

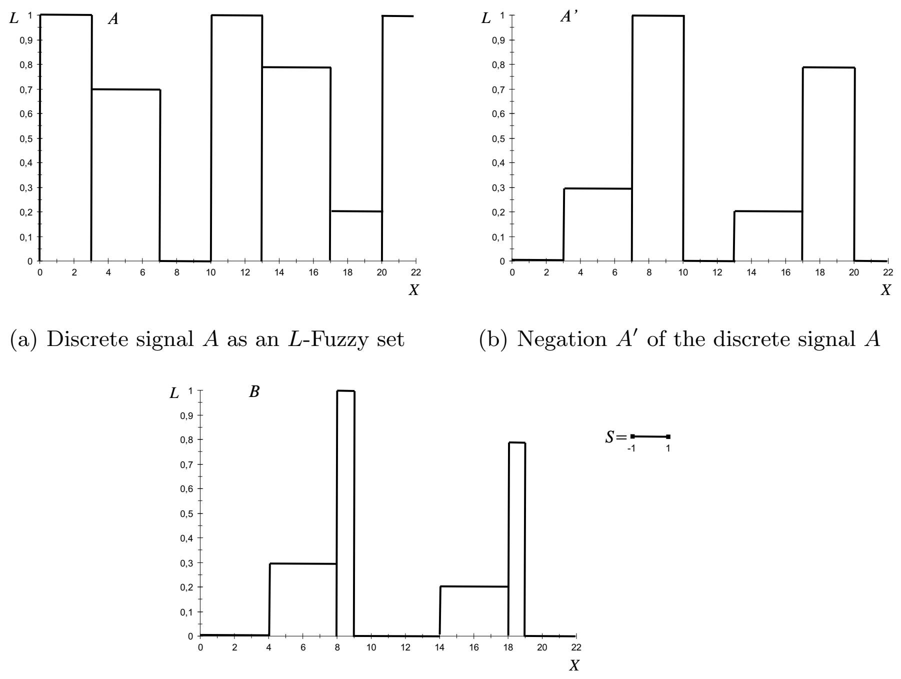
Example 2. Interpretation of some open digital signals as fuzzy concepts.

It is known that the erosion $\varepsilon_S(A)$ of an image A by a binary structuring element S can be rewritten in terms of infimum of the traslations of A by elements of S [27]:

$$\varepsilon_S(A) = \bigwedge_{s \in S} A_{-s} \quad \text{where } A_k(x) = A(x - k)$$

If $X \subseteq \mathbb{Z}$ and $L = \{0, 0.1, 0.2, ..., 0.9, 1\}$ then, the maps $A : X \to L$ can be interpret as 1-D discrete signals. In Figure 2 there are some examples of discrete signals.

The signal in Fig 2(c) is the erosion of A' in Fig 2(b), using a line segment of three pixels as a structuring element, the middle pixel being its origin (S is a crisp set).



(a) Discrete signal A as an L-Fuzzy set (b) Negation A' of the discrete signal A

(c) Discrete signal $B = \varepsilon_S(A')$

Fig. 2. Discrete signals

Here, we can also find the erosions in terms of intersections of image traslations: $\varepsilon_S(A') = \bigwedge\{A'_{-1}, A'_0, A'_{+1}\}$.

It can be proved that A' verifies $\gamma_S(A') = A'$. So, with A in Fig 2 the pair (A, B) with $B = \varepsilon_S(A')$ is a fuzzy concept of the context $(L, \mathbb{Z}, \mathbb{Z}, R^c_S)$ with the crisp incidence relation $xR^c_Sy \Leftrightarrow (x - y) \notin S$, (that is, $|x - y| > 1$).

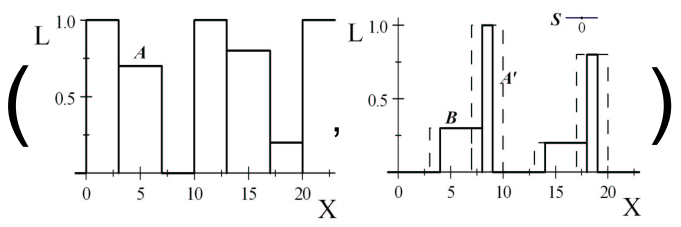


Fig. 3. L-Fuzzy concept (A, B) of the L-Fuzzy context $(L, \mathbb{Z}, \mathbb{Z}, R_S^c)$

4 Conclusions and Future work

The main results of this work show an interesting relation between the L-fuzzy Concept Analysis and the Fuzzy Mathematical Morphology that we want to develop in future works. So, we can apply the algorithms for the calculus of L-fuzzy concepts in Fuzzy Mathematical Morphology and vice versa.

On the other hand, we are extending these results to other type of operators as other implications, t-norms, conjunctive uninorms etc... and to some L-fuzzy contexts where the objects and the attributes are not related to signal or images.

Acknowledgements

This work has been partially supported by the Research Group “Intelligent Systems and Energy (SI+E)” of the Basque Government, under Grant IT519-10, and by the Research Group “Adquisición de conocimiento y minería de datos, funciones especiales y métodos numéricos avanzados” of the Public University of Navarra.

References

1. Alcalde, C., Burusco, A., Fuentes-González, R., Zubia, I.: Treatment of L-fuzzy contexts with absent values. Information Sciences. 179 (1-2), 1–15 (2009)
2. Alcalde, C., Burusco, A., Fuentes-González, R.: Contextos con conjuntos de objetos y atributos iguales. In: Proc. of ESTYLF08, pp. 85–89. Mieres - Langreo (2008)
3. Alcalde, C., Burusco, A., Fuentes-González, R.: Analysis of some L-fuzzy relational equations and the study of its solutions by means of the L-fuzzy Concept Theory. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 20 (1), 21–40 (2012)
4. De Baets, B.: Fuzzy morphology: a logical approach. In: Ayyub, B., Gupta, M. (eds.) Uncertainty Analysis, Engineering and Science: Fuzzy Logic, Statistics and neural network Approach, pp. 53–67, Kluwer Academic Publishers (1997)
5. Belohlavek, R., Konecny, J.: Concept lattices of isotone vs. antitone Galois connections in graded setting: Mutual reducibility revisited. Information Sciences. 199, 133–137 (2012)

6. Bloch, I., Maître, H.: Fuzzy Mathematical Morphologies: a comparative study. Télécom Paris 94D001 (1994)
7. Bloch, I.: Duality vs.adjunction for fuzzy mathematical morphology and general form of fuzzy erosions and dilations. Fuzzy Sets and Systems. 160, 1858–1867 (2009)
8. Burillo, P., Fuentes-González, R., Frago, N.: Inclusion grade and fuzzy implication operators. Fuzzy Sets and Systems. 114, 417–429 (2000)
9. Burillo, P., Frago, N., Fuentes-González, R.: Generation of Fuzzy Mathematical Morphologies. Mathware & Soft Computing. VIII (1), 31–46 (2001)
10. Burusco, A., Fuentes-González, R.: Concept lattices defined from implication operators. Fuzzy Sets and Systems. 114, 431–436 (2000)
11. Burusco, A., Fuentes-González, R.: The Study of the L-fuzzy Concept Lattice. Mathware and Soft Computing. I (3), 209–218 (1994)
12. Burusco, A., Fuentes-González, R.: Construction of the L-fuzzy Concept Lattice. Fuzzy Sets and Systems. 97 (1), 109–114 (1998)
13. Djouadi, Y., Prade, H.: Interval-Valued Fuzzy Galois Connections: Algebraic Requirements and Concept Lattice Construction. Fundamenta Informaticae. 99 (2), 169–186 (2010)
14. Frago, N., Fuentes-González, R.: Descubrimiento de Conocimiento en Bases de Datos Utilizando Técnicas de Morfología Matemática Borrosa. Información tecnológica. 18 (6), 39–50 (2007)
15. Frago, N., Fuentes-González, R.: Procesos de descubrimiento de conocimiento en bases de datos usando grafos asociados a filtros morfológicos. In: Proc. of ESTYLF08, pp. 85–89. Mieres - Langreo (2008)
16. Fuentes-González, R.: The incorporation of the Mathematical Morphology, the Formal Concept Analysis and the Fuzzy Logic techniques and tools to the data cleaning, aggregating, reducing and mining stages of the Knowledge Discovery process. In: Troya, J.M., Ossowski, S. (eds.) Proceedings of CEDI, pp. 147–154. Granada (2005)
17. Ganter, B., Wille, R.: Formal concept analysis: Mathematical foundations. Springer, Berlin - New York (1999)
18. Goutsias, J., Heijmans, H.J.A.M.: Fundamenta Morphologicae Mathematicae. Fundamenta Informaticae. 41, 1–31 (2000)
19. Heijmans, H.J.A.M.: Morphological Image Operators. Academic Press Inc. (1994)
20. Maragos, P.: Lattice Image Precessing: A Unification of Morphological and Fuzzy Algebric Systems. Journal of Mathematical Imaging and Vision. 22, 333–353 (2005)
21. Mas, M., Monserrat, M., Torrens, J.: S-implications and R-implications on a finite chain. Kybernetika. 40 (1), 3–20 (2004)
22. Matheron, G.: Eléments pour une théorie des milieux poreux. Masson, Paris (1967)
23. Matheron, G.: Random Sets and Integral Geometry. Wiley, New York (1975)
24. Medina, J.: Multi-adjoint property-oriented concept lattices. Information Sciences. 190, 95–106 (2012)
25. Medina, J., Ojeda-Aciego, M.: Multi-adjoint t-concept lattices. Information Sciences. 180 (5), 712–725 (2010)
26. Serra, J.: Image Analysis and Mathematical Morphology. Academic Press. I (fourth printing 1990) and II (second printing 1992)
27. Soille, P.: Morphological Image Analysis. Principles and Applications. Second Edition. Springer (2004)
28. Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Rival, I. (ed.), Ordered Sets, pp. 445–470. Reidel, Dordrecht (1982)

Relationship between Two FCA Approaches on Heterogeneous Formal Contexts[⋆]

Ľubomír Antoni, Stanislav Krajči[⋆⋆], Ondrej Krídlo,
Bohuslav Macek, Lenka Pisková

Institute of Computer Science, Faculty of Science,
Pavol Jozef Šafárik University in Košice, Jesenná 5, 040 01 Košice, Slovakia.
`lubomir.antoni@student.upjs.sk`, `stanislav.krajci@upjs.sk`,
`ondrej.kridlo@upjs.sk`, `bohus.macek@gmail.com`,
`lenka.piskova@student.upjs.sk`

Abstract. We show a relationship between two theoretical approaches of Formal Concept Analysis working with so-called heterogeneous formal context i.e. such context in which each object and attribute can have own data-type. One of them is presented in [19]; each value in a formal context is some Galois connection between the lattices corresponding to the appropriate object and attribute. Another approach is presented in our paper [1] and it is a unifying platform of approaches from [14] and [11], [12]. In this paper, we prove that each of them can be derived from another.

Keywords: Formal Concept Analysis, Galois connection, G-ideal

1 Introduction

The Formal Concept Analysis is a well-known data-mining method on a rectangle matrix of data where each row corresponds to some object, each column corresponds to some attribute and a matrix field value expresses the presence of the column attribute to the row object. One of the goals of this method is to find so-called concepts – the stable (in some sense) pairs of subsets of objects and attributes. This method can be considered as a nice application of the algebraic notion of a Galois connection. The Formal Concept Analysis is based and deeply described in the classical Ganter & Wille's book [9] where authors concentrate mainly to the so-called crisp case with binary data in the matrix. The natural question arose: What if the matrix data have a non-binary character?

[⋆] This work was partially supported by the grant VEGA 1/0832/12, by the Slovak Research and Development Agency under contract APVV-0035-10 "Algorithms, Automata, and Discrete Data Structures" by the Agency of the Slovak Ministry of Education for the Structural Funds of the EU, under project ITMS:26220120007.

[⋆⋆] Corresponding author at: Institute of Computer Science, Faculty of Science, Pavol Jozef Šafárik University in Košice, Jesenná 5, 040 01 Košice, Slovakia, `stanislav.krajci@upjs.sk`.

Beside the conceptual scaling from [9] which returns concepts with crisp subsets in both coordinates, some other answers arose which return concepts with fuzzy subsets at least in one coordinate: The first one was done by Burusco & Fuentes-Gonzalez [8] and it was improved (independently) by Bělohlávek [2], [3] and Pollandt [21], [22] which use values from the same residual lattice for values of the matrix and for the fuzziness of subsets of the objects and the attributes. Another approach independently (and with slight differences) given by Ben Yahia & Jaoua [7], Bělohlávek, Sklenář, & Zacpal [4], and Krajči [10] was not so symmetric – it considers fuzzy subsets in one coordinate and crisp (binary) subsets in another one. All these approaches where covered by a common platform – so-called generalized concept lattices [12], [13] which diversifies fuzziness of subsets of the attributes, fuzziness of subsets of the objects and moreover fuzziness of the matrix values.

Then Medina and Ojeda-Aciego brought the idea of multi-adjointness used in logic-programming [16], [17], [18] to the Formal Concept Analysis too [14], [15]. Because of this novelty and originality, this approach is not (at least immediately) covered by the above-mentioned generalized concept lattices.

This fact has inspired us to modify our old approach in such a way that this will work with different mutual relationships between the objects and the attributes. Moreover we work with different lattices for different rows and columns and for the matrix data. To compare with the till known approaches which works with attributes and objects of the same type, an important advantage of this new, totally diversifying, approach is the possibility to apply the Formal Concept Analysis to heterogeneous data too. This is the reason why we will call this new approach heterogeneous. We have described this approach in [1] and recall it in Section 2.

Another answer to the problem of data heterogeneity was given by [19] and [20]. In this approach, each datum in a formal context are not a simple number or other singular value but (sic!) a Galois connection which describes in a some way the behavior between the corresponding object and attribute. We recall this approach in Section 3.

2 Heterogeneous formal context

In this section we recall the basic definitions and results from [1].

Let A and B be non-empty sets. Let $\mathcal{P} = ((P_{a,b}, \leq_{P_{a,b}}) : a \in A, b \in B)$ be a system of posets and let R be a function from $A \times B$ such that $R(a,b) \in P_{a,b}$, for all $a \in A$ and $b \in B$. Let $\mathcal{C} = ((C_a, \leq_{C_a}) : a \in A)$ and $\mathcal{D} = ((D_b, \leq_{D_b}) : b \in B)$ be systems of complete lattices. (For simplicity, we will omit the indices of all noticed $\leq_?$, it will be always clear which of one is used.)

Let $\odot = ((\bullet_{a,b}) : a \in A, b \in B)$ be a system of operations such that $\bullet_{a,b}$ is from $C_a \times D_b$ to $P_{a,b}$ and it is isotone and left-continuous in both arguments, i. e.

1a) $c_1 \leq c_2$ implies $c_1 \bullet_{a,b} d \leq c_2 \bullet_{a,b} d$ for all $c_1, c_2 \in C_a$ and $d \in D_b$,
1b) $d_1 \leq d_2$ implies $c \bullet_{a,b} d_1 \leq c \bullet_{a,b} d_2$ for all $c \in C_a$ and $d_1, d_2 \in D_b$,

2a) if $c \bullet_{a,b} d \leq p$ for some $d \in D_b$, $p \in P_{a,b}$ and for all $c \in X \subseteq C_a$ then $\sup X \bullet_{a,b} d \leq p$,
2b) if $c \bullet_{a,b} d \leq p$ for some $c \in C_a$, $p \in P_{a,b}$ and for all $d \in Y \subseteq D_b$ then $c \bullet_{a,b} \sup Y \leq p$.

Then the tuple $\langle A, B, \mathcal{P}, R, \mathcal{C}, \mathcal{D}, \odot \rangle$ will be called a *heterogeneous formal context.*

Notice that if $C_a = D_b$ and $\bullet_{a,b}$ is commutative these conditions can be reduced to these two:

1) $c_1 \leq c_2$ implies $c_1 \bullet_{a,b} d \leq c_2 \bullet_{a,b} d$ for all $c_1, c_2, d \in C_a = D_b$,
2) if $c \bullet_{a,b} d \leq p$ for some $d \in C_a = D_b$, $p \in P$ and for all $c \in X \subseteq C_a = D_b$ then $\sup X \bullet_{a,b} d \leq p$.

Let F be the set of all functions f with the domain A such that $f(a) \in C_a$, for all $a \in A$ (i. e., more formally, $F = \prod_{a \in A} C_a$) and G be the set of all functions g with the domain B such that $g(b) \in D_b$, for all $b \in B$. (i. e. $G = \prod_{b \in B} D_b$).

Define the following mapping $\nearrow : G \to F$: If $g \in G$ then $\nearrow(g) \in F$ is defined by

$$(\nearrow(g))(a) = \sup\{c \in C_a : (\forall b \in B) c \bullet_{a,b} g(b) \leq R(a,b)\}.$$

Symmetrically define the mapping $\swarrow : F \to G$: If $f \in F$ then $\swarrow(f) \in G$ is defined as following:

$$(\swarrow(f))(b) = \sup\{d \in D_b : (\forall a \in A) f(a) \bullet_{a,b} d \leq R(a,b)\}.$$

Theorem 1. *Let $f \in F$ and $g \in G$. Then the following conditions are equivalent:*

1) $f \leq \nearrow(g)$.
2) $g \leq \swarrow(f)$.
3) $f(a) \bullet_{a,b} g(b) \leq R(a,b)$ for all $a \in A$ and $b \in B$.

Corollary 1. *Mappings $\nearrow$ and $\swarrow$ form a Galois connection.*

Corollary 2.

1a) $g_1 \leq g_2$ implies $\nearrow(g_1) \geq \nearrow(g_2)$.
1b) $f_1 \leq f_2$ implies $\swarrow(f_1) \geq \swarrow(_2)$.
2a) $g \leq \swarrow(\nearrow(g))$.
2b) $f \leq \nearrow(\swarrow(f))$.
3a) $\nearrow(g) = \nearrow(\swarrow(\nearrow(g)))$.
3b) $\swarrow(f) = \swarrow(\nearrow(\swarrow(f)))$.

We use a Galois connection $(\nearrow, \swarrow)$ for the concept lattice construction via classical Ganter-Wille's approach from [9].

Lemma 1. *1) Let $\{g_i : i \in I\} \subseteq G$. Then*

$$\nearrow\left(\bigvee_{i \in I} g_i\right) = \bigwedge_{i \in I} \nearrow(g_i).$$

2) Let $\{f_i : i \in I\} \subseteq F$. Then

$$\swarrow\left(\bigvee_{i\in I} f_i\right) = \bigwedge_{i\in I} \swarrow(f_i).$$

By a *concept* we will understand a pair $\langle g, f\rangle$ from $G \times F$ such that $\nearrow(g) = f$ and $\swarrow(f) = g$.

Lemma 2. *If $\langle g_1, f_1\rangle$ and $\langle g_2, f_2\rangle$ are concepts then $g_1 \leq g_2$ iff $f_1 \geq f_2$.*

This lemma allows to define the following ordering of concepts: $\langle g_1, f_1\rangle \leq \langle g_2, f_2\rangle$ iff $g_1 \leq g_2$ (or equivalently $f_1 \geq f_2$).

The poset of all such concepts ordered by $\leq$ will be called a *heterogeneous concept lattice* and denoted by $\mathrm{HCL}(A, B, \mathcal{P}, R, \mathcal{C}, \mathcal{D}, \odot, \swarrow, \nearrow, \leq)$.

The following theorem shows that the word *lattice* in its name corresponds with reality.

Theorem 2. *(The Basic Theorem on Heterogeneous Concept Lattices)*

1) A heterogeneous concept lattice $\mathrm{HCL}(A, B, \mathcal{P}, R, \mathcal{C}, \mathcal{D}, \odot, \swarrow, \nearrow, \leq)$ is a complete lattice in which

$$\bigwedge_{i\in I}\langle g_i, f_i\rangle = \left\langle \bigwedge_{i\in I} g_i, \nearrow\left(\swarrow\left(\bigvee_{i\in I} f_i\right)\right)\right\rangle$$

and

$$\bigvee_{i\in I}\langle g_i, f_i\rangle = \left\langle \swarrow\left(\nearrow\left(\bigvee_{i\in I} g_i\right)\right), \bigwedge_{i\in I} f_i\right\rangle.$$

2) For each $a \in A$, $b \in B$, let $P_{a,b}$ have the least element $0_{P_{a,b}}$ such that $0_{C_a} \bullet_{a,b} d = c \bullet_{a,b} 0_{D_b} = 0_{P_{a,b}}$, for all $c \in C_a$, $d \in D_b$. Then a complete lattice L is isomorphic to $\mathrm{HCL}(A, B, \mathcal{P}, R, \mathcal{C}, \mathcal{D}, \odot, \swarrow, \nearrow, \leq)$ if and only if there are mappings $\alpha : \bigcup_{a\in A}(\{a\} \times C_a) \to L$ and $\beta : \bigcup_{b\in B}(\{b\} \times D_b) \to L$ such that:

a) α does not increase in the second argument (for the fixed first one).

b) β does not decrease in the second argument (for the fixed first one).

c) $\mathrm{Rng}(\alpha)$ is $\inf$-dense in L.

d) $\mathrm{Rng}(\beta)$ is $\sup$-dense in L.

e) For every $a \in A$, $b \in B$ and $c \in C_a$, $d \in D_b$

$$\alpha(a, c) \geq \beta(b, d) \qquad \text{if and only if} \qquad c \bullet_{a,b} d \leq R(a, b).$$

3 Galois connectional approach

In this section, we recall the basic definitions and results of approach from [19], [20] which is inspired by the (homogeneous) approach from [23].

Let A and B be non-empty sets. Let $\mathcal{C} = ((C_a, \leq_{C_a}) : a \in A)$ and $\mathcal{D} = ((D_b, \leq_{D_b}) : b \in B)$ be systems of complete lattices. Let $\mathcal{G} = ((\phi_{a,b}, \psi_{a,b}) : a \in A, b \in B)$ be a system of (antitone) Galois connection s.t. $(\phi_{a,b}, \psi_{a,b})$ is a Galois connection from $(C_a, \leq_{C_a})$ to $(D_b, \leq_{D_b})$. (Again we will omit the indices of all noticed $\leq_?$.)

Define the following mapping $\uparrow : G \to F$: If $g \in G$ then $\uparrow(g) \in F$ is defined by

$$(\uparrow(g))(a) = \bigwedge_{b \in B} \psi_{a,b}(g(b)).$$

Symmetrically define the mapping $\downarrow : F \to G$: If $f \in F$ then $\downarrow(f) \in G$ is defined as following:

$$(\downarrow(f))(b) = \bigwedge_{a \in A} \phi_{a,b}(f(a)).$$

Theorem 3. *$(\uparrow, \downarrow)$ is a Galois connection.*

Hence the classical Ganter-Wille's process can be used for the concept lattice construction, so it can be obtained the following.

By a *concept* in this approach it will be understand a pair $\langle g, f \rangle$ from $G \times F$ such that $\uparrow(g) = f$ and $\downarrow(f) = g$.

Lemma 3. *If $\langle g_1, f_1 \rangle$ and $\langle g_2, f_2 \rangle$ are concepts then $g_1 \leq g_2$ iff $f_1 \geq f_2$.*

This lemma allows to define the following ordering of concepts: $\langle g_1, f_1 \rangle \leq \langle g_2, f_2 \rangle$ iff $g_1 \leq g_2$ (or equivalently $f_1 \geq f_2$).

The poset of all such concepts ordered by $\leq$ will be called a *connectional concept lattice* and denoted by $\mathrm{CCL}(A, B, \mathcal{C}, \mathcal{D}, \mathcal{G}, \downarrow, \uparrow, \leq)$.

Theorem 4. *(The Basic Theorem on Connectional Concept Lattices)*

1) A connectional concept lattice $\mathrm{CCL}(A, B, \mathcal{C}, \mathcal{D}, \mathcal{G}, \downarrow, \uparrow, \leq)$ *is a complete lattice in which*

$$\bigwedge_{i \in I} \langle g_i, f_i \rangle = \left\langle \bigwedge_{i \in I} g_i, \uparrow\left(\downarrow\left(\bigvee_{i \in I} f_i\right)\right)\right\rangle$$

and

$$\bigvee_{i \in I} \langle g_i, f_i \rangle = \left\langle \downarrow\left(\uparrow\left(\bigvee_{i \in I} g_i\right)\right), \bigwedge_{i \in I} f_i \right\rangle.$$

2) A complete lattice L is isomorphic to $\mathrm{CCL}(A, B, \mathcal{C}, \mathcal{D}, \mathcal{G}, \downarrow, \uparrow, \leq)$ *if and only if there are mappings $\alpha : \bigcup_{a \in A}(\{a\} \times C_a) \to L$ and $\beta : \bigcup_{b \in B}(\{b\} \times D_b) \to L$ such that for every $a \in A$, $b \in B$ and $c \in C_a$, $d \in D_b$*

$$\alpha(a, c) \geq \beta(b, d) \qquad \textit{iff} \qquad d \leq \phi_{a,b}(c) \qquad \textit{iff} \qquad c \leq \psi_{a,b}(d).$$

4 Heterogeneous approach can be expressed by connectional one

In this section, we modify method from [19] which was used for the proof that connectional approach covers a generalized approach from [11] and [12]. It used a notion of G-ideals defined in [24].

Let $(L, \leq_L)$, $(M, \leq_M)$ be complete lattices. Then $J \subseteq L \times M$ is called a *G-ideal* of $L \times M$ when the following conditions hold:

1) If $(\ell, m) \in J$ and $(\ell', m') \leq (\ell, m)$ (coordinate-wise, i.e. $\ell' \leq \ell$ and $m' \leq m$) then $(\ell', m') \in J$.
2) If $\{(\ell_i, m_i) : i \in I\} \subseteq J$ then $(\bigvee_{i\in I} \ell_i, \bigwedge_{i\in I} m_i), (\bigwedge_{i\in I} \ell_i, \bigvee_{i\in I} m_i) \in J$.
   If $I = \emptyset$ then $(0_L, 1_M), (1_L, 0_M) \in J$.

Theorem 5. *[24] Let $(L, \leq_L)$, $(M, \leq_M)$ be complete lattices.*

1) If (ϕ, ψ) is an (antitone) Galois connection from $(L, \leq_L)$ to $(M, \leq_M)$ then

$$\{(\ell, m) : \phi(\ell) \geq_M m\} = \{(\ell, m) : \psi(m) \geq_L \ell\}$$

is a G-ideal on $L \times M$.

2) If J is a G-ideal on $L \times M$ then the mappings $\phi : L \to M$ and $\psi : M \to L$ defined by

$$\phi(\ell) = \bigvee\{m \in M : (\ell, m) \in J\}$$

and

$$\psi(m) = \bigvee\{\ell \in L : (\ell, m) \in J\}$$

form a Galois connection from $(L, \leq_L)$ to $(M, \leq_M)$.

Moreover, this correspondences between Galois connections and G-ideals are each other inverse.

The paper [19] uses these facts in the following way:

Lemma 4. *Let $(L, \leq_L)$, $(M, \leq_M)$ be complete lattices, $(P, \leq_P)$ be poset and $\bullet : L \times M \to P$ is isotone and left-continuous in both arguments. Then*

$$\{(\ell, m) : \ell \bullet m \leq p\}$$

is a G-ideal.

Assume that we have a heterogeneous concept lattice HCL$(A, B, \mathcal{P}, R, \mathcal{C}, \mathcal{D}, \odot, \swarrow, \nearrow, \leq)$. For each $a \in A$ and $b \in B$ define

$$J_{a,b} = \{(c, d) \in C_a \times D_b : c \bullet_{a,b} d \leq R(a, b)\},$$

by the previous Lemma 4 we know that $J_{a,b}$ is a G-ideal on $C_a \times D_b$. Then again for each $a \in A$ and $b \in B$ define the mappings $\phi_{a,b} : C_a \to D_b$ and $\psi_{a,b} : D_b \to C_a$ defined by

$$\phi_{a,b}(c) = \bigvee\{d \in D_b : (c, d) \in J_{a,b}\}$$

and

$$\psi_{a,b}(d) = \bigvee\{c \in C_a : (c,d) \in J_{a,b}\}$$

and we know by Theorem 5 that $(\phi_{a,b}, \psi_{a,b})$ is a Galois connection from C_a to D_b. Finally, we define mappings $\downarrow$ and $\uparrow$ as before:

$$(\uparrow(g))(a) = \bigwedge_{b\in B} \psi_{a,b}(g(b)), \qquad (\downarrow(f))(b) = \bigwedge_{a\in A} \phi_{a,b}(f(a)).$$

Theorem 6. $(\uparrow, \downarrow) = (\nearrow, \swarrow)$.

Proof. We prove $\uparrow = \nearrow$ only, the second equality can be proved dually. Let $g \in G$ and $a \in A$, we are going to prove $(\uparrow(g))(a) = (\nearrow(g))(a)$.

By the definition we have

$$(\uparrow(g))(a) = \bigwedge_{b\in B} \psi_{a,b}(g(b)) = \bigwedge_{b\in B} \bigvee\{c \in C_a : (c, g(b)) \in J_{a,b}\}$$

$$= \bigwedge_{b\in B} \bigvee\{c \in C_a : c \bullet_{a,b} g(b) \le R(a,b)\}$$

and

$$(\nearrow(g))(a) = \sup\{c \in C_a : (\forall b \in B) c \bullet_{a,b} g(b) \le R(a,b)\}.$$

Denote

$$X = \{c \in C_a : (\forall b \in B) c \bullet_{a,b} g(b) \le R(a,b)\}$$

and, for each $b \in B$,

$$X_b = \{c \in C_a : c \bullet_{a,b} g(b) \le R(a,b)\},$$

then we want to prove $\bigwedge_{b\in B} \sup X_b = \sup X$.

$\geq$ For each $b \in B$ we have $X_b \supseteq X$ hence $\sup X_b \geq \sup X$. It follows that $\bigwedge_{b\in B} \sup X_b \geq \sup X$.

$\leq$ Let $b \in B$. Then for each $c \in X_b$ we have $c \bullet_{a,b} g(b) \le R(a,b)$. By the left--continuity of $\bullet_{a,b}$ in the first argument we have $\sup X_b \bullet_{a,b} g(b) \le R(a,b)$. Because clearly $\bigwedge_{b'\in B} \sup X_{b'} \le \sup X_b$, by the isotonity of $\bullet_{a,b}$ in the first argument $\bigwedge_{b'\in B} \sup X_{b'} \bullet_{a,b} g(b) \le R(a,b)$. This holds for each $b \in B$, which means that $\bigwedge_{b'\in B} \sup X_{b'} \in X$, hence $\bigwedge_{b'\in B} \sup X_{b'} \le \sup X$.

□

5 Connectional approach can be expressed by heterogeneous one

In this section we show opposite direction to the previous one, namely that the heterogeneous approach covers the connectional one, moreover by the surprisingly simply way.

Firstly, one fact from [24] analogous to Lemma 1:

Lemma 5. *Let $(L, \leq_L)$, $(M, \leq_M)$ be complete lattices and (ϕ, ψ) be a Galois connection from $(L, \leq_L)$ to $(M, \leq_M)$.*

1) For arbitrary subset $\{\ell_i : i \in I\}$ of L

$$\phi\left(\bigvee_{i\in I} \ell_i\right) = \bigwedge_{i\in I} \phi(\ell_i).$$

2) For arbitrary subset $\{m_i : i \in I\}$ of M

$$\psi\left(\bigvee_{i\in I} m_i\right) = \bigwedge_{i\in I} \psi(m_i).$$

We use it in the following way:

Theorem 7. *Let $(L, \leq_L)$, $(M, \leq_M)$ be complete lattices and (ϕ, ψ) be a Galois connection from $(L, \leq_L)$ to $(M, \leq_M)$. Let $\bullet : L \times M \to (\{0,1\}, \leq)$ be defined in the following way:*

$$\ell \bullet m = \begin{cases} 0 & \textit{if } \phi(\ell) \geq m \textit{ (iff } \psi(m) \geq \ell\textit{)}, \\ 1 & \textit{elsewhere.} \end{cases}$$

Then $\bullet$ is isotone and left-continuous in both arguments.

Proof. Because of duality, it is enough to prove isotonity and left-continuity in the first argument.

- Let $\ell_1, \ell_2 \in L$ where $\ell_1 \leq \ell_2$ and $m \in M$. We want to prove that $\ell_1 \bullet m \leq \ell_2 \bullet m$.
  - If $\ell_2 \bullet m = 1$, the inequality is trivial.
  - If $\ell_2 \bullet m = 0$, then by the definition $\phi(\ell_2) \geq m$. Because (ϕ, ψ) be a Galois connection and $\ell_1 \leq \ell_2$, we have $\phi(\ell_1) \geq \phi(\ell_2)$ which by transitivity implies $\phi(\ell_1) \geq m$. So, by the definition $\ell_1 \bullet m = 0$ hence $\ell_1 \bullet m \leq \ell_2 \bullet m$.
- Let $m \in M$, $X \subseteq L$ and $\ell \bullet m \leq p$ for all $\ell \in X$. We want to prove that $\sup X \bullet m \leq p$.
  - If $p = 1$, the inequality is trivial.
  - If $p = 0$, then by the definition $\phi(\ell) \geq m$ for all $\ell \in X$ which means $\bigwedge_{\ell\in X} \phi(\ell) \geq m$. Because (ϕ, ψ) is a Galois connection, by Lemma 5 we have $\bigwedge_{\ell\in X} \phi(\ell) = \phi(\sup_{\ell\in X} \ell)$. This implies $\phi(\sup_{\ell\in X} \ell) \geq m$, so, by the definition $\sup_{\ell\in X} \ell \bullet m = 0$.

Assume that we have a connectional concept lattice $\mathrm{CCL}(A, B, \mathcal{C}, \mathcal{D}, \mathcal{G}, \downarrow, \uparrow, \leq)$. For each $a \in A$ and $b \in B$ take the same $P_{a,b} = (\{0,1\}, \leq)$, $R(a,b) = 0$ (sic!) and $\bullet_{a,b} : C_a \times D_b \to P_{a,b}$ such that for all $c \in C_a$ and $d \in D_b$,

$$c \bullet_{a,b} d = \begin{cases} 0 & \text{if } \phi_{a,b}(c) \geq d \text{ (iff } \psi_{a,b}(d) \geq c), \\ 1 & \text{elsewhere.} \end{cases}$$

By Theorem 7 $\bullet_{a,b}$ is isotone and left-continuous in both arguments, so we have a frame for heterogeneous approach a we can define the mappings $\nearrow$ and $\swarrow$ as before.

Theorem 8. $(\nearrow, \swarrow) = (\uparrow, \downarrow)$.

Proof. We prove $\nearrow = \uparrow$ only, the second equality can be proved dually. Let $g \in G$ and $a \in A$. Then by the definitions we have

$$(\nearrow(g))(a) = \sup\{c \in C_a : (\forall b \in B) c \bullet_{a,b} g(b) \leq R(a,b)\} =$$
$$= \sup\{c \in C_a : (\forall b \in B) c \bullet_{a,b} g(b) \leq 0\} =$$
$$= \sup\{c \in C_a : (\forall b \in B) \psi_{a,b}(g(b)) \geq c\} =$$
$$= \sup\{c \in C_a : \bigwedge_{b \in B} \psi_{a,b}(g(b)) \geq c\} = \bigwedge_{b \in B} \psi_{a,b}(g(b)) = (\uparrow(g))(a).$$

□

6 Conclusions

In this paper we recall two rather new common platform for till-known fuzzifications of the Formal Concept Analysis which work on the context without limitation of the same data-types of objects and/or attributes. The first one arises, defined in [1], as rather straightforward extension of the previous so-called generalized approach from [11] and [12] to such heterogeneous context. The second one, from [19] and [20] is based on interesting idea to put some Galois connection to each field of the table. We show that each of these two approaches covers and is covered by the other one (in some canonical way).

In the end, let us say one "philosophical" aspect about our approach (that from Section 2). In this case, a pair consisting of some $\bullet$ and some value is put into each field of the table. The part $\bullet$ can be understood as behavior of the corresponding object with respect to the corresponding attribute. This behavior can be known long before than data come to the table, hence it can be thought as *metadata*. Data can change through the time but this metadata are fixed. In other words, we divide information on relationship of an object and an attribute to the stable and dynamic part. (Of course, this division has meaning only in the case that we consider possible changing of the data in the table.) In our opinion, the connectional approach has not this advantage, because it mixes metadata and data parts.

Then we can formulate this problem: In Section 5 we can see a surprising (and maybe suspicious) transformation of connectional approach to heterogeneous with the data part constantly equal to 0, i.e. all this is transformed to metadata part. The question is: Is there some other (natural, canonical) transformation which is not constant?

References

1. Ľ. Antoni, S. Krajči, O. Krídlo, B. Macek, L. Pisková, On heterogeneous formal contexts. Submitted to Fuzzy Sets and Systems.
2. R. Bělohlávek, Fuzzy concepts and conceptual structures: induced similarities, JCIS′98, Vol. I, pp. 179–182, Durham, USA, 1998

3. R. Bělohlávek, Concept Lattices and Order in Fuzzy Logic, Annals of Pure and Applied Logic, 128 (2004), 277–298
4. R. Bělohlávek, V. Sklenář, J. Zacpal, Crisply generated fuzzy concepts, in: B. Ganter and R. Godin (Eds.): ICFCA 2005, Lecture Notes in Computer Science 3403, pp. 268–283, Springer-Verlag, Berlin/Heidelberg, 2005.
5. R. Bělohlávek, V. Vychodil, Reducing the size of fuzzy concept lattices by hedges, FUZZ-IEEE 2005, USA, 663–668, ISBN: 0-7803-9159-4
6. R. Bělohlávek, V. Vychodil, Formal concept analysis and linguistic hedges, submitted to International Journal of General Systems, 2012
7. S. Ben Yahia, A. Jaoua, Discovering knowledge from fuzzy concept lattice. In: Kandel A., Last M., Bunke H.: Data Mining and Computational Intelligence, 169–190, Physica-Verlag, 2001
8. A. Burusco, R. Fuentes-Gonzalez, The study of L-fuzzy concept lattice, Mathware & Soft Computing 3(1994), 209–218
9. B. Ganter, R. Wille, Formal Concept Analysis, Mathematical Foundation, Springer Verlag 1999, ISBN 3-540-62771-5
10. S. Krajči, Cluster based efficient generation of fuzzy concepts, Neural Network World 13,5 (2003) 521–530
11. S. Krajči, A generalized concept lattice, Logic Journal of IGPL, 13 (5): 543–550, 2005.
12. S. Krajči, The basic theorem on generalized concept lattice, CLA 2004, Ostrava, proceedings of the 2nd international workshop, eds. V. Snášel, R. Bělohlávek, ISBN 80-248-0597-9, 25–33
13. S. Krajči, Every Concept Lattice With Hedges Is Isomorphic To Some Generalized Concept Lattice, CLA 2005, proceedings of CLA 2005: The 3rd international conference on Concept Lattice and Their Applications, eds. V. Snášel, R. Bělohlávek, ISBN 80-248-0863-3, 1–9
14. J. Medina, M. Ojeda-Aciego: Multi-adjoint t-concept lattices, Information Sciences 180 (5), pp. 712–725, 2010.
15. J. Medina, M. Ojeda-Aciego, J. Ruiz-Calviño, Formal concept analysis via multi-adjoint concept lattices. Fuzzy Sets and Systems, 160(2), 130–144, 2009.
16. J. Medina, M. Ojeda-Aciego, A. Valverde, P. Vojtáš, Towards biresiduated multi-adjoint logic programming. Lect. Notes in Artificial Intelligence, 3040: 608–617, 2004.
17. J. Medina, M. Ojeda-Aciego, P. Vojtáš, Multi-adjoint logic programming with continuous semantics. Logic programming and Non-Monotonic Reasoning, LP-NMR'01, 351–364, Lect. Notes in Artificial Intelligence 2173, 2001.
18. J. Medina, M. Ojeda-Aciego, P. Vojtáš, Similarity-based unification: a multi-adjoint approach. Fuzzy Set and Systems, 146: 43–62, 2004.
19. J. Pócs, Note on generating fuzzy concept lattices via Galois connections, Information Sciences 185 (1), pp. 128–136, 2012.
20. J. Pócs, On possible generalization of fuzzy concept lattices using dually isomorphic retracts, Information Sciences 210, pp. 89–98, 2012.
21. S. Pollandt, Fuzzy Begriffe, Springer, 1997
22. S. Pollandt, Datenanalyse mit Fuzzy-Begriffen, in: G. Stumme, R. Wille: Begriffliche Wissensverarbeitung. Methoden und Anwendungen, Springer, Heidelberg 2000, 72–98
23. A. Popescu, A general approach to fuzzy concepts, Mathematical Logic Quarterly 50 (3) (2004), pp. 265–280
24. Z. Shmuely, The structure of Galois connections, Pacific Journal of Mathematics, Vol. 54, No. 2, 1974, pp. 209–225

Concepts and Types – An Application to Formal Language Theory

Christian Wurm

Fakultät für Linguistik und Literaturwissenschaften
Universität Bielefeld
cwurm@uni-bielefeld.de

Abstract. We investigate how formal concept analysis can be applied in a type-theoretic context, namely the context of λ-terms typed à la Curry. We first show some general results, which reveal that concept lattices generally respect and reflect the type structure of terms. Then, we show an application of the results in formal language theory, where we vastly generalize existing approaches of capturing the distributional structure of languages by means of formal concept analysis. So type theory is interesting to formal concept analysis not only as a particular context, but also because it allows to generalize existing contexts.

1 Introduction

Formal concept analysis (FCA) operates within what is called a context, that is typically a set of objects, a set of attributes and a relation between them. We will have a closer look at formal concept analysis over terms in a type theoretic context, and show how this can be applied to formal language theory. Our first contribution[1] is that we consider the case not of an atomic typing, but rather recursive, Curry-style typing. Our objects are λ-terms, which have to be assigned types according to their syntactic structure. We show some interesting correlations between the type theoretic structure of sets of terms, their principal typing, and how this these interact with FCA and its lattice-theoretic operations. Our second main contribution is that we show an application to formal language theory. There exist interesting approaches to the distributional structure of languages using FCA over the relation of strings and the contexts in which they occur. However, all of these have some major limitations, as they only work with simple string concatenation, but cannot cope with, for example, the concept of string duplication in a language. We use a type theoretic encoding of strings as terms, and show that this allows us to vastly generalize existing approaches.

2 A Simple Type Theory

Type theory starts with a (usually) finite set of basic types, and a finite, (usually) small set of type constructors. Types are usually interpreted as sets; we denote the

[1] There is considerable work on FCA in a typed context, see, for example, [8]).

set of all objects of type τ by $\|\tau\|$. We will consider only a single type constructor, the usual $\to$, where for types σ, τ, $\sigma \to \tau$ is the type of all functions from $\|\sigma\|$ to $\|\tau\|$. Basic objects are assigned some type, and all new objects we can construct in our universe must be constructed in accordance with a typing procedure, that is, we have to make sure that they can be assigned at least one type. Objects which are not well-typed do not exist in the typed universe.

Given a non-empty set A of atomic types, the set of types $Tp(A)$ is defined as closure of A under type constructors: $A \subseteq Tp(A)$, and if $\sigma, \tau \in Tp(A)$, then $\sigma \to \tau \in Tp(A)$. The **order** of a type is defined as $ord(\sigma) = 0$ for $\sigma \in A$, $ord(\sigma \to \tau) = max(ord(\sigma) + 1, ord(\tau))$.

We define a higher order signature as $\Sigma := (A, C, \phi)$, where A is a finite set of atomic types, C is a set of constants, and $\phi : C \to Tp(A)$ assigns types to constants. The order of Σ is $max(\{ord(\phi(c)) : c \in C\})$. Let X be a countable set of variables. The set $\mathtt{Tm}(\Lambda(\Sigma))$, the set of all λ terms over Σ, is the closure of $C \cup X$ under the following rules: 1. $C \cup X \subseteq \mathtt{Tm}(\Lambda(\Sigma))$; 2. if $\mathtt{m}, \mathtt{n} \in \mathtt{Tm}(\Lambda(\Sigma))$, then $(\mathtt{mn}) \in \mathtt{Tm}(\Lambda(\Sigma))$; 3. if $x \in X, \mathtt{m} \in \mathtt{Tm}(\Lambda(\Sigma))$, then $(\lambda x.\mathtt{m}) \in \mathtt{Tm}(\Lambda(\Sigma))$.

We omit the outermost parentheses $(,)$ for λ terms, and write $\lambda x_1...x_n.\mathtt{m}$ for $\lambda x_1.(\dots(\lambda x_n.\mathtt{m})...)$; furthermore, we write $\mathtt{m}_1\mathtt{m}_2\dots\mathtt{m}_i$ for $(\dots(\mathtt{m}_1\mathtt{m}_2)\dots\mathtt{m}_i)$. The set of free variables of a term $\mathtt{m}$, $FV(\mathtt{m})$, is defined by 1. $FV(x) = \{x\} : x \in X$, 2. $FV(c) = \emptyset : c \in C$, 3. $FV(\mathtt{mn}) = FV(\mathtt{m}) \cup FV(\mathtt{n})$, and 4. $FV(\lambda x.\mathtt{m}) = FV(\mathtt{m}) - \{x\}$. $\mathtt{m}$ is closed if $FV(\mathtt{m}) = \emptyset$. We write $\mathtt{m}[\mathtt{n}/x]$ for the result of substituting $\mathtt{n}$ for all free occurrences of x in $\mathtt{m}$. α conversion is defined as $\lambda x.\mathtt{m} \rightsquigarrow_\alpha \lambda y.\mathtt{m}[y/x]$. A β-redex is a term of the form $(\lambda x.\mathtt{m})\mathtt{n}$. We write $\rightsquigarrow_\beta$ for β reduction, so we have $(\lambda x.\mathtt{m})\mathtt{n} \rightsquigarrow_\beta \mathtt{m}[\mathtt{n}/x]$. The inverse of β reduction is β expansion. Let $[\mathtt{m}]_\beta$ denote the β normal form of $\mathtt{m}$, that is, the term without any β redex. This term is unique up to α conversion for every term $\mathtt{m}$. We denote by $=_{\alpha\beta}$ the smallest **congruence** which contains both $\rightsquigarrow_\alpha$ and $\rightsquigarrow_\beta$. We thus write $\mathtt{m} =_{\alpha\beta} \mathtt{n}$, if $\mathtt{n}$ can be derived from $\mathtt{m}$ with any finite series of steps of β-reduction, expansion or α-conversion of any of its subterms.

We now come to the procedure of assigning types to terms.[2] A *type environment* is a (possibly empty) set $\{x_1 : \alpha_1, \dots x_n : \alpha_n\}$ of pairs of variables and types, where each variable occurs at most once. A λ-term $\mathtt{m}$ with $FV(\mathtt{m}) = \{x_1, \dots, x_n\}$ can be assigned a type α in the signature $\Sigma = (A, C, \phi)$ and type environment $\{x_1 : \alpha_1, \dots x_n : \alpha_n\}$, in symbols

(1) $x_1 : \alpha_1, \dots x_n : \alpha_n \vdash_\Sigma \mathtt{m} : \alpha$,

if it can be derived according to the following rules:

(cons) $\vdash_\Sigma c : \phi(c)$, for $c \in C$;

(var) $x : \alpha \vdash_\Sigma x : \alpha$, where $x \in X$ and $\alpha \in Tp(A)$;

[2] We adopt what is known as Curry-style typing: in Church-style typing, terms cannot be constructed without types; in Curry-style typing, terms are first constructed and then assigned a type; so there might be the case that there is no possible assignment.

(abs) $$\frac{\Gamma \vdash_{\Sigma} \mathtt{m} : \beta}{\Gamma - \{x : \alpha\} \vdash_{\Sigma} \lambda x.\mathtt{m} : \alpha \to \beta}$$, provided $\Gamma \cup \{x : \alpha\}$ is a type environment;

(app) $$\frac{\Delta \vdash_{\Sigma} \mathtt{n} : \alpha \quad \Gamma \vdash_{\Sigma} \mathtt{m} : \alpha \to \beta}{\Gamma \cup \Delta \vdash_{\Sigma} \mathtt{mn} : \beta}$$, provided $\Gamma \cup \Delta$ is a type environment.

An expression of the form $\Gamma \vdash_{\Sigma} \mathtt{m} : \alpha$ is called a *judgment*, and if it is derivable by the above rules, it is called the *typing* of $\mathtt{m}$. A term $\mathtt{m}$ is called *typable* if it has a typing. If in a judgment we do not refer to any particular signature, we also write $\Gamma \vdash \mathtt{m} : \alpha$. Regarding β reduction, we have the following well-known result:

Theorem 1 *(Subject Reduction Theorem) If $\Gamma \vdash \mathtt{m} : \alpha$, $\mathtt{m} \leadsto_{\beta} \mathtt{m}'$, then $\Gamma' \vdash \mathtt{m}' : \alpha$, where Γ' is the restriction of Γ to $FV(\mathtt{m}')$.*

Let $\mathtt{m} \leadsto_{\beta} \mathtt{m}'$ be a contraction of a redex $(\lambda x.\mathtt{n})\mathtt{o}$. This reduction is *non-erasing* if $x \in FV(\mathtt{n})$, and *non-duplicating* if x occurs free in $\mathtt{n}$ at most once. A reduction from $\mathtt{m}$ to $\mathtt{m}'$ is non-erasing (non-duplicating) if all of its reduction steps are non-erasing (non-duplicating). We say a term $\mathtt{m}$ is linear, if for each subterm $\lambda x.\mathtt{n}$ of $\mathtt{m}$, x occurs free in $\mathtt{n}$ exactly once, and each free variable of $\mathtt{m}$ has just one occurrence free in $\mathtt{m}$. Linear λ-terms are thus the terms, for which each β-reduction is non-erasing and non-duplicating. We will be mainly interested in a slightly larger class. A term $\mathtt{m}$ is a λI term, if for each subterm $\lambda x.\mathtt{n}$ of $\mathtt{m}$, x occurs free in $\mathtt{m}$ at least once. λI terms are thus the terms which do not allow for vacuous abstraction (see [1], chapter 9 for extensive treatment). Another important result for us is the following: obviously, by our typing procedure a single term might be possibly assigned many types. We call a type substitution a map $\pi : A \to Tp(A)$, which respects the structure of types: $\pi(\beta \to \gamma) = (\pi(\beta)) \to (\pi(\gamma))$, for $\beta, \gamma \in Tp(A)$.

Theorem 2 *(Principal Type Theorem) Let $\mathtt{m}$ be a term, and let $\Theta := \{\alpha : \Gamma \vdash \mathtt{m} : \alpha$ is derivable$\}$ be the set of all types which can be assigned to $\mathtt{m}$. If $\Theta \neq \emptyset$, then there exists a* ***principal type*** *β for $\mathtt{m}$, such that $\Gamma \vdash \mathtt{m} : \beta$ is derivable, and for each $\alpha \in \Theta$, there is a substitution π_{α} such that $\alpha = \pi_{\alpha}(\beta)$.*

Obviously, β is unique up to isomorphism; we will write $pt(\mathtt{m})$ for the principal type of $\mathtt{m}$. The proof of the theorem is constructive, that is, β can be effectively computed or shown to be nonexistent, see [6].

3 Types and Concepts

3.1 A Context of Terms

We now give a short introduction into formal concept analysis. A context is a triple $(\mathcal{G}, \mathcal{M}, I)$, where $\mathcal{G}, \mathcal{M}$ are sets and $I \subseteq \mathcal{G} \times \mathcal{M}$. In FCA, the entities in $\mathcal{G}$ are thought of as objects, the objects in $\mathcal{M}$ as attributes, and for $m \in \mathcal{M}$, $g \in \mathcal{G}$, we have $(g, m) \in I$ if the object g has the attribute m. This is all we need as basic structure to get the machine of FCA going. For $A \subseteq \mathcal{G}, B \subseteq \mathcal{M}$, we put $A^{\triangleright} := \{m \in \mathcal{M} : \forall a \in A, (a, m) \in I\}$, and $B^{\triangleleft} := \{g \in \mathcal{G} : \forall m \in B, (a, m) \in I\}$.

A concept is a pair (A, B) such that $A^{\triangleright} = B, B^{\triangleleft} = A$. We call A the **extent** and B the **intent**. A is the extent of a concept iff $A = A^{\triangleright\triangleleft}$, dually for intents. The maps $[-]^{\triangleright}, [-]^{\triangleleft}$ are called **polar maps**. We order concepts by inclusion of extents, that is, $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2$.

Definition 3 *Given a context* $\mathfrak{B} = (\mathcal{G}, \mathcal{M}, I)$*, we define the concept lattice of* $\mathfrak{B}$ *as* $\mathcal{L}(\mathfrak{B}) = \langle \mathfrak{C}, \wedge, \vee, \top, \bot \rangle$*, where* $\top = (\mathcal{G}, \mathcal{G}^{\triangleright})$*,* $\bot = (\mathcal{M}^{\triangleleft}, \mathcal{M})$*, and for* $(A_i, B_i), (A_j, B_j) \in \mathfrak{C}$*,* $(A_i, B_i) \wedge (A_j, B_j) = (A_i \cap A_j, (B_i \cup B_j)^{\triangleleft\triangleright})$*, and* $(A_i, B_i) \vee (A_j, B_j) = ((A_i \cup A_j)^{\triangleright\triangleleft}, B_i \cap B_j)$*.*

We define our type theoretic context as follows. Recall that $\mathtt{Tm}(\Lambda(\Sigma))$ is the set of all λ-terms over Σ. We put $\mathtt{Tm}_c(\Lambda(\Sigma)) := \{\mathtt{m} \in \mathtt{Tm}(\Lambda(\Sigma)) : FV(\mathtt{m}) = \emptyset\}$, the set of closed terms, and we call $\mathtt{Tm}_c(\Lambda I(\Sigma))$ the set of all closed λI terms. Furthermore, define $\mathtt{WTT}$ as the set of all closed and well-typed terms, that is, the set of all terms $\mathtt{m}$ such that $\vdash \mathtt{m} : \alpha$ is derivable for some α by our rules; $\mathtt{WTT}_I = \mathtt{WTT} \cap \mathtt{Tm}(\Lambda I(\Sigma))$. Recall that $=_{\alpha\beta}$ is a congruence. Let σ be a given type, and $L' \subseteq \|\sigma\|$ be a distinguished subset of the terms of type σ; we define $L := \{\mathtt{m} : \exists \mathtt{n} \in L' : \mathtt{m} =_{\alpha\beta} \mathtt{n}\}$, that is, as closure of L' under $=_{\alpha\beta}$. Put $\mathcal{G} = \mathcal{M} = \mathtt{Tm}_c(\Lambda(\Sigma))$, and define the relation $I \subseteq \mathtt{Tm}_c(\Lambda(\Sigma)) \times \mathtt{Tm}_c(\Lambda(\Sigma))$ as follows: for $\mathtt{m}, \mathtt{n} \in \mathtt{Tm}_c(\Lambda(\Sigma))$, we have $(\mathtt{m}, \mathtt{n}) \in I$ if $\mathtt{nm} \in L$. So the relation of objects in $\mathcal{M}$ and $\mathcal{G}$ is that of function and argument, and the relation I tells us whether the two yield a desired value. Same can be done with $\mathtt{Tm}_c(\Lambda I(\Sigma))$.

Obviously, we have $\bot = (\emptyset, \mathtt{Tm}_c(\Lambda(\Sigma)))$. Regarding upper bounds, we have to distinguish two important concepts: we first have a concept we denote $\mathsf{T} := (\mathtt{WTT}, \Lambda V)$, where ΛV (V for vacuous) is the set of all terms of the form $\lambda x.\mathtt{m}$, where $\mathtt{m} \in L$ and $x \notin FV(\mathtt{m})$. There is however a larger concept $\top \geq \mathsf{T}$, which is defined as $\top := (\mathtt{Tm}_c(\Lambda(\Sigma)), \emptyset)$. The reason for this slight complication is as follows: we want our terms to be closed, because open terms are meaningless for us. Now, it holds that the concatenation of closed terms is again a closed term; but the concatenation of well-typed terms need not be well-typed: for $\mathtt{m}, \mathtt{n} \in \mathtt{WTT}$, it might be that $\mathtt{nm} \notin \mathtt{WTT}$. Furthermore, there are λ-terms $\mathtt{n}$ with vacuous abstraction such that the set $\{\mathtt{nm} : \mathtt{m} \in \mathsf{T}\} \lneq \mathsf{T}$; we would however like our $\top$ to be absorbing; and in fact, if $\mathtt{m} \notin \mathtt{WTT}$, then for any term $\mathtt{n}$, $\mathtt{nm}, \mathtt{mn} \notin \mathtt{WTT}$. So for every term $\mathtt{m} \notin \mathtt{WTT}$, $\{\mathtt{m}\}^{\triangleright} = \emptyset$. For all interesting results we have to restrict ourselves to $\mathtt{WTT}$, but for completeness of some operations we have to consider $\mathtt{Tm}_c(\Lambda(\Sigma))$. Note however that if we restrict $\mathtt{Tm}_c(\Lambda(\Sigma))$ to $\mathtt{Tm}_c(\Lambda I(\Sigma))$, then $\top$ and T coincide.[3]

3.2 Concept Structure and Type Structure

We have seen that each term can be assigned a most general type. Importantly, the same holds for sets of types:

[3] This is actually not straightforward, but follows as a corollary from results we present later on.

Lemma 4 *Let $T \subseteq \texttt{WTT}$ be a set of terms, such that the set of principal types $\{pt(\texttt{m}) : \texttt{m} \in T\}$ is finite. If there is a set of types Θ, such that for each $\texttt{m} \in T$ and all $\theta \in \Theta$, $\vdash \texttt{m} : \theta$ is a derivable judgment, then there is a (up to isomorphism) unique type α, such that for every $\texttt{m} \in T$, $\vdash \texttt{m} : \alpha$ is derivable, and every $\theta \in \Theta$ can be obtained by α through a type substitution.*

α is usually called the **most general unifier** of Θ; for a set of terms T, we also directly call it $pt(T)$, the principal type of T; for Θ a set of types, we denote it by $\bigvee \Theta$. A proof for this fundamental lemma can be found in [6]; again the proof is constructive. Note that if we do not assume that the set of principal types of terms $\texttt{m} \in T$ is finite, then there is no upper bound on the length of types, and so there cannot be a finite common unifier. For convenience, we introduce an additional type $\top \notin A$, such that our types are the set $\{\top\} \cup Tp(A)$. If a set of types Θ does not have a common unifier, then we put $\bigvee(\Theta) = \top$.

This is of immediate importance for us, as it allows us both to speak of the principal type of a set of terms, as well as of the least upper bound of a set of types. From there we easily arrive at the greatest lower bound of two types α, β, which we denote by $\alpha \wedge \beta$, and which intuitively is the amount of structure which α and β share. Write $\alpha \leq \beta$, if there is a substitution π such that $\pi(\alpha) = \beta$. This is, up to isomorphism, a partial order. We now can simply define $\alpha \wedge \beta := \bigvee\{\gamma : \gamma \leq \alpha, \beta\}$. It is clear that the set $\{\gamma : \gamma \leq \alpha, \beta\}$ modulo isomorphism is finite, so the (finite) join exists in virtue of the above lemma. So $Tp(A)$ is lattice ordered up to isomorphism.

How does type structure behave wrt. concept structure? First of all, if $A \subseteq B$, then $pt(A) \leq pt(B)$. So the inclusion relation reflects type structure. This entails that $pt(A) \leq pt(B^{\triangleright\triangleleft})$. Stronger results are hard to obtain; for example, if we know $pt(A)$, there is nothing we can say in general about an upper bound for $pt(A^{\triangleright\triangleleft})$.

Fortunately, there is more we can say about the lattice order of concepts and type order. Define $\vee$ and $\wedge$ on concepts as usual. For a concept (A, B) over the term context, we put $pt_1(A, B) = pt(A)$, $pt_2(A, B) = pt(B)$.

Lemma 5 *For concepts $\mathcal{C}_1, \mathcal{C}_2$ of the term context, the following holds: (1) If $\mathcal{C}_1 \leq \mathcal{C}_2$, then $pt_1(\mathcal{C}_1) \leq pt_1(\mathcal{C}_2)$, and $pt_2(\mathcal{C}_2) \leq pt_2(\mathcal{C}_1)$. (2) $pt_1(\mathcal{C}_1 \wedge \mathcal{C}_2) \leq pt_1(\mathcal{C}_1) \wedge pt_1(\mathcal{C}_2)$, and (3) $pt_1(\mathcal{C}_1) \vee pt_1(\mathcal{C}_2) \leq pt_1(\mathcal{C}_1 \vee \mathcal{C}_2)$.*

Proof. The first claim is immediate by set inclusion. To see the second, consider that for every $\texttt{m} \in A_1 \cap A_2$, we must have $pt(\{\texttt{m}\}) \leq pt(A_1), pt(\{\texttt{m}\}) \leq pt(A_2)$ by set inclusion; and so $pt(\{\texttt{m}\}) \leq pt_1(C_1) \wedge pt_1(\mathcal{C}_2)$. To see the third claim, consider the following: we can easily show that $pt(A_1) \vee pt(A_2) = pt(A_1 \cup A_2)$. Then the claim follows from considering that $pt(A_1 \cup A_2) \leq pt((A_1 \cup A_2)^{\triangleright\triangleleft})$. $\Box$

Definition 6 *A term $\texttt{m}$ is a **left equalizer**, if we have $\vdash \texttt{m} : \theta_1 \to \alpha$, $\vdash \texttt{m} : \theta_2 \to \alpha$, and $\theta_1 \neq \theta_2$. $\texttt{m}$ is a **right equalizer**, if $\vdash \texttt{m} : \alpha_1$, $\vdash \texttt{m} : \alpha_2$, and $\alpha_1 \neq \alpha_2$.*

Easy examples of left equalizers are terms with vacuous abstraction; easy examples of right equalizers are terms which do not contain constants. A term which is both a left and right equalizer is $\lambda yx.x$. The following results are a bit tedious to obtain, yet not very significant; we therefore omit the proof.

Lemma 7 *Let $T \subseteq \mathtt{WTT}$, such that $pt(T) = \top$. Then each $\mathtt{m} \in T^{\triangleright}$ is a left equalizer.*

We can thus also speak of equalizer concepts. If we restrict our context to λI terms, we get a stronger result:

Lemma 8 *Let $\mathtt{m}$ be a left equalizer and λI-term, such that $\vdash_{\Sigma} \mathtt{m} : \theta_1 \to \alpha$ and $\vdash_{\Sigma} \mathtt{m} : \theta_2 \to \alpha$. Then both θ_1, θ_2 must be types inhabited by terms in $\mathtt{Tm}(\Lambda I(\Sigma))$, that is, there are terms $\mathtt{m}_i$, for which $\vdash_{\Sigma} \mathtt{m}_i : \theta_i$ is derivable for $i \in \{1, 2\}$ and $\mathtt{m}_i \in \mathtt{Tm}(\Lambda I(\Sigma))$.*

So when we restrict ourselves to ΛI, we have proper restrictions on the class of possible equalizers, in the general case we do not. For example, assume there is a set T of terms, and $pt(T) \neq \top$. Still, we might have $pt(T^{\triangleright}) = \top$. Conversely, from the fact that $pt(T) = \top$, it does not follow that $T^{\triangleright} = \emptyset$.

Of course, all general results of FCA also hold in this particular setting. For us, the question is not in how far is the type theoretic context interesting as a *particular* context, but rather: in how far can type theoretic contexts be used in order to *generalize* existing contexts? As is well-known, type theory is a very powerful tool; we will show this by way of example in formal language theory.

4 A Language-theoretic Context

4.1 Syntactic Concepts

Syntactic concept lattices form a particular case of formal concept lattices. In linguistics, they have been introduced in [9]. They were brought back to attention and enriched with residuation in [2], [3], as they turn out to be useful representations for language learning (for background on residuated lattices, see [5]). Syntactic concept are useful to describe distributional patterns of strings[4]. The most obvious way to do so is by partitioning strings/substrings into *equivalence classes*: we say that two strings w, v are equivalent in a language $L \subseteq T^*$, in symbols, $w \sim_L v$, iff for all $x, y \in T^*$, $xwy \in L \Leftrightarrow xvy \in L$.[5] The problem with equivalence classes is that they are too restrictive: a single word can ruin an equivalence class. In particular in linguistic applications, this is bad, because restrictions of datasets or some particular constructions might prevent us from having, say, an equivalence class of nouns. Syntactic concepts provide a somewhat less rigid notion of equivalence, which can be conceived of as equivalence restricted to a given set of string-contexts (not to be confused with contexts in the sense of FCA!).

[4] Or words, respectively, depending on whether we think of our language as a set of words or a set of strings of words.

[5] This defines the well-known Nerode-equivalence.

We now define our first language theoretic context. For $L \subseteq T^*$, $\mathfrak{B}_C(L) = (T^*, T^* \times T^*, I)$, where $(b, (a, c)) \in I$ iff $abc \in L$. This gives rise to polar maps $\triangleright : \wp(T^*) \to \wp(T^* \times T^*)$, and $\triangleleft : \wp(T^* \times T^*) \to \wp(T^*)$, where

1. for $S \subseteq T^*$, $S^\triangleright := \{(x, y) : \forall w \in S, xwy \in L\}$; and dually
2. for $C \subseteq T^* \times T^*$, $C^\triangleleft := \{x : \forall (v, w) \in C, vxw \in L\}$.

That is, a set of strings is mapped to the set of string contexts, in which all of its elements can occur. For a set of string contexts C, $C^\triangleleft$ can be thought of as an equivalence class with respect to the string contexts in C; but not in general: there might be elements in $C^\triangleleft$ which can occur in a string context $(v, w) \notin C$ (and conversely).

Definition 9 *A syntactic c-concept A is a pair, consisting of a set of strings, and a set of string contexts, written $\mathcal{C} = (S_\mathcal{C}, C_\mathcal{C})$, such that $S_\mathcal{C}^\triangleright = C_\mathcal{C}$ and $C_\mathcal{C}^\triangleleft = S_\mathcal{C}$. The* ***syntactic c-concept lattice*** *of a language L is defined as $\mathcal{L}(\mathfrak{B}_C(L)) := \langle \mathfrak{C}_L^C, \wedge, \vee, \top, \bot \rangle$, where $\mathfrak{C}_L^C$ is the set of syntactic c-concepts of L, and with all constants and connectors defined in the usual way.*

For example, given a language L, we have $(\epsilon, \epsilon)^\triangleleft = L$, as all and only the strings in L can occur in L in the string context (ϵ, ϵ); so L is a closed set of strings. We can give the syntactic concept lattice some more structure. We define a monoid structure on concepts as follows: for concepts $(S_1, C_1), (S_2, C_2)$, we define:

$$(2) \qquad (S_1, C_1) \circ (S_2, C_2) = ((S_1 S_2)^{\triangleright\triangleleft}, (S_1 S_2)^\triangleright),$$

where $S_1 S_2 = \{xy : x \in S_1, y \in S_2\}$. Obviously, the result is a concept. $'\circ'$ is associative on concepts, that is, for $X, Y, Z \in \mathfrak{B}$, $X \circ (Y \circ Z) = (X \circ Y) \circ Z$ (see [10] for discussion). It is easy to see that the neutral element of the monoid is $(\{\epsilon\}^{\triangleright\triangleleft}, \{\epsilon\}^\triangleright)$, and that the monoid structure respects the partial order of the lattice: For concepts $X, Y, Z, W \in \mathfrak{B}$, if $X \leq Y$, then $W \circ X \circ Z \leq W \circ Y \circ Z$.

We define a similar operation $\bullet$ for the string contexts of concepts: $(x, y) \bullet (w, z) = (xw, zy)$. This way, we still have $f \bullet (g \bullet h) = (f \bullet g) \bullet h$ for singleton string contexts f, g, h. The operation can be extended to sets in the natural way, preserving associativity. For example, $C \bullet (\epsilon, S) = \{(x, ay) : (x, y) \in C, a \in S\}$. We will use this as follows:

Definition 10 *Let $X = (S_X, C_X), Y = (S_Y, C_Y)$ be concepts. We define the right residual $X/Y := ((C_1 \bullet (\epsilon, S_Y))^\triangleleft, (C_1 \bullet (\epsilon, S_Y))^{\triangleleft\triangleright})$, and the left residual $Y \backslash X := ((C_1 \bullet (S_Y, \epsilon))^\triangleleft, (C_1 \bullet (S_Y, \epsilon))^{\triangleleft\triangleright})$.*

For the closed sets of strings S, T, define $S/T := \{w : \text{for all } v \in T, wv \in S\}$. We then have $S_X/S_Y = S_{X/Y}$. So residuals are unique and satisfy the following lemma:

Lemma 11 *For $X, Y, Z \in \mathfrak{C}_L^C$, we have $Y \leq X \backslash Z$ iff $X \circ Y \leq Z$ iff $X \leq Z/Y$.*

For a proof, see [2]. This shows that the syntactic concept lattice can be enriched to a residuated lattice (a residuated lattice is precisely a lattice with monoid structure and satisfying the law of residuation).

4.2 Problems and Limitations

Syntactic c-concepts form a very well-behaved structure, which even forms a complete class of models for the Full Lambek calculus, an important substructural logic (see [10]). A major limitation of these concepts is that our only objects are strings, and our only operation concatenation. To see why this is a restriction, consider the following. Let $L_1 \subseteq T^*$ be an arbitrary language, and put $L_2 := \{ww : w \in L_1\}$. Now, in the general case, L_1 will not be a closed concept of L_2, because there is no general string context in which all and only the words in L_1 can occur. The appropriate string context would have to be a *function*, as each $w \in L_1$ has the string context (ϵ, w). So there is a clear and simple generalization regarding the distribution of L_1 in L_2, but we cannot express it.

As a second example, consider $L_3 := \{a^{2^n} : n \in \mathbb{N}\}$. Here we have the following problem: for each $n \in \mathbb{N}$, a^n will have a distinguishing string context. Nonetheless, there is a very simple pattern in the language: taking a word of L_3, we just have to concatenate it with itself, and we get a new word in L_3. In our analysis, however, there are no concepts $\mathcal{C}_1, \mathcal{C}_2$, such that $\mathcal{C}_1 \circ (L_3, L_3^{\triangleright}) \circ \mathcal{C}_2 = (L_3, L_3^{\triangleright})$. So our concepts are uninformative on the pattern of this language. What we want to have in this case is a concept of *duplication*. We will remedy these shortcomings in what is to follow; we will need, however, some type-theoretic background.

5 Strings as λ-Terms

The following, type theoretic encoding of language theoretic entities has been developed in the framework on on **abstract categorial grammars** (introduced in [4]). We follow the standard presentation given in [7]. Given a finite alphabet T, a string $a_1 \dots a_n \in T^*$ over T can be represented by a λ term over the signature $\Sigma_T^{string} := (\{o\}, T, \phi)$, where for all $a \in T$, $\phi(a) = o \to o$; we call this a *string signature*. The term is linear and written as $/a_1 \dots a_n/ := \lambda x.a_1(\dots(a_n x)\dots)$. Obviously, the variable x has to be type o, in order to make the term typable. We then have, for every string $w \in T^*$, $\vdash_{\Sigma_T^{string}} /w/ : o \to o$.

Under this representation, string concatenation is not entirely trivial, and cannot be done by juxtaposition, as the result would not be typable. We can concatenate strings by the combinator $\mathbf{B} := \lambda xyz.x(yz)$, which concatenates its first argument to the left of its second argument, as can be easily checked.[6] We can also represent tuples of strings by terms. Let $/w_1/, \dots, /w_n/$ represent strings. Then a tuple of these strings is written as $/(w_1, \dots, w_n)/ := \lambda x.((\dots(x/w_1/)\dots)/w_n/)$. The type of x here depends on the size of the tuple. We define $\alpha \to_n \beta$ by $\alpha \to_0 \beta = \beta$, $\alpha \to_{n+1} \beta = \alpha \to (\alpha \to_n \beta)$. In general, for a term $\mathtt{m}$ encoding an n-tuple , we have $\vdash_{\Sigma_T^{string}} \mathtt{m} : ((o \to o) \to_n (\alpha))) \to \alpha$. So the types get larger with the size of tuples; the *order* of the term however remains invariantly 2.

We indicate how to manipulate tuple components separately. The function which concatenates the tuple components in their order is obtained as

[6] See [7] for more examples, also for what is to follow. A *combinator* is in general a function over functions.

follows: Given a tuple $/(w,v)/ = \lambda x.((x/w/)/v/$, we obtain $/wv/$ through application of the term: $\lambda x_1.x_1(\lambda x_2 y.\mathbf{B}x_2 y))$. We can also manipulate tuples to form new tuples: take again $/(v_1, w_1)/ = \lambda x.((x/v_1/)/w_1/)$; we want to convert it into a tuple $/(v_1v_2, w_1w_2)/ = \lambda x.((x/v_1v_2/)/w_1w_2/)$. This is done by the term $\lambda y x_1.y(\lambda x_2 x_3((x_1\mathbf{B}x_2v_2)\mathbf{B}x_3w_2))$. This term takes the tuple as argument and returns a tuple of the same type. If we abstract over the term $/(v_2, w_2)/$, this gives us a function which concatenates two 2-tuples componentwise.

However, the general componentwise concatenation of tuples of arbitrary size (considering strings as 1-tuples) cannot be effected by a typed λ-term. The reason is: if we do not fix an upper bound on tuple size, the types of tuples get higher and higher, and there is no finite upper bound. So there is no finite term which could have the appropriate type.[7] This means that in this setting, we must refrain from a notion of general concatenation of any type. This will however do little harm, as we will see.

6 Generalizing the Language-theoretic Context

Take a finite alphabet T, and fix a language $L_1 \subseteq T^*$. As we have seen in the last section, there is a bi-unique mapping $i : T^* \to \texttt{WTT}$ between strings in T^* and λ-terms of the signature Σ_T^{string}. Note that i is properly bi-unique and not up to $=_{\alpha\beta}$ equivalence; we map strings only onto their standard encoding, using a standard variable. We thus obtain $i[L_1] \subseteq \texttt{WTT}$, where $i[-]$ is the pointwise extension of i to sets. We close $i[L_1]$ under $=_{\alpha\beta}$, and obtain $L := \{\texttt{m} : \text{there is } \texttt{n} \in i[L] : \texttt{n} =_{\alpha\beta} \texttt{m}\}$. This is the language we are working with, the type theoretic counterpart of L_1. In the sequel, for any $M \subseteq T^*$, we will denote the closure of $i[M]$ under $=_{\alpha\beta}$ by M^λ; so we have $L = (L_1)^\lambda$.

We now define a context $\mathfrak{B}_T(L) = (\mathcal{G}, \mathcal{M}, I)$, where $\mathcal{G} = \mathcal{M} = \texttt{Tm}_c(\Lambda(\Sigma_T^{string}))$, that is the set of closed terms over the signature Σ_T^{string}; and for $\texttt{m}, \texttt{n} \in \texttt{Tm}_c(\Lambda(\Sigma_T^{string}))$, we have $(\texttt{m}, \texttt{n}) \in I$ iff $\texttt{nm} \in L$. So for S a set of terms, we have $S^\triangleright := \{t : \forall s \in S : ts \in L\}$, and $S^\triangleleft := \{t : \forall s \in S : st \in L\}$.

Definition 12 *A t-concept is a concept (S, T) over the context $\mathfrak{B}_T(L)$, where $S = T^\triangleleft$, $T = S^\triangleright$. The **syntactic t-concept lattice** of a language L is defined as $\mathcal{L}_T(L) := \mathcal{L}(\mathfrak{B}_T(L)) = \langle \mathfrak{C}_L^T, \wedge, \vee, \top, \bot\rangle$, where $\mathfrak{C}_L^T$ is the set of syntactic t-concepts of L, and with all constants and connectors defined in the usual way.*

What we are still missing is an operator which allows us to define fusion and residuation. Recall that for terms, our primitive objects, juxtaposition is interpreted as function application. We extend this interpretation to sets of terms: for $S_1, S_2 \subseteq \texttt{Tm}_c(\Lambda(\Sigma_T^{string}))$, we define $S_1S_2 := \{\texttt{mn} : \texttt{m} \in S, \texttt{n} \in T\}$. Next, for t-concepts $(S_1, T_1), (S_2, T_2)$, we simply put $(S_1, T_1) \circ (S_2, T_2) := ((S_1S_2)^{\triangleright\triangleleft}, (S_1S_2)^\triangleright)$.

[7] On the other side, once we fix an upper bound k to tuple size, it is easy to see how to define $\circ$ as λ term: for $i \leq k$, we simply encode all tuples as k-tuples with all jth components, $i < j$, containing the empty string. Then $\circ$ is simply componentwise concatenation of k-tuples, which is λ-definable, as we have seen.

That is, as before we use the closure of concatenation of extents to define $\circ$. But there is an important restriction: concatenation of terms is *not* associative. Consequently, the operation $\circ$ is not associative on concepts, we have, for concepts $M, N, O \in \mathfrak{C}_L^T$, $(M \circ N) \circ O \neq M \circ (N \circ O)$. For example, $M \circ N$ might be $\top$, because MN contains a term $\mathtt{mn} \notin \mathtt{WTT}$, and consequently we have $\top \circ O = \top$. Still, $M \circ (N \circ O)$ might be well-typed. So the structure of $(\mathfrak{B}, \circ)$ is not a monoid, but rather a groupoid. We furthermore have a left identity element 1_l, such that for every concept S, $1_l \circ S = S$. This is the concept of the identity function $(\{\lambda x.x\}^{\triangleright\triangleleft}, \{\lambda x.x\}^{\triangleright})$. (By the way, the identity function is also the encoding of the empty string $/\epsilon/$). There is no general right identity, though: for assume we have a term $\mathtt{m} : \alpha$ for a constant atomic type α; then there is no term $\mathtt{n}$ such that $\mathtt{mn}$ can be typed. Consequently, no $\mathtt{n}$ can be the right identity for $\mathtt{m}$.

What are the residuals in this structure? Given the fusion operator, they are already implicitly defined by the law of residuation $O \leq M/N \Leftrightarrow O \circ N \leq M \Leftrightarrow N \leq O \backslash M$; what we have to show that they exist and are unique. In the sequel we will use residuals both on sets of terms and on concepts; this can be done without any harm, as the extent order and the concept order are isomorphic. To see more clearly what residuation means in our context, note that for $S \subseteq \mathtt{Tm}_c(\Lambda(\Sigma_T^{string}))$, we have $S^{\triangleright} := L/S$; because $S^{\triangleright}$ is the set of all terms $\mathtt{m}$, such that for all $\mathtt{n} \in S$, $\mathtt{mn} \in L$. Dually, we have $S^{\triangleleft} := S \backslash L$. Consequently, we have $S^{\triangleright\triangleleft} = (L/S)\backslash L$, and dually, we get $S^{\triangleleft\triangleright} = L/(S\backslash L)$. So we see that the polar maps of our Galois connection form a particular case of the residuals, or conversely, the residuals form a generalization of the polar maps. The closure operators are equivalent to a particular case of what is known as *type raising*. More generally, we can explicitly define residuals over a ternary relation: put $(\mathtt{m}, \mathtt{n}, \mathtt{o}) \in R$ if and only if $\mathtt{mn} =_{\alpha\beta} \mathtt{o}$. Then we define

1. $O/N := \{\mathtt{m} : \forall \mathtt{n} \in N, \exists \mathtt{o} \in O : (\mathtt{m}, \mathtt{n}, \mathtt{o}) \in R\}$; dually:
2. $M\backslash O := \{\mathtt{n} : \forall \mathtt{m} \in M, \exists \mathtt{o} \in O : (\mathtt{m}, \mathtt{n}, \mathtt{o}) \in R\}$.

As is easy to see, $M^{\triangleright} := \{\mathtt{n} : \forall \mathtt{m} \in M, \exists \mathtt{o} \in L : (\mathtt{m}, \mathtt{n}, \mathtt{o}) \in R\}$; and $M^{\triangleleft} := \{\mathtt{n} : \forall \mathtt{m} \in M, \exists o \in L : (\mathtt{m}, \mathtt{n}, \mathtt{o}) \in R\}$. This way, we explicitly define residuals for sets of terms. Given this, it easily follows that residuals also exist and are unique for concepts: $(S_1, T_1)/(S_2, T_2) = ((S_1/S_2), (S_1/S_2)^{\triangleright})$.

So residuals allow us to form the closure not only with respect to L, but with respect to any other concept. This provides us with a much more fine-grained access to the hierarchical structure of languages. On the negative side, the $\circ$ operation and residuals do not tell us anything about directionality of concatenation on the string level. This however is unsurprising, as our treatment of strings as λ-terms serves precisely the purpose of abstracting away from this: concatenation is done by terms automatically, and we need no longer care for this. Obviously t-concepts provide a vast generalization of c-concepts. An immediate question is whether this extension is conservative, in the sense that each c-closed set is also t-closed. This is generally wrong, but holds with some restrictions:

Theorem 13 *Let $M, L \subseteq T^*$; let M^{λ}, L^{λ} be their type theoretic counterpart in the signature Σ_T^{string}. If $M = M^{\triangleright\triangleleft}$ is closed wrt. the language theoretic context*

$\mathfrak{B}_C(L)$, then we have $M^\lambda = (M^\lambda)^{\triangleright\triangleleft} \cap (T^*)^\lambda$, where $(M^\lambda)^{\triangleright\triangleleft}$ is closed wrt. the type theoretic context $\mathfrak{B}_T(L^\lambda)$.

Proof. Let M be c-closed; every string context $(w, v) \in M^\triangleright$ corresponds to a function of the form $\lambda x.\mathbf{B}(\mathbf{B}/w/x)/v/$, which takes a term $/u/$ as argument, concatenating it with a $/w/$ to its left and $/v/$ to its right, resulting in a term $/wuv/$. Call the set of these functions $(M^\lambda)^\blacktriangleright$. We now take $(M^\lambda)^{\blacktriangleright\triangleleft}$. Obviously we have $M^\lambda \subseteq (M^\lambda)^{\blacktriangleright\triangleleft}$. We show that $M^\lambda \supseteq (M^\lambda)^{\blacktriangleright\triangleleft} \cap (T^*)^\lambda$: if we have, for $w \in T^*$, $w \notin M$, but $/w/ \in (M^\lambda)^{\blacktriangleright\triangleleft} \cap (T^*)^\lambda$, then we have $i^{-1}(/w/) \in M^{\triangleright\triangleleft}$, because each type context in $(M^\lambda)^\blacktriangleright$ corresponds to a string context in $M^\triangleright$. This is a contradiction, as M is closed under $[-]^{\triangleright\triangleleft}$.

So we have $M^\lambda = (M^\lambda)^{\blacktriangleright\triangleleft} \cap (T^*)^\lambda$, and $(M^\lambda)^{\blacktriangleright\triangleleft}$ is a closed set. Furthermore, as $(M^\lambda)^\blacktriangleright \subseteq (M^\lambda)^\triangleright$, we have (by the laws of Galois connections) $(M^\lambda)^{\blacktriangleright\triangleleft} \supseteq (M^\lambda)^{\triangleright\triangleleft}$. So we get $M^\lambda \supseteq (M^\lambda)^{\triangleright\triangleleft} \cap (T^*)^\lambda$. To see that $M^\lambda \subseteq (M^\lambda)^{\triangleright\triangleleft} \cap (T^*)^\lambda$, consider that as $M \subseteq T^*$, we have $M^\lambda \subseteq (T^*)^\lambda$; furthermore, $M^\lambda \subseteq (M^\lambda)^{\triangleright\triangleleft}$. Therefore, $M^\lambda \subseteq (M^\lambda)^{\triangleright\triangleleft} \cap (T^*)^\lambda$. This completes the proof. □

As expected, the converse does not hold, not even for terms which encode strings. In this sense t-concepts yield a proper generalization of c-concepts. This however does not obtain for the extension of the lattice with fusion and residuals: fusion in the t-concept lattice is completely incomparable to fusion in the c-concept lattice of a language.

7 Conclusion and Possible Restrictions

One main objection to our type theoretic approach to language might be that we produce many concepts to which we might not be able to assign any intuitive meaning, and which tell us very little about the language in question. We easily see what is meant by the concept of a term $/w/$ in a language L. We can also make perfectly sense of the concept duplication. It is less easy to see what is meant by the concept of the term $\mathbf{B}$, which we discussed above. What can the the distribution of such a concept in a language tell us about the language?[8] So we do not have a problem with the formalism in the first place, but with its interpretation.

Therefore, it might be reasonable to restrict our approach. We propose here two main restrictions: First, as we already mentioned, we might restrict the universe of terms $\mathtt{Tm}_c(\Lambda(\Sigma_T^{string}))$ to λI terms. A language-theoretic argument for this point is that we are interested in the distributional structure of languages. Vacuous abstraction, as yielding constant functions, allows us to delete arguments or certain parts thereof. This seems to us an "unlinguistic" procedure, as we cannot say we talk about the distribution of an object if we allow to delete parts of it. A further restriction to linear λ-terms, on the other side, does not

[8] What is less unclear is its meaning as intent rather than extent: apart from some additional technical difficulties, it must take two arguments which, when concatenated, give a term in L.

seem to be desirable, as then we get the same problems with our toy language $\{a^{2^n} : n \in \mathbb{N}\}$ as before.

A second restriction which might be reasonable is the restriction of **type order**. As we have seen, types of second order allow us to yield all strings and tuples of strings. If we restrict only $\mathcal{G}$ to second order types, we will have all functions from second order types to second order types in $\mathcal{M}$. This seems to us a very reasonable restriction, the consequence of which we cannot discuss here for reasons of space.

In conclusion, there are many options in further pursuing our approach, and at this point it is unclear which direction is the most promising. But in either way our approach might provide some contribution to the old problem of learning infinite languages from the distributional structure of a finite fragment thereof.

References

1. Henk Barendregt. *The Lambda Calculus. Its Syntax and Semantics.* Number 103 in Studies in Logic. Elsevier, Amsterdam, 2 edition, 1985.
2. Alexander Clark. A learnable representation for syntax using residuated lattices. In Philippe de Groote, Markus Egg, and Laura Kallmeyer, editors, *Proceedings of the 14th Conference on Formal Grammar*, volume 5591 of *Lecture Notes in Computer Science*, pages 183–198. Springer, 2009.
3. Alexander Clark. Learning context free grammars with the syntactic concept lattice. In José M. Sempere and Pedro García, editors, *10th International Colloquium on Grammatical Inference*, volume 6339 of *Lecture Notes in Computer Science*, pages 38–51. Springer, 2010.
4. Philippe de Groote. Towards Abstract Categorial Grammars. In *Association for Computational Linguistics, 39th Annual Meeting and 10th Conference of the European Chapter*, pages 148–155, Toulouse, 2001.
5. Nikolaos Galatos, Peter Jipsen, Tomasz Kowalski, and Hiroakira Ono. *Residuated Lattices: An Algebraic Glimpse at Substructural Logics.* Elsevier, 2007.
6. J.Roger Hindley. *Basic Simple Type Theory.* Number 42 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, Cambridge, 2008.
7. Makoto Kanazawa. Second-order Abstract Categorial Grammars as Hyperedge Replacement Grammars. *Journal of Logic, Language and Information*, 19(2):137–161, 2010.
8. Ian D. Peake, Ian E. Thomas, and Heinz W. Schmidt. Typed formal concept analysis. In Karl Erich Wolff, Sebastian Rudolph, and Sebastien Ferre, editors, *Contributions to ICFCA 2009 (Supplementary Proceedings), 7th International Conference on Formal Concept Analysis*, Darmstadt, 2009. Verlag Allgemeine Wissenschaft.
9. A. Sestier. Contributions à une théorie ensembliste des classifications linguistiques. (Contributions to a set–theoretical theory of classifications). In *Actes du Ier Congrès de l'AFCAL*, pages 293–305, Grenoble, 1960.
10. Christian Wurm. Completeness of Full Lambek calculus for syntactic concept lattices. In *Proceedings of the 17th Conference on Formal Grammar, Springer Lecture Notes in Computer Science*, in press.

Distributed Closed Pattern Mining in Multi-Relational Data based on Iceberg Query Lattices: Some Preliminary Results

Hirohisa Seki[*] and Sho-ich Tanimoto

Dept. of Computer Science, Nagoya Inst. of Technology,
Showa-ku, Nagoya 466-8555, Japan
seki@nitech.ac.jp

Abstract. We study the problem of mining frequent closed patterns in multi-relational databases in a distributed environment. In multi-relational data mining (MRDM), relational patterns involve multiple relations from a relational database, and they are typically represented in datalog language (a class of first order logic). Our approach is based on the notion of *iceberg query lattices*, a formulation of MRDM in terms of formal concept analysis (FCA), and we apply it to a distributed mining setting. We assume that a database considered contains a special predicate called *key*, which determines the entities of interest and what is to be counted, and that each datalog query contains an atom key, where variables in a query are linked to a given target object corresponding to the key. We show that the iceberg query lattice in this case can be defined similarly in the literature. Next, given two local databases (*horizontal* partitions) and their sets of closed patterns (concepts), we show that the subposition operator, which constructs a global Galois (concept) lattice from the direct product of two lattices studied in the literature, can be utilized to generate the set of closed patterns in the global database. The correctness of our algorithm is shown, and some preliminary experimental results using a MapReduce framework are also given.

1 Introduction

Multi-relational data mining (MRDM) has been extensively studied for more than a decade (e.g., [5, 6] and references therein). The research topics discussed in the conventional data mining have been considered in this more expressive framework of MRDM, where data and patterns (or queries) are represented in the form of logical formulae such as Datalog (a class of first order logic). In contrast to the traditional data mining dealing with rather simple patterns such as itemsets, the expressive formalism of MRDM allows us to use more complex and structured data in a uniform way, including trees and graphs in particular, and multi-relational patterns in general.

[*] This work was partially supported by JSPS Grant-in-Aid for Scientific Research (C) 24500171 and the Kayamori Foundation of Information Science Advancement.

On the other hand, Formal Concept Analysis (FCA) has been developed as a field of applied mathematics based on a clear mathematization of the notions of concept and conceptual hierarchy [7]. It has attracted much interest from various application areas including, among others, data mining, knowledge acquisition and software engineering (e.g., [8]).

Stumme [20] has proposed the notion of *iceberg query lattices*, which combines the notions of the above two fields, i.e., MRDM and FCA; Iceberg query lattices combine the notion of frequent datalog queries in MRDM with iceberg concept lattices (or *frequent closed itemsets*) in FCA. Then, it has been shown that we can apply the "full arsenal" of FCA-based methods to frequent queries, thereby allowing us to mine and visualize relational association rules. Condensed representations such as closed patterns and free patterns in MRDM have been also studied in *c-armr* [4], and in RelLCM2 [9].

In this paper, we study the problem of mining closed queries in multi-relational data based on these precursors. We apply the notion of iceberg query lattices to a distributed mining setting. The assumption that a given dataset is distributed and stored in different sites is reasonable, because we will not be able to move local datasets into a centralized site due to too much data size and/or privacy concerns. We also assume that a database considered contains a special predicate called *key* (e.g., [3, 4]), and that each datalog query is supposed to contain an atom key, where variables in a query are *linked* to a given target object corresponding to the key. Using an key atom, we can define the notion of the frequency of a datalog query, since the key atom determines the entities of interest and what is to be counted. We show that the iceberg query lattice in this case can be defined similarly in the literature. Next, given two local databases (*horizontal* partitions) and their sets of closed queries (concepts), we show that the we can construct the set of closed queries in the global database, by using *subposition* operator [7, 23], which constructs a global Galois (concept) lattice from the direct product of two lattices. We also present some preliminary experimental results using a distributed framework of MapReduce [2].

The organization of the rest of this paper is as follows. After summarizing some basic notations and definitions in FCA in Sect. 2, we reconsider the notion of iceberg query lattices with key in Sect. 3. We then explain our approach to distributed closed query mining in MRDB in Sect. 4. In Section 5, we show the effectiveness of our method by some preliminary experimental results. Finally, we give a summary of this work in Section 6.

2 Preliminaries: Formal Concept Analysis

We assume that the reader is familiar with the basic notions of Formal Concept Analysis (FCA), which are found in [7]. However, we recall some of the important definitions and notations.

Definition 1. A *(formal) context* $\mathcal{K} = (O, A, I)$ consists of a set O of objects, a set A of attributes, and a binary relation $I \subseteq O \times A$.

The mapping $f : \mathcal{P}(O) \rightarrow \mathcal{P}(A)$ is given by $f(X) = \{a \in A \mid \forall o \in X : (o, a) \in I\}$. The mapping $g : \mathcal{P}(A) \rightarrow \mathcal{P}(O)$ is given by $g(Y) = \{o \in O \mid \forall a \in Y : (o, a) \in I\}$.

If it is clear from the context whether f or g is meant, then we abbreviate both $f(\cdot)$ and $g(\cdot)$ just by $'$. In particular, Y'' stands for $f(g(Y))$.

A *(formal) concept* is a pair (X, Y) with $X \subseteq O, Y \subseteq A,\ X' = Y$, and $Y' = X$. X is called *extent*, and Y is called *intent* of the concept. The set $\mathcal{C}_{\mathcal{K}}$ of all concepts of $\mathcal{K}$ together with the partial order $(X_1, Y_1) \leq (X_2, Y_2) \leftrightarrow X_1 \subseteq X_2$ (which is equivalent to $Y_1 \supseteq Y_2$) is called the *concept lattice* of $\mathcal{K}$. □

In FCA [7], a set of context-oriented operators has been studied, including *apposition/subposition* operators, and they are extensively studied by Valtchev and Missaoui [23, 24]. The following definitions and lemma are due to [23].

Definition 2. Let $\mathcal{K}_1 = (O_1, A, I_1)$ and $\mathcal{K}_2 = (O_2, A, I_2)$ be two contexts with the same set of attributes A. Then the context $\mathcal{K} = (O_1 \uplus O_2, A, I_1 \uplus I_2)$ is called the *subposition* of $\mathcal{K}_1$ and $\mathcal{K}_2$, denoted by $\mathcal{K} = \frac{\mathcal{K}_1}{\mathcal{K}_2}$.

Usually, the extent of $\mathcal{K}$ is set to the disjoint union (denoted by $\uplus$) of the involved context extents, and this constraint is suitable for our current study.

Let $\mathcal{K}_i$ $(i = 1, 2)$ be a context, and $\mathcal{L}_i$ the corresponding lattice. The direct product of a pair of lattices $\mathcal{L}_1$ and $\mathcal{L}_2$, denoted by $\mathcal{L}_\times = \mathcal{L}_1 \times \mathcal{L}_2$, is itself a lattice $\mathcal{L}_\times = \langle \mathcal{C}_{\mathcal{K}_\times}, \leq_\times \rangle$, where $\mathcal{C}_{\mathcal{K}_\times} = \mathcal{C}_{\mathcal{K}_1} \times \mathcal{C}_{\mathcal{K}_2}$, and $(c_1, c_2) \leq_\times (\bar{c}_1, \bar{c}_2) \Leftrightarrow c_1 \ \leq_{\mathcal{L}_1} \ \bar{c}_1$ and $c_2 \ \leq_{\mathcal{L}_2} \ \bar{c}_2$.

Any concept of $\mathcal{L}$ can be projected upon the concept lattice, $\mathcal{L}_1$ $(\mathcal{L}_2)$ by restricting its extent to the set of "visible" objects, e.g., those in O_1 (O_2), respectively. The resulting mapping constitutes an order homomorphism between $\mathcal{L}$ and the direct product [7].

Definition 3. The function $\varphi : \mathcal{C}_{\mathcal{K}} \rightarrow \mathcal{C}_{\mathcal{K}_\times}$ maps a concept from the global lattice into a pair of concepts of the partial lattices by splitting its extent over the partial context object sets O_1 and O_2:

$$\varphi((X, Y)) = ((X \cap O_1, (X \cap O_1)'),\ (X \cap O_2, (X \cap O_2)')).$$

From the above definitions, we have the following property [23]:

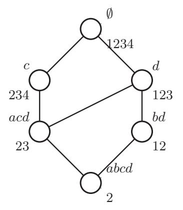
Lemma 1. [23] For any global concept $c = (X, Y)$ and its image $\varphi(c) = ((X_1, Y_1), (X_2, Y_2))$, it holds that $X = X_1 \cup X_2$ and $Y = Y_1 \cap Y_2$. □

Example 1. Consider a context $\mathcal{K}$ in Fig. 1 (upper left). The concept lattice $\mathcal{C}_{\mathcal{K}}$ derived from $\mathcal{K}$ is shown in the right. Let $\mathcal{K}_1, \mathcal{K}_2$ (lower right of the figure) be a horizontal decomposition of $\mathcal{K}$, where $O_1 = \{1, 2\}$ and $O_2 = \{3, 4\}$. Then, $\mathcal{K}$ is the subposition of $\mathcal{K}_1$ and $\mathcal{K}_2$, i.e., $\mathcal{K} = \frac{\mathcal{K}_1}{\mathcal{K}_2}$. The concept lattices $\mathcal{C}_{\mathcal{K}_1}$ $(\mathcal{C}_{\mathcal{K}_2})$ derived from $\mathcal{K}_1$ $(\mathcal{K}_2)$ are shown in the right, respectively.

Consider a global concept $c = (123, d)$ in $\mathcal{C}_{\mathcal{K}}$. Then, $\varphi(c) = ((12, bd), (3, acd))$, and we have from Lemma 1 that $\{123\} = \{12\} \cup \{3\}$ and $\{d\} = \{bd\} \cap \{acd\}$. □

$\mathcal{K}$

	a	b	c	d
1		×		×
2	×	×	×	×
3	×		×	×
4			×	



$\mathcal{K}_1$

	a	b	c	d
1		×		×
2	×	×	×	×

$\mathcal{K}_2$

	a	b	c	d
3	×		×	×
4			×	

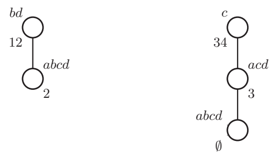


Fig. 1. Upper: A Context $\mathcal{K}$ and the Hasse diagram of the concept lattice derived from $\mathcal{K}$. Lower: A horizontal decomposition $\mathcal{K}_1, \mathcal{K}_2$ of $\mathcal{K}$, and the Hasse diagrams of the concept lattices derived from $\mathcal{K}_1, \mathcal{K}_2$.

FCA provides a framework for frequent itemset mining (FIM), where the intent of a concept corresponds to a closed itemset. The subposition operator will be readily used for mining frequent closed itemsets (FCIs) in a global transaction database $\mathcal{D}$ from the local FCIs from two disjoint (horizontal) partitions $\mathcal{D}_1$ and $\mathcal{D}_2$, provided that we mine all the partitions with an (absolute) support being set to 1, i.e. when we consider as frequent any itemset which occur at least once in $\mathcal{D}$. In fact, Lucchese et al. [15] show the following property:

Theorem 1 (Lucchese et al. [15]). Let $\mathcal{D}$ be transaction database, and $\mathcal{D}_1$, $\mathcal{D}_2$ two disjoint (horizontal) partitions of $\mathcal{D}$. Let $\mathcal{C}$ be the set of FCIs of $\mathcal{D}$, and $\mathcal{C}_1$ ($\mathcal{C}_2$) the set of local FCIs of $\mathcal{D}_1$ ($\mathcal{D}_2$), respectively. Then, $\mathcal{C}$ is computed from $\mathcal{C}_1$ and $\mathcal{C}_2$ as $\mathcal{C} = (\mathcal{C}_1 \cup \mathcal{C}_2) \cup \{C_1 \cap C_2 \mid (C_1, C_2) \in (\mathcal{C}_1 \times \mathcal{C}_2)\}$. □

Namely, $\mathcal{C}$ is obtained by collecting the closed itemsets contained in $\mathcal{C}_1$ and $\mathcal{C}_2$, and intersecting them to obtain further ones. It is easy to see that this exactly corresponds to Lemma 1 based on the subposition operator. In the following, we will apply the subposition operator to a more expressive framework of MRDM.

3 Iceberg Query Lattices in Multi-Relational DM

3.1 Multi-Relational Data Mining

In the task of frequent pattern mining in multi-relational databases, we assume that we have a given database **r**, a language of patterns, and a notion of frequency which measures how often a pattern occurs in the database. We use Datalog

Customer
key
allen
carol
diana
fred

Parent	
SR.	*JR.*
allen	bill
allen	jim
carol	bill
diana	eve
fred	eve
fred	hera

Buys	
key	*item*
allen	pizza
carol	pizza
diana	cake
fred	cake

Male
person
bill
jim

Female
person
eve
hera

Fig. 2. An Example of Datalog Database **r** with customer relation as a key

to represent data and patterns. We assume some familiarity with the notions of logic programming (e.g., [14, 16]), although we introduce some notions and terminology in the following.

An *atom* (or *literal*) is an expression of the form $p(t_1, \ldots .t_n)$, where p is a *predicate* (or *relation*) of arity n, denoted by p/n, and each t_i is a *term*, i.e., a constant or a variable.

A substitution $\theta = \{X_1/t_1, \ldots, X_n/t_n\}$ is an assignment of terms to variables. The result of applying a substitution θ to an expression E is the expression $E\theta$, where all occurrences of variables V_i have been simultaneously replaced by the corresponding terms t_i in θ. The set of variables occurring in E is denoted by $Var(E)$.

A *pattern* is expressed as a conjunction of atoms (literals) $l_1 \wedge \cdots \wedge l_n$, denoted simply by $l_1, \ldots, l_n$. A pattern is sometimes called a *query*. Let C be a pattern (i.e., a conjunction) and θ a substitution of $Var(C)$. When $C\theta$ is logically entailed by a database $\mathbf{r}$, we write it by $\mathbf{r} \models C\theta$. Let $answerset(C, \mathbf{r})$ be the set of substitutions satisfying $\mathbf{r} \models C\theta$. We will represent conjunctions in list notation, i.e., $[l_1, \ldots, l_n]$. For a conjunction C and an atom p, we denote by $[C, p]$ the conjunction that results from adding p after the last element of C.

In multi-relational data mining, one of predicates is often specified as a *key* (or *target*), which determines the entities of interest and what is to be counted.

Example 2. Let $\mathbf{r}$ be a multi-relational DB in Fig. 2, which consists of five relations, including Customer, Parent, Buys and so on. For each relation, we introduce a corresponding predicate, e.g., *customer* for relation Customer.

Let P be a pattern of the form: $customer(X), parent(X, Y), buys(X, pizza)$. $P\theta$ is logically entailed by $\mathbf{r}$, if there exists a tuple (a_1, a_2) such that $a_1 \in$ Customer, $(a_1, a_2) \in$ Parent, and $(a_1, pizza) \in$ Buys. Then, $answerset(P, \mathbf{r}) = \{\{X/allen, Y/bill\}, \{X/allen, Y/jim\}, \{X/carol, Y/bill\}\}$. □

As explained in Sect. 1, in a typical task of MRDM, a user is usually expected to specify a special predicate *key* (or *target*) (e.g., [3, 4]). The key is an atom which determines the entities of interest and what is to be counted. The key (target) is thus to be present in all patterns considered. In Example 2, the key is predicate *customer*.

A pattern containing a key is not always meaningful to be mined. For example, let $C = [customer(X), parent(X, Y), buys(Z, pizza)]$ be a conjunction in Example 2. Variable Z in C is not *linked* to variable X in key atom $customer(X)$; an object represented by Z will have nothing to do with key object X. It will be inappropriate to consider such a conjunction as an intended pattern to mine. In ILP, the following notion of *linked literals* [10] is a standard one to specify the so-called *language bias*.

Definition 4 (Linked Literal). [10] Let $key(X)$ be a key atom and l a literal. l is said to be *linked* to $key(X)$, if either $X \in Var(l)$ or there exists a literal l_1 such that l is linked to $key(X)$ and $Var(l_1) \cap Var(l) \neq \emptyset$. □

Given a database $\mathbf{r}$ and a key atom $key(X)$, we assume that there are predefined finite sets of predicate (resp. variables; resp. constant symbols), and that, for each literal l in a conjunction C, it is constructed using the predefined sets. Moreover, each pattern C of conjunctions to be mined satisfies the following conditions: $key(X) \in C$ and, for each $l \in C, l$ is linked to $key(X)$. In the following, we denote by $\mathcal{Q}$ the set of queries (or patterns) satisfying the above bias condition.

Let $\mathbf{r}$ be a database and Q be a query containing a key atom $key(X)$. Then, the *support* (or *frequency*) of C, denoted by $supp(Q, \mathbf{r}, key)$, is defined as:

$$supp(Q, \mathbf{r}, key) = \frac{|\{\theta_{key} \mid \theta \in answerset(Q, \mathbf{r})\}|}{|answerset(key(X), \mathbf{r})|},$$

where θ_{key} is the *restriction* of $\theta = \{X/t, \dots\}$ w. r. t. $key(X)$, defined by $\theta_{key} = \{X/t\}$ for some term t. The numerator in the above formula is called the *support count* (or *absolute support*). Q is said to be *frequent*, if $supp(Q, \mathbf{r}, key)$ is no less than some user defined threshold min_sup.

3.2 Iceberg Query Lattices with Key

We now consider the notion of a formal context in MRDM, following [20].

Definition 5. [20] Let $\mathbf{r}$ be a datalog database and $\mathcal{Q}$ a set of datalog queries. The *formal context associated to* $\mathbf{r}$ and $\mathcal{Q}$ is defined by $\mathcal{K}_{\mathbf{r},\mathcal{Q}} = (O_{\mathbf{r},\mathcal{Q}}, A_{\mathbf{r},\mathcal{Q}}, I_{\mathbf{r},\mathcal{Q}})$, where $O_{\mathbf{r},\mathcal{Q}} = \{\theta \mid \theta$ is a grounding substitution for all $Q \in \mathcal{Q}\}$, and $A_{\mathbf{r},\mathcal{Q}} = \mathcal{Q}$, and $(\theta, Q) \in I_{\mathbf{r},\mathcal{Q}}$ if and only if $\theta \in answerset(Q, \mathbf{r})$. □

Each $\theta \in answerset(Q, \mathbf{r})$ is often called an *occurrence* of Q in $\mathbf{r}$. We denote by $\mathcal{O}(Q; \mathbf{r})$ the set of the occurrences of Q in $\mathbf{r}$, namely, $\mathcal{O}(Q; \mathbf{r}) = answerset(Q, \mathbf{r})$.

From this formal context, we can define the concept lattice the same way as in [20]. We first introduce an equivalence relation $\sim_{\mathbf{r}}$ on the set of queries: Two queries Q_1 and Q_2 are said to be *equivalent* with respect to database $\mathbf{r}$ if and only if $answerset(Q_1, \mathbf{r}) = answerset(Q_2, \mathbf{r})$.

Definition 6 (Closed Query). Let $\mathbf{r}$ be a datalog database and $\sim_{\mathbf{r}}$ the equivalence relation on a set of datalog queries $\mathcal{Q}$. A query (or pattern) Q is said to be *closed* (w.r. t. $\mathbf{r}$ and $\mathcal{Q}$), iff Q is the most specific query among the equivalence class to which it belongs: $\{Q_1 \in \mathcal{Q} \mid Q \sim_{\mathbf{r}} Q_1\}$. □

For any query Q_1, its *closure* is a closed query Q such that Q is the most specific query among $\{Q \in \mathcal{Q} \mid Q \sim_{\mathbf{r}} Q_1\}$. Since it uniquely exists, we denote it by $Clo(Q_1; \mathbf{r})$. Note that $Var(Q_1) = Var(Clo(Q_1; \mathbf{r}))$ by definition. We refer to this as the *range-restricted* condition here.

Stumme [20] showed that the set of frequent closed queries forms a lattice. In our framework, it is necessary to take our bias condition into consideration. To do that, we employ the well-known notion of the most specific generalization (or *least generalization*) [18, 16].

For queries Q_1 and Q_2, we denote by $lg(Q_1, Q_2)$ the least generalization of Q_1 and Q_2. Moreover, the *join* of Q_1 and Q_2, denoted by $Q_1 \vee Q_2$, is defined as: $Q_1 \vee Q_2 = lg(Q_1, Q_2)|_{\mathcal{Q}}$, where, for a query Q, $Q|_{\mathcal{Q}}$ is the *restriction* of Q to $\mathcal{Q}$, defined by a conjunction consisting of every literal l in Q which is linked to $key(X)$, i.e., deleting every literal in Q not linked to $key(X)$.

Definition 7. [20] Let $\mathbf{r}$ be a datalog database and $\mathcal{Q}$ a set of datalog queries. The *iceberg query lattice associated to* $\mathbf{r}$ and $\mathcal{Q}$ for *minsupp* $\in [0, 1]$ is defined as: $\mathcal{C}_{\mathbf{r},\,\mathcal{Q}} = (\{Q \in \mathcal{Q} \mid Q \text{ is closed w.r.t. } \mathbf{r} \text{ and } \mathcal{Q}, \text{ and } Q \text{ is frequent}\}, \models)$, where $\models$ is the usual logical implication. □

Theorem 2. Let $\mathbf{r}$ be a datalog database and $\mathcal{Q}$ a set of datalog queries where all queries contain an atom *key* and they are linked. Then, $\mathcal{C}_{\mathbf{r},\,\mathcal{Q}}$ is a $\vee$-semi-lattice.

Proof. (Sketch) Let Q_1, Q_2 be frequent closed queries in $\mathcal{Q}$. Then, it is easy to see that their least generalization $lg(Q_1, Q_2)$ is closed and frequent. However, it might not be linked to $key(X)$. For example, consider that Q_1 (Q_2) is of the form: $Q_1 = key(X), p(X, Y), m(Y)$ ($Q_2 = key(X), q(X, Y), m(Y)$), respectively. Then, $lg(Q_1, Q_2) = key(X), m(Y)$, which is not linked to $key(X)$, although it is a closed query. In this case, $Q_1 \vee Q_2 = lg(Q_1, Q_2)|_{\mathcal{Q}} = key(X)$, which satisfies the bias condition from the definition. We can show that the resulting $Q_1 \vee Q_2$ is in fact a closed query in the sense of Def. 6. □

Example 3. Continued from Example 2. Fig. 3 shows the iceberg query lattice associated to $\mathbf{r}$ in Ex. 2 and $\mathcal{Q}$ with the support count 1, where each query $Q \in \mathcal{Q}$ has $customer(X)$ as a key atom, denoted by $key(X)$ for short, $Var(Q) \subseteq \{X, Y\}$ and the 2nd argument of predicate *buys* is a constant. □

4 Distributed Closed Pattern Mining in MRDB

Our purpose in this work is to mine global concepts in a distributed setting, where a global database is supposed to be horizontally partitioned appropriately, and stored possibly in different sites. Our approach is to first perform the

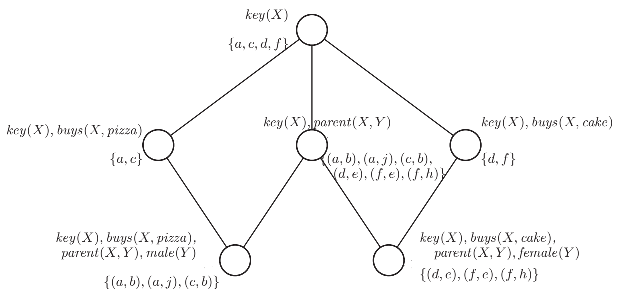


Fig. 3. The Iceberg Query Lattice Associated to **r** in Ex. 2: In the figure, a substitution $\theta = \{X/t_1, Y/t_2\}$ (resp., $\theta = \{X/t_1\}$) in an occurrence set is denoted simply by (t_1, t_2) (resp., t_1). The name of each person in **r** is abbreviated to its first character.

computations of local concepts on each partition of the global DB, and then combine the local concepts by using the subposition operator.

4.1 Horizontal Decomposition of MRDB

We first consider the notion of a *horizontal decomposition* of a multi-relational DB. Since a multi-relational DB consists of multiple relations, its horizontal decomposition is not immediately clear.

Definition 8. Let $\mathbf{r}$ be a multi-relational datalog database with a key predicate *key*. We call a pair $\mathbf{r}_1, \mathbf{r}_2$ a *horizontal decomposition* of $\mathbf{r}$, if

1. $\text{key}_{\mathbf{r}} = \text{key}_{\mathbf{r}_1} \uplus \text{key}_{\mathbf{r}_2}$, i.e., the key relation $\text{key}_{\mathbf{r}}$ in $\mathbf{r}$ is disjointly decomposed into $\text{key}_{\mathbf{r}_1}$ and $\text{key}_{\mathbf{r}_2}$ in $\mathbf{r}_1$ and $\mathbf{r}_2$, respectively, and
2. for any query Q, $answerset(Q, \mathbf{r}) = answerset(Q, \mathbf{r}_1) \cup answerset(Q, \mathbf{r}_2)$. □

The second condition in the above states that the relations other than $\text{key}_{\mathbf{r}}$ are decomposed so that any answer substitution in $answerset(Q, \mathbf{r})$ is computed either in $\mathbf{r}_1$ or $\mathbf{r}_2$, thereby being preserved in this horizontal decomposition.

Given a horizontal decomposition of a multi-relational DB, we can utilize any preferable concept (or closed pattern) mining algorithm for computing local concepts on each partition, as long as the mining algorithm is applicable to MRDM and its resulting patterns satisfy our bias condition. For example, Stumme [20] discussed the algorithm called TITANIC [21], which is based on a level-wise approach. We use here an algorithm called ffLCM [19], which is based on the notion of *closure extension* due to Pasquier et al. [17] in FIM, and then elaborated by Uno et al. [22].

4.2 Subposition Operator in MRDM

We now present the counterpart to Lemma 1 in closed pattern mining in MRDB. We first modify the mapping φ in Def. 3 suitably for our purpose.

Definition 9. Let $\mathbf{r}$ be a datalog database, and $\mathbf{r}_1, \mathbf{r}_2$ a horizontal decomposition of $\mathbf{r}$. Let $(\mathcal{O}(Q; \mathbf{r}), Q)$ be a concept in $\mathbf{r}$, i.e., Q is a closed query and $\mathcal{O}(Q; \mathbf{r}) = \mathit{answerset}(Q, \mathbf{r})$. Then,

$$\tilde{\varphi}((\mathcal{O}(Q; \mathbf{r}), Q)) = ((\mathcal{O}(Q; \mathbf{r}_1), \mathit{Clo}(Q; \mathbf{r}_1)),\ (\mathcal{O}(Q; \mathbf{r}_2),\ \mathit{Clo}(Q, \mathbf{r}_2))).$$

To give the counterpart to Lemma 1 in MRDM, we need another definition of join. Let Q_1 and Q_2 be queries which contain the same set $\mathcal{V}$ of variables, i.e., $\mathit{Var}(Q_1) = \mathit{Var}(Q_2) = \mathcal{V}$. We define $Q_1 \vee_{RR} Q_2 = \mathit{lg}(Q_1, Q_2)|_{\mathcal{V},\mathcal{Q}}$, where, for a query Q, $Q|_{\mathcal{V},\mathcal{Q}}$ is the *restriction* of Q to $\mathcal{V}$ and $\mathcal{Q}$, defined by a conjunction consisting of every literal l in Q such that $\mathit{Var}(l) \subseteq \mathcal{V}$ and l is linked to *key*, i.e., $Q|_{\mathcal{V},\mathcal{Q}}$ is constructed from Q by deleting every literal in Q which contains a variable not in $\mathcal{V}$, then deleting every remaining literal not linked to *key*.

Theorem 3. Let $\mathbf{r}$ be a datalog database, and $\mathbf{r}_1, \mathbf{r}_2$ a horizontal decomposition of $\mathbf{r}$. For any global concept $c = (\mathcal{O}(Q; \mathbf{r}), Q)$ in $\mathbf{r}$, and its image $\tilde{\varphi}(c) = ((\mathcal{O}(Q_1; \mathbf{r}_1), Q_1), (\mathcal{O}(Q_2; \mathbf{r}_2), Q_2))$, it holds that

$$\mathcal{O}(Q; \mathbf{r}) = \mathcal{O}(Q_1; \mathbf{r}_1) \cup \mathcal{O}(Q_2; \mathbf{r}_2) \text{ and } Q = Q_1 \vee_{RR} Q_2.$$

Remark 1. We omit the proof here, since we can prove the theorem similarly to [23]. Instead, we give an example which will be helpful to understand why we need an extra provision for considering the least generalization in this case.

Let Q be a query of the form: $\mathit{key}(A), p(A, B)$. Suppose that Q_1 (Q_2) is a query of the form: $Q_1 = \mathit{key}(A), p(A, B), q(A, B)$ ($Q_2 = \mathit{key}(A), p(A, B), q(B, A)$), respectively. Then, $\mathit{lg}(Q_1, Q_2) = \mathit{key}(A), p(A, B), q(C, D)$, where C and D are newly introduced variables in the least generalization. In this case, since $\mathit{Var}(Q) = \mathit{Var}(Q_1) = \mathit{Var}(Q_2) = \{A, B\}$, $Q_1 \vee_{RR} Q_2$ is $\mathit{key}(A), p(A, B)$, which coincides with Q.

Finally, we note that, in the case of transaction databases, the above theorem coincides with Theorem 1 in Sect. 2. □

Example 4. Continued from Example 3. We consider a horizontal decomposition $\mathbf{r}_1, \mathbf{r}_2$ of $\mathbf{r}$ such that the key relation $\mathrm{key}_{\mathbf{r}}$ (i.e., Customer) in $\mathbf{r}$ is decomposed into $\mathrm{key}_{\mathbf{r}_1} = \{\mathrm{allen}, \mathrm{carol}\}$ and $\mathrm{key}_{\mathbf{r}_2} = \{\mathrm{dian}, \mathrm{fred}\}$, and the other relations than Customer are decomposed so that they satisfy the second condition of Def. 8.

Let Q be a pattern of the form: $[\mathit{key}(X), \mathit{parent}(X, Y)]$ in Fig. 3. We have that $Q_1 = \mathit{Clo}(Q; \mathbf{r}_1) = [Q, \mathit{buys}(X, \mathit{pizza}), \mathit{male}(Y)]$, and $Q_2 = \mathit{Clo}(Q; \mathbf{r}_2) = [Q, \mathit{buys}(X, \mathit{cake}), \mathit{female}(Y)]$. Then, it holds that $Q = Q_1 \vee_{RR} Q_2$. □

5 Distributed Mining Using MapReduce Framework

Since the computation of local concepts can be done independently, it is expected that our algorithm is amenable to data-parallelism. We have therefore implemented our algorithm using MapReduce framework [2], although any framework supporting data-parallelism will do for our purpose.

In MapReduce framework, the user expresses the computation in terms of two functions: *map* and *reduce*. The map function takes an input key/value pair and produces a set of intermediate key/value pairs. Then, the set of intermediate key/value pairs are passed to the reduce function. The reduce function accepts an intermediate key and a set of values for that key, and it then merges (or aggregate) these values together to form a possibly smaller set of values.

However, our use of MapReduce framework is very simple; We use map operation to each local DB to compute a set of its local concepts. An intermediate key/value pair simply consists of $(DB\_id, \mathcal{C}_{id})$, where $\mathcal{C}_{id}$ is the set of local concepts of DB_id. We then apply a reduce operation which simply combines the derived results to form an input to the subsequent subposition operator. We thus simply exploited map for computing local concepts independently. We employed Hadoop[1],an open source implementation of MapReduce.

We now present some preliminary results of our experiments. We implemented our algorithm by using Java 1.6.0_22. Experiments were performed on 6 PCs with Intel Core i5 processors running at 2.8GHz, 8GB of main memory, and 8MB of L2 cache, working under Ubuntu 11.04. We used Hadoop 0.20.2 using 6 PCs, and 2 mappers working on each PC.

Fig. 4 summarizes the results of the execution time for a test data on the mutagenicity prediction,[2] containing 30 chemical compounds. Each compound is represented by a set of facts using predicates such as *atom*, *bond*, for example. The size of the set of predicate symbols is 12. The size of key atom $(active(X))$ is 230, and minimum support $min\_sup = \frac{1}{230}$. We assume that patterns contain at most 4 variables and they contain no constant symbols. The number of the concepts mined is $4,831$.

Fig. 4 shows that the execution times t_1 for mining local concepts are reduced almost linearly with the number of partitions from 1 (i.e., no partitioning) to 8. When the number of partitions is 16, the speed-up did not scale well compared to the other cases. This is a reasonable result; Due to the restriction of our current experiment environment, we used 6 PCs. Therefore, at most 12 mappers are simultaneously available. On the other hand, the execution times t_2 for merging local concepts to obtain global concepts increase almost linearly with the number p of partitions from 1 (i.e., no partitioning) to 16. This is also reasonable; the number of subposition operators applied is $(p-1)$ when we have p partitions.

[1] Hadoop: Open source implementation of MapReduce. http://lucene.apache.org/hadoop/.

[2] http://www.comlab.ox.ac.uk/activities/machinelearning/mutagenesis.html

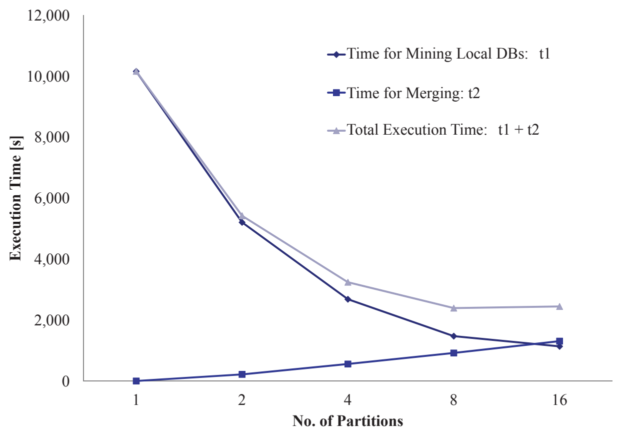


Fig. 4. Execution Time

6 Concluding Remarks

We have studied the problem of mining frequent closed patterns in multi-relational databases in a distributed environment. To do that, we have first reconsidered the notion of iceberg query lattices, where each datalog query contains an atom key, and the variables in a query are linked to the key. We have then proposed the notion of a horizontal decomposition of a given MRDB, and explained how the subposition operator can be utilized to generate the set of closed queries in the global database from the two sets of local closed queries in the two partitions. We have exemplified the effectiveness of our method by some preliminary experimental results using Hadoop.

As discussed in [1], efficiency and scalability have been major concerns in MRDM. Krajca et al. [11, 12] have proposed algorithms which allow us to compute search trees for concepts simultaneously either in parallel or in a distributed manner. Since their approaches are orthogonal to ours, it would be beneficial to employ their algorithms for computing local concepts in our method.

In this work, we have confined ourselves to horizontal partitions of a global context. It will be interesting to study *vertical* partitioning and their mixture in MRDM, where the apposition operator studied by Valtchev et al. [24] will play an important role. Our future work includes developing an efficient algorithm for handling such a general case, as well as accumulating more experimental results on different MRDBs to confirm the effectiveness of our subposition operator.

Acknowledgement The authors would like to thank anonymous reviewers for their useful comments on our paper. The authors are grateful to Seiji Yamazaki for preparing the experiments in this paper.

References

1. Blockeel, H., Sebag, M.: Scalability and efficiency in multi-relational data mining. SIGKDD Explorations Newsletter 2003, Vol.4, Issue 2, pp.1-14 (2003)
2. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM, Vol. 51, No. 1, pp.107–113, 2008.
3. Dehaspe, L.: Frequent pattern discovery in first-order logic, PhD thesis, Dept. Computer Science, Katholieke Universiteit Leuven, 1998.
4. De Raedt, L., Ramon, J.: Condensed representations for Inductive Logic Programming. In: Proc. KR'04, pp. 438-446 (2004)
5. Dzeroski, S.: Multi-Relational Data Mining: An Introduction. SIGKDD Explorations Newsletter 2003, Vol.5, Issue 1, pp.1-16 (2003)
6. Dzeroski, S., Lavrač, N. (eds.): Relational Data Mining. Springer-Verlag, Inc. 2001.
7. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, 1999.
8. Ganter, B., Stumme, G., Wille, R.: Formal Concept Analysis, Foundations and Applications. LNCS 3626, Springer, 2005.
9. Garriga,G. C., Khardon, R., De Raedt, L.: On Mining Closed Sets in Multi-Relational Data. IJCAI 2007, pp.804-809 (2007)
10. Helft, N.: Induction as nonmonotonic inference. In Proc. KR'89, pp. 149–156, 1989.
11. Krajca, P., Vychodil, V.: Distributed Algorithm for Computing Formal Concepts Using Map-Reduce Framework, IDA '09, Springer-Verlag, pp. 333–344, 2009.
12. Krajca, P., Outrata, J., Vychodil, V.: Parallel algorithm for computing fixpoints of Galois connections, AMAI, Vol. 59, No. 2, pp. 257–272, Kluwer Academic Pub., 2010.
13. Kuznetsov, S. O., Obiedkov, S. A.: Comparing performance of algorithms for generating concept lattices. J. Exp. Theor. Artif. Intell., 14(2-3):189.216, 2002.
14. Lloyd, J. W.: Foundations of Logic Programming, Springer, 1987, Second edition.
15. Lucchese, C., Orlando, S., Rergo, R.: Distributed Mining of Frequent Closed Itemsets: Some Preliminary Results. International Workshop on High Performance and Distributed Mining (2005).
16. Nienhuys-Cheng, S-H., de Wolf, R.: Foundations of Inductive Logic Programming, LNAI 1228, Springer, 1997.
17. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering Frequent Closed Itemsets for Association Rules. Proc. ICDT'99, LNAI 3245, pp. 398-416 (1999)
18. Plotkin, G.D.: A Note on Inductive Generalization. Machine Intelligence, Vol. 5, pp. 153-163, 1970.
19. Seki, H., Honda, Y., Nagano, S.: On Enumerating Frequent Closed Patterns with Key in Muti-relational Data. LNAI 6332, pp. 72-86 (2010)
20. Stumme, G.: Iceberg Query Lattices for Datalog. In Conceptual Structures at Work, LNCS 3127, Springer-Verlag, pp. 109-125, 2004.
21. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., Lakhal, L.: Computing Iceberg Concept Lattices with Titanic. J. KDE 42(2), 2002, pp. 189-222.
22. Uno, T., Asai, T. Uchida, Y., Arimura, H.: An Efficient Algorithm for Enumerating Closed Patterns in Transaction Databases. Proc. DS'04, LNAI 3245, pp. 16-31 (2004)
23. Valtchev, P., Missaoui, R.: Building Concept (Galois) Lattices from Parts: Generalizing the Incremental Methods. In Proc. of the 9th Int'l. Conf. on Conceptual Structures: Broadening the Base (ICCS '01), Springer-Verlag, London, UK, 290-303.
24. Valtchev, P., Missaoui, R., Pierre Lebrun, P.: A Partition-based Approach towards Constructing Galois (Concept) Lattices. Discrete Mathematics 256(3): 801-829 (2002)

Attribute Dependencies in a Fuzzy Setting

Cynthia Vera Glodeanu

Technische Universität Dresden,
01062 Dresden, Germany
Cynthia_Vera.Glodeanu@mailbox.tu-dresden.de

Abstract. We present a new framework for modelling users preferences in a fuzzy setting. Starting with a formal fuzzy context, the user enters so-called attribute dependency formulas based on his priorities. The method then yields the "interesting" formal concepts, that is, interesting from the point of view of the user. Our approach is designed for compounded attributes, i.e., attributes which include more than one trait. In this paper, after studying some properties of the formulas, we start investigating the computation of non-redundant bases for them. Such bases are wishful for a better overview of the preferences.

Keywords: Formal Concept Analysis, fuzzy data, data reduction

1 Introduction

Attribute dependency formulas were introduced in [1] and further studied in a series of papers, see for instance [2]. They were developed as a method of controlling the size of crisp concept lattices. The most appealing aspect of this method is that the reduction is done based on the user's preferences, namely he is allowed to define a sort of order on the attributes. In accordance with these preferences, the user receives just the "interesting" concepts, "interesting" from his point of view. In [1] such preferences were modelled in the language of Formal Concept Analysis as follows: An **attribute dependency formula (AD formula)** over a set M of attributes is $A \sqsubseteq B$, where $A, B \subseteq M$. The meaning of the formula is "the attributes from A are less important than the attributes from B". The AD formula $A \sqsubseteq B$ is **true** in $N \subseteq M$, written $N \models A \sqsubseteq B$, if

$$\text{if } A \cap N \neq \varnothing, \text{ then } B \cap N \neq \varnothing.$$

A formal concept $(C, D) \in \mathfrak{B}(G, M, I)$ satisfies $A \sqsubseteq B$ if $D \models A \sqsubseteq B$.

These formulas were the starting point of our work. However, we develop a different kind of AD formulas, namely some that are appropriate for compounded attributes, i.e., attributes which include more than one trait. For instance the notion "wealth" is a compounded attribute consisting of "investment" and "fluency". A person who is wealthy has to have high values on both investment and fluency. We develop such formulas for the fuzzy setting and automatically obtain the crisp case by choosing $\mathbf{L} = \{0, 1\}$ for the residuated lattice.

Some proofs are omitted due to lack of space but can be found in [3].

The paper is structured as follows: In Section 2 we give brief introductions to Fuzzy Sets, Fuzzy Logic and Formal Fuzzy Concept Analysis. Section 3 contains our new framework. Concluding remarks and further research topics are given in the last section.

2 Preliminaries

2.1 Fuzzy Sets and Fuzzy Logics

In this subsection we present some basics about fuzzy sets and fuzzy logic. The interested reader may find more details for instance in [4, 5].

A **complete residuated lattice with (truth-stressing) hedge** is an algebra $\mathbf{L} := (L, \wedge, \vee, \otimes, \rightarrow, ^*, 0, 1)$ such that: $(L, \wedge, \vee, 0, 1)$ is a complete lattice; $(L, \otimes, 1)$ is a commutative monoid; 0 is the least and 1 the greatest element; the adjointness property, i.e., $a \otimes b \leq c \Leftrightarrow a \leq b \rightarrow c$, holds for all $a, b, c \in L$. The hedge * is a unary operation on L satisfying the following conditions: i) $a^* \leq a$; ii) $(a \rightarrow b)^* \leq a^* \rightarrow b^*$; iii) $a^{**} = a^*$; iv) $\bigwedge_{i \in I} a_i^* = (\bigwedge_{i \in I} a_i)^*$; for every $a, b, a_i \in L$ $(i \in I)$. Elements of L are called **truth degrees**, $\otimes$ and $\rightarrow$ are (truth functions of) "fuzzy conjunction" and "fuzzy implication". The hedge * is a (truth function of) logical connective "very true", see [4, 6]. The properties (i)-(iv) have natural interpretations, i.e., (i) can be read as "if a is very true, then a is true", (ii) can be read as "if $a \rightarrow b$ is very true and if a is very true, then b is very true", etc. From the mathematical point of view, the hedge operator is a special kernel operator controlling the size of the fuzzy concept lattice.

A common choice of $\mathbf{L}$ is a structure with $L = [0, 1]$, $\wedge$ and $\vee$ being minimum and maximum, $\otimes$ being a left-continuous t-norm with the corresponding $\rightarrow$. The three most important pairs of adjoint operations on the unit interval are:

$$\text{Łukasiewicz: } a \otimes b := max(0, a + b - 1) \text{ with } a \rightarrow b := min(1,\ 1 - a + b),$$

$$\text{Gödel: } a \otimes b := min(a, b) \text{ with } a \rightarrow b := \begin{cases} 1, \, a \leq b \\ b, \, a \gneqq b \end{cases},$$

$$\text{Product: } a \otimes b := ab \text{ with } a \rightarrow b := \begin{cases} 1, & a \leq b \\ b/a, & a \gneqq b \end{cases}.$$

Typical examples for the hedge are the *identity*, i.e., $a^* := a$ for all $a \in L$, and the *globalisation*, i.e., $a^* := 0$ for all $a \in L \setminus \{1\}$ and $a^* := 1$ if and only if $a = 1$.

Let $\mathbf{L}$ be the structure of truth degrees. A **fuzzy set** (**L-set**) A in a universe U is a mapping $A : U \rightarrow L$, $A(u)$ being interpreted as "the degree to which u belongs to A". We denote by $u \in A$ the fact that $A(u) = 1$. If $U = \{u_1, \dots, u_n\}$, then A can be denoted by $A = \{^{a_1}/u_1, \dots, ^{a_n}/u_n\}$ meaning that $A(u_i)$ equals a_i for each $i \in \{1, \dots, n\}$. Let $\mathbf{L}^U$ denote the collection of all $\mathbf{L}$-sets in U. The operations with $\mathbf{L}$-sets are defined component-wise. For instance, the intersection of $\mathbf{L}$-sets $A, B \in \mathbf{L}^U$ is an $\mathbf{L}$-set $A \cap B$ in U s. t. $(A \cap B)(u) = A(u) \wedge B(u)$ for each $u \in U$, etc. Binary fuzzy relations ($\mathbf{L}$-relations) between X and Y can be thought of as $\mathbf{L}$-sets in the universe $X \times Y$. For $A, B \in \mathbf{L}^U$, the **subsethood degree**, which generalises the classical subsethood relation $\subseteq$, is defined as $\mathrm{S}(A, B) := \bigwedge_{u \in U}(A(u) \to B(u))$. Therefore, $\mathrm{S}(A, B)$ represents a degree to which A is a subset of B. In particular, we write $A \subseteq B$ iff $\mathrm{S}(A, B) = 1$.

Fuzzy closure operators were introduced in [7] and studied further by Belohlavek *at al.*, see for instance [8, 9]. The definition given in [7] mirrors more a crisp thinking, representing a special case of the one given in [8]. Therefore, we will use the latter.

Definition 1. *Define on a set Y two mappings $\mathrm{C}, \kappa : \mathbf{L}^Y \to \mathbf{L}^Y$ satisfying*

$$A \subseteq \mathrm{C}(A), \qquad \kappa(A) \subseteq A, \tag{1}$$

$$\mathrm{S}(A_1, A_2)^* \leq \mathrm{S}(\mathrm{C}(A_1), \mathrm{C}(A_2)), \qquad \mathrm{S}(A_1, A_2)^* \leq \mathrm{S}(\kappa(A_1), \kappa(A_2)), \tag{2}$$

$$\mathrm{C}(A) = \mathrm{C}(\mathrm{C}(A)), \qquad \kappa(A) = \kappa(\kappa(A)), \tag{3}$$

for every $A, A_1, A_2 \in \mathbf{L}^Y$. Then, C *is called an* $\mathbf{L}^*$***-closure operator*** *and κ an* $\mathbf{L}^*$***-kernel operator*** *on Y. A system $\mathcal{S} := \{A_j \in \mathbf{L}^Y \mid j \in J\}$ is a* $\mathbf{L}^*$***-closure system*** *if for each $A \in \mathbf{L}^U$ it holds that*

$$\bigcap_{j \in J} (\mathrm{S}(A, A_j)^* \to A_j) \in \mathcal{S}. \tag{4}$$

The system $\mathcal{S}$ is called an $\mathbf{L}^*$***-kernel system*** *if for each $A \in \mathbf{L}^U$ it holds that*

$$\bigcup_{j \in J} (\mathrm{S}(A, A_j)^* \otimes A_j) \in \mathcal{S}. \tag{5}$$

For the globalisation, (2) becomes

$$A_1 \subseteq A_2 \Longrightarrow \mathrm{C}(A_1) \subseteq \mathrm{C}(A_2), \qquad A_1 \subseteq A_2 \Longrightarrow \kappa(A_1) \subseteq \kappa(A_2),$$

and (4) and (5) become $\bigcap_{j \in J, A \subseteq A_j} A_j$ and $\bigcup_{j \in J, A \subseteq A_j} A_j$, respectively.

Theorem 1. *([8, 9]) A system $\mathcal{S}$ which is closed under arbitrary intersections is an $\mathbf{L}^*$-closure system iff for each $a \in L$ and $A \in \mathcal{S}$ it holds that $a^* \to A \in \mathcal{S}$. A system $\mathcal{S}$ closed under arbitrary unions is an $\mathbf{L}^*$-kernel system iff for each $a \in L$ and $A \in \mathcal{S}$ it holds $a^* \otimes A \in \mathcal{S}$.*

2.2 Formal Fuzzy Concept Analysis

In the following we give brief introductions to Formal Fuzzy Concept Analysis [10, 5, 11].

A triple (G, M, I) is called a **formal fuzzy context** if $I : G \times M \to L$ is an $\mathbf{L}$-relation between the sets G and M and L is the support set of some residuated lattice. Elements from G and M are called **objects** and **attributes**, respectively. The $\mathbf{L}$-relation I assigns to each $g \in G$ and each $m \in M$ the truth degree $I(g, m) \in L$ to which the object g has the attribute m. For $\mathbf{L}$-sets $A \in \mathbf{L}^G$ and $B \in \mathbf{L}^M$, the **derivation operators** are defined by

$$A^{\uparrow}(m) := \bigwedge_{g \in G} (A(g)^* \to I(g, m)), \; B^{\downarrow}(g) := \bigwedge_{m \in M} (B(m)^* \to I(g, m)) \tag{6}$$

for $g \in G$ and $m \in M$. Then, $A^{\uparrow}(m)$ is the truth degree of the statement "m is shared by all objects from A", and $B^{\downarrow}(g)$ is the truth degree of "g has all attributes from B". The operators $^{\uparrow}, ^{\downarrow}$ form a so-called Galois connection with hedges ([11]). A **formal fuzzy concept (L-concept)** is a tuple (A, B) with $A \in \mathbf{L}^G, B \in \mathbf{L}^M$ such that $A^{\uparrow} = B$ and $B^{\downarrow} = A$. Then, A is called the **(fuzzy) extent** and B the **(fuzzy) intent** of (A, B). We denote the set of all **L**-concepts of a given context (G, M, I) by $\mathfrak{B}(G^*, M^*, I)$. Concepts serve for classification. Consequently, the super- and subconcept relation plays an important role. The concept (A_1, B_1) is a **subconcept** of (A_2, B_2), written $(A_1, B_1) \leq (A_2, B_2)$, iff $A_1 \subseteq A_2$ (or, equivalently, $B_1 \supseteq B_2$). Then, we call (A_2, B_2) the **superconcept** of (A_1, B_1). The set of all **L**-concepts ordered by this concept order forms a complete fuzzy lattice (with hedge), the so-called **fuzzy concept lattice** which is denoted by $\underline{\mathfrak{B}}(G^*, M^*, I) := (\mathfrak{B}(G^*, M^*, I), \leq)$, see [11].

3 Fuzzy Attribute Dependencies

Now we are ready to present our new framework. Given a fuzzy formal context, the user obtains the "interesting" concepts after entering a sort of order on the groups of attributes, and fix the truth values for their relevance. Recall, we designed this kind of AD formulas for compounded attributes. Thus, this notion is not the fuzzy equivalent one of the formulas presented in Section 1. For a straightforward fuzzification of those formulas see Remark 1.

Definition 2. *A **fuzzy attribute-dependency formula** (fAD) over a set M of attributes is an expression $A \sqsubseteq B$, where $A, B \in \mathbf{L}^M$ are $\mathbf{L}$-sets of attributes. $A \sqsubseteq B$ is **true** in an $\mathbf{L}$-set $N \in \mathbf{L}^M$ for $\alpha, \beta \in L \setminus \{0\}$ and $\alpha \leq \beta$, written $N \models_{\alpha,\beta} A \sqsubseteq B$, if the following condition is satisfied:*

$$\textit{if } \mathrm{S}(A, N) \geq \alpha, \textit{ then } \mathrm{S}(B, N) \geq \beta. \tag{7}$$

*For an fAD formula or a set T of fAD formulas, the values α and β are called the **thresholds** of $A \sqsubseteq B$ and T. An $\mathbf{L}$-concept $(C, D) \in \mathfrak{B}(G, M, I)$ satisfies $A \sqsubseteq B$ if $D \models_{\alpha,\beta} A \sqsubseteq B$.*

For notational simplicity we will sometimes omit α and β from $\models_{\alpha,\beta}$ provided they are clear from the setting.

The set of all formal concepts from $\mathfrak{B}(G, M, I)$ that satisfy a given set T of fAD formulas is denoted by $\mathfrak{B}_T(G, M, I)$. We call $\mathfrak{B}_T(G, M, I)$ together with the restricted concept order the **fuzzy concept lattice of** (G, M, I) **constrained by** T and denote it by $\underline{\mathfrak{B}}_T(G, M, I)$.

The fAD formulas permit a two-sided modelling of the extracted **L**-concepts. On the one hand, α and β provide the thresholds to which an intent has to contain all elements of A and B. On the other hand, the truth degrees of the elements contained in A and B fix the thresholds to which we want the attributes to be contained in the intent of a concept satisfying the fAD formula. We will illustrate this fact in the forthcoming example.

In applications it is particularly useful to associate to the truth values of a residuated lattice **L** a Likert scale $\mathcal{L}$. This allows the user to have a better understanding of the truth values. For instance let $L = \{0, 0.25, 0.5, 0.75, 1\}$ be the support set of some residuated lattice with the associated Likert scale $\mathcal{L} =$ {not important, less important, important, very important, most important}, i.e, 0 =not important, 0.25 =less important, etc.

Example 1. Consider the fuzzy context given in Figure 1. It represents the evaluation of the employees of a small business regarding some qualities. Here each quality is compounded of two or more traits. For instance, an employee is a "good team player" if he/she is collaborative and not discriminative. The context has 44 **L**-concepts with the Gödel logic which are far too many to be analysed by a busy manager. The manager however knows how good or bad the employees do their jobs and he is interested more in their collaboration than their organisational skills and more in their adaptivity than in their confidentiality. Therefore, he chooses the following two fAD formulas

	good team player		good organisational skills			adaptive towards new			confidential		computer skills	
	a: collaborative	b: not discriminative	c: time management	d: problem solver	e: analytical thinking	f: environment	g: assignment	h: priority	i: judgement	j: discretion	k: word processing	l: database
1	0	0.5	0.5	1	1	0	0	0.5	0.5	0.5	1	1
2	1	1	1	1	1	1	1	1	0.5	0.5	0	0
3	0.5	0.5	0.5	1	1	0	1	1	1	1	0.5	0.5
4	1	0.5	1	1	1	0	1	1	0.5	0.5	1	1
5	0	0.5	0	0.5	0.5	0	0	0.5	0.5	0.5	0	0
6	1	1	0.5	1	1	1	1	1	0.5	0.5	0.5	0.5
7	0	0.5	0	0	0.5	0	0	0.5	0	0.5	0	0

Fig. 1. Fuzzy context about employees

$$\{{}^{0.5}/c, d, e\} \sqsubseteq \{a, b\} \text{ and } \{{}^{0.5}/i, {}^{0.5}/j\} \sqsubseteq \{f, g, h\}, \tag{8}$$

with $\alpha = 0.5$ and $\beta = 1$. Then, he obtains 11 **L**-concepts which are given in Figure 2.

The manager realises that the company does neither send its employees to business trips nor to other companies and the employees should know their priorities. Therefore, he changes the second fAD formula into

$$\{{}^{0.5}/i, {}^{0.5}/j\} \sqsubseteq \{g, {}^{0.5}/h\}. \tag{9}$$

	Extent							Intent											
	1	2	3	4	5	6	7	a	b	c	d	e	f	g	h	i	j	k	l
1	0	0	0	0	0	0.5	0	1	1	1	1	1	1	1	1	1	1	1	1
2	0	0.5	0	0	0	0.5	0	1	1	1	1	1	1	1	1	1	1	0	0
3	0	1	0	0	0	0.5	0	1	1	1	1	1	1	1	1	0.5	0.5	0	0
4	0	0	0	0	0	1	0	1	1	0.5	1	1	1	1	1	0.5	0.5	0.5	0.5
5	0	1	0	0	0	1	0	1	1	0.5	1	1	1	1	1	0.5	0.5	0	0
6	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0	1	0	0	1	0	0	1	0	1	0	0
7	0.5	1	0.5	0.5	0.5	1	0.5	0	1	0	0	1	0	0	1	0	0.5	0	0
8	0.5	0.5	1	0.5	0.5	0.5	0.5	0	0.5	0	0	1	0	0	1	0	1	0	0
9	0.5	1	1	1	0.5	1	0.5	0	0.5	0	0	1	0	0	1	0	0.5	0	0
10	1	1	1	1	0.5	1	0.5	0	0.5	0	0	1	0	0	0.5	0	0.5	0	0
11	1	1	1	1	1	1	1	0	0.5	0	0	0.5	0	0	0.5	0	0.5	0	0

Fig. 2. Concepts satisfying (8)

Obviously the concepts from the first fAD formulas are a subset of the concepts from the second fAD formulas. The second couple of formulas yields 16 concepts, namely those from Figure 2 and 3.

	Extent							Intent											
	1	2	3	4	5	6	7	a	b	c	d	e	f	g	h	i	j	k	l
12	0	0	0.5	0.5	0	0.5	0	1	1	1	1	1	0	1	1	1	1	1	1
13	0	0.5	0.5	0.5	0	0.5	0	1	1	1	1	1	0	1	1	1	1	0	0
14	0	1	0.5	0.5	0	0.5	0	1	1	1	1	1	0	1	1	0.5	0.5	0	0
15	0	0	0.5	0.5	0	1	0	1	1	0.5	1	1	0	1	1	0.5	0.5	0.5	0.5
16	0	1	0.5	0.5	0	1	0	1	1	0.5	1	1	0	1	1	0.5	0.5	0	0

Fig. 3. Concepts satisfying (9) and the first fAD from (8)

We already have the framework of how to select interesting concepts based on the preferences of the user. Let us further investigate some properties of the fAD formulas.

Proposition 1. *Let T be a set of fAD formulas. Then, $\underline{\mathfrak{B}}_T(G, M, I)$ is a complete fuzzy lattice, which is a $\bigvee$-sublattice of $\underline{\mathfrak{B}}(G, M, I)$.*

Proof. Clearly, $\mathfrak{B}_T(G, M, I) \subseteq \mathfrak{B}(G, M, I)$ and $\underline{\mathfrak{B}}(G, M, I)$ with the restricted concept order, is a partially ordered subset of $\underline{\mathfrak{B}}(G, M, I)$. Further note that $\underline{\mathfrak{B}}_T(G, M, I)$ is bounded from below because the least **L**-concept of $\underline{\mathfrak{B}}(G, M, I)$ is $(M^{\downarrow}, M)$, concept which is compatible with every fAD formula. Now, we have to show that $\underline{\mathfrak{B}}_T(G, M, I)$ is closed under arbitrary suprema in $\underline{\mathfrak{B}}(G, M, I)$. To this end let $(A_j, B_j) \in \underline{\mathfrak{B}}_T(G, M, I)$ $(j \in J)$ be **L**-concepts. For any fAD formula $A \sqsubseteq B \in T$, we have $B_j \models A \sqsubseteq B$ for every $j \in J$. Now, if there is $j \in J$ such that $B_j(a) < \alpha$ for some $a \in A$, then $\cap_{j \in J} B_j(a) < \alpha$ and we are done because then $\cap_{j\in J}B_j \models A \sqsubseteq B$. Contrary, if for all $j \in J$ and all $a \in A$ we have $B_j(a) \geq \alpha$, then $\cap_{j\in J}B_j(a) \geq \alpha$ for all $a \in A$. Since $B_j \models A \sqsubseteq B$ holds for all $j \in J$, we also have that $B_j(b) \geq \beta$ for all $j \in J$ and $b \in B$. Due to the same argument as before, it holds $\cap_{j\in J}B_j(b) \geq \beta$ for all $b \in B$ and hence we have $\cap_{j\in J}B_j \models A \sqsubseteq B$, showing that $\underline{\mathfrak{B}}_T(G,M,I)$ is closed under arbitrary suprema. □

In general $\underline{\mathfrak{B}}_T(G,M,I)$ is not closed under arbitrary infima in $\underline{\mathfrak{B}}(G,M,I)$.

The fAD formulas are entered by the user. Thus, the chance that these formulas are redundant is very high. Wishful thinking suggests to have a set of non-redundant formulas because these are then easier to follow and to modify. Therefore, in the following we will develop methods for removing such redundancies.

Definition 3. *An* $\mathbf{L}$*-set* $N \in \mathbf{L}^M$ *is a* ***model*** *of a set* T *of fAD formulas if we have* $N \models A \sqsubseteq B$ *for each* $A \sqsubseteq B \in T$*. Let* $\mathrm{Mod}(T)$ *denote the set of all models of* T*, i.e.,*

$$\mathrm{Mod}(T) := \{N \in \mathbf{L}^Y \mid N \models A \sqsubseteq B \textit{ for each } A \sqsubseteq B \in T\}.$$

An fAD formula $A \sqsubseteq B$ ***follows semantically*** *from* T*, written* $T \models A \sqsubseteq B$*, if for each* $N \in \mathrm{Mod}(T)$*, we have* $N \models A \sqsubseteq B$*.*

Lemma 1. *i)* $N \models A \sqsubseteq \{^{l_1}/y_1, \dots, {}^{l_n}/y_n\}$ *iff* $N \models A \sqsubseteq \{^{l_i}/y_i\}$ *holds for each* $i \in \{1,\dots,n\}$*.*
ii) For each set T *of fAD formulas and each fAD formula* φ*, we have* $T \models \varphi$ *iff* $\lfloor T \rfloor \models \varphi$*, where* $\lfloor T \rfloor := \{A \sqsubseteq \{^{l}/y\} \mid A \sqsubseteq B \in T \textit{ and } B(y) = l\}$*.*

Proof. i) If $\mathrm{S}(A,N) < \alpha$, then we are done. Therefore, suppose $\mathrm{S}(A,N) \geq \alpha$. By the definition of S, we have $\mathrm{S}(\{^{l_1}/y_1, \dots, {}^{l_n}/y_n\}, N) \geq \beta$ if and only if we have $N(\{^{l_i}/y_i\}) \geq \beta$ for all $i \in \{1,\dots,n\}$ and thus $N \models A \sqsubseteq \{^{l_i}/y_i\}$ for all $i \in \{1,\dots,n\}$.
ii) Follows by i), omitted due to lack of space. □

Thanks to Lemma 1 we may merge fAD formulas with the same left-hand side into a single fAD formula. The new formula is true in a model if and only if all its component fAD formulas are true in that model. This lemma allows us also to test semantic entailment in fAD formulas $A \sqsubseteq \{^{l}/y\}$ rather than on the whole $A \sqsubseteq B$.

In the following we will study the connection between the models of fAD formulas and $\mathbf{L}^*$-closure systems. It will turn out that any $\mathbf{L}^*$-closure system can be described by a set of fAD formulas.

Proposition 2. *Let* T *be a set of fAD formulas. Then,* $\mathrm{Mod}(T)$ *is an* $\mathbf{L}^*$*-closure system with* * *being the globalisation.*

Proof. Let $\{N_j \in \mathrm{Mod}(T) \mid j \in J\}$. We will be showing that $\mathrm{Mod}(T)$ is closed under arbitrary intersection, i.e., $\bigcap_{j\in J} N_j$ is a model of T. For any fAD formula

$A \sqsubseteq B \in T$ we have $N_j \models A \sqsubseteq B$ for every $j \in J$. Now, if there is $j \in J$ such that $N_j(a) < \alpha$ for some $a \in A$, then $\cap_{j\in J} N_j(a) < \alpha$ and we are done because then $\cap_{j\in J} N_j \models A \sqsubseteq B$. Contrary, if for all $j \in J$ and all $a \in A$ we have $N_j(a) \geq \alpha$, then $\cap_{j\in J} N_j(a) \geq \alpha$ for all $a \in A$. Since $N_j \models A \sqsubseteq B$ for all $j \in J$ holds, we also have that $N_j(b) \geq \beta$ for all $j \in J$ and $b \in B$. Due to the same argument as before, $\cap_{j\in J} N_j(b) \geq \beta$ for all $b \in B$ and hence we have $\cap_{j\in J} N_j \models A \sqsubseteq B$, showing that $\mathrm{Mod}(T)$ is a closed under arbitrary intersection.

$\mathrm{Mod}(T)$ is a closed under arbitrary intersection and due to Theorem 1 we just have to show that for any $N \in \mathrm{Mod}(T)$ and any $a \in L$, $a^* \to N$ is a model of T. However, this condition only holds if * is the globalisation because, then we have

$$\mathrm{S}(A, a^* \to N) = a^* \to \mathrm{S}(A, N) = \begin{cases} 1, & a = 0, \\ \mathrm{S}(A, N), & a = 1, \end{cases}$$

i.e., $a^* \to N$ trivially satisfies any fAD formula if $a = 0$ or we do not gain anything new to N in the case that $a = 1$. □

Remark 1. One may argue that due to Proposition 2 the fAD formulas are not strong enough. However, we consider that this is not the case. In the crisp setting, the AD formulas form a kernel system, and due to the connection between AD formulas and attribute implications (for details see [1]) one may efficiently compute a non-redundant base of formulas. For instance, if we generalise in a straight-forward way the AD formulas from [1] to the fuzzy setting, then we also obtain just crisp like kernel systems. This can be done as follows: Define a fAD formula $A \sqsubseteq B$, where $A, B \in \mathbf{L}^M$. Further, $A \sqsubseteq B$ is *true* in an $\mathbf{L}$-set $N \in \mathbf{L}^M$ for $\alpha, \beta \in L \setminus \{0\}$ and $\alpha \leq \beta$, written $N \models_{\alpha,\beta} A \sqsubseteq B$, if the following condition is satisfied:

$$\text{if } A \cap N\ \alpha\text{-true, then } B \cap N\ \beta\text{-true,}$$

where an $\mathbf{L}$-set $X \cap Y \in \mathbf{L}^M$ is α-true if there is at least one attribute $m \in M$ such that $(X \cap Y)(m) \geq \alpha$, where $\alpha \in L$. We need the thresholds in order to ensure that the obtained concepts are indeed relevant. Now, the models of such formulas form a crisp like kernel system which can be shown in an analogous way to Proposition 2.

Lemma 2. *For any $\mathbf{L}^*$-closure system $\mathcal{S}$ in M there is a set T of fAD formulas over M such that $\mathcal{S} = \mathrm{Mod}(T)$.*

Proof. Define a set T of fAD formulas by $T := \{A \sqsubseteq C_{\mathcal{S}}(A) \mid A \in \mathbf{L}^M\}$, where $C_{\mathcal{S}}(A)$ is the closure of A given by the L^*-closure operator $C_{\mathcal{S}}$. Further, choose $\alpha = \beta = 1$. Let $N \in \mathcal{S}$, i.e., $N = C_{\mathcal{S}}(N)$. We have to show that N is a model of T, thus let $N \models A \sqsubseteq C_{\mathcal{S}}(A)$ for every $A \sqsubseteq C_{\mathcal{S}}(A) \in T$. If $\mathrm{S}(A, N) < 1$, then $N \models A \sqsubseteq C_{\mathcal{S}}(A)$ and we are done. Now take $\mathrm{S}(A, N) \geq 1$, i.e., $A \subseteq N$. Since $C_{\mathcal{S}}$ is a closure operator we have $C_{\mathcal{S}}(A) \subseteq C_{\mathcal{S}}(N) = N$, hence $\mathrm{S}(C_{\mathcal{S}}(A), N) \geq 1$, i.e., $N \models A \sqsubseteq C_{\mathcal{S}}(A)$. Thus, N is a model of T and we have the first inclusion, namely $\mathcal{S} \subseteq \mathrm{Mod}(T)$. For the converse inclusion let $N \in \mathrm{Mod}(T)$. We have

$$\text{if } \mathrm{S}(N, N) \geq 1, \text{ then } \mathrm{S}(C_{\mathcal{S}}(N), N) \geq 1,$$

where $\mathrm{S}(N, N) \geq 1$ obviously holds and since N is a model of T we must also have $\mathrm{S}(C_{\mathcal{S}}(N), N) \geq 1$ yielding that $N = C_{\mathcal{S}}(N)$, i.e., $N \in \mathcal{S}$ and hence $\mathrm{Mod}(T) \subseteq \mathcal{S}$. □

According to Proposition 2, $\mathrm{Mod}(T)$ is an $\mathbf{L}^*$-closure system, and therefore there must exist an $\mathbf{L}^*$-closure operator $C_{\mathrm{Mod}(T)} : \mathbf{L}^M \to \mathbf{L}^M$ such that $N = C_{\mathrm{Mod}(T)}(N)$ if and only if $N \in \mathrm{Mod}(T)$. Hence, by definition $C_{\mathrm{Mod}(T)}(N)$ is the least model in $\mathrm{Mod}(T)$ which contains N. This definition of the $\mathbf{L}^*$-closure operator does not provide a useful method to compute the closure of a given N. First, because one has to iterate over all models in $\mathrm{Mod}(T)$ and second, such a iteration may be impossible if $\mathbf{L}$ is infinite because then $\mathrm{Mod}(T)$ is infinite.

Similarly to (fuzzy) attribute implications we proceed as follows: For any set T of fAD formulas and for any $\mathbf{L}$-set $N \in \mathbf{L}^M$ of attributes, we define an $\mathbf{L}$-set $N^T \in \mathbf{L}^M$ of attributes as follows:

$$N^T := N \cup \bigcup \{\beta \otimes B \mid A \sqsubseteq B \in T, \mathrm{S}(A, N) \geq \alpha\}. \tag{10}$$

Further, we define an $\mathbf{L}$-set $N^{T_n} \in \mathbf{L}^Y$ of attributes for each non-negative integer by

$$N^{T_n} := \begin{cases} N & \text{if } n = 0, \\ (N^{T_{n-1}})^T & \text{if } n \geq 1. \end{cases} \tag{11}$$

We define an operator $\mathrm{cl_T} : \mathbf{L}^M \to \mathbf{L}^M$ by

$$\mathrm{cl_T}(N) := \bigcup_{n=0}^{\infty} N^{T_n}. \tag{12}$$

Proposition 3. *For each $N \in \mathrm{Mod}(T)$ we have $\mathrm{cl_T}(N) = N$.*

Proof. Omitted due to lack of space. □

The next lemma shows that the $\mathbf{L}^*$-closure operator defined on the models of T coincides with the $\mathrm{cl_T}$-operator defined in (12).

Lemma 3. *Let T be a set of fAD formulas over M. Further let both M and $\mathbf{L}$ be finite. Then, $\mathrm{cl_T}$ is an $\mathbf{L}^*$-closure operator such that for each $N \in \mathbf{L}^M$, $C_{\mathrm{Mod}(T)}(N) = \mathrm{cl_T}(N)$.*

Proof. $C_{\mathrm{Mod}(T)}$ is an $\mathbf{L}^*$-closure operator, therefore it suffices to check that $C_{\mathrm{Mod}(T)}$ and $\mathrm{cl_T}$ coincide. To this end let $N \in \mathbf{L}^M$ be an $\mathbf{L}$-set of attributes. By the definition of $\mathrm{cl_T}$ we have $N \subseteq \mathrm{cl_T}(N)$. We still have to show that $\mathrm{cl_T}(N)$ belongs to $\mathrm{Mod}(T)$ and that $\mathrm{cl_T}(N)$ is the least model containing N. First of all note that the finiteness of $\mathbf{L}$ and M imply that $\mathbf{L}^M$ is finite and that there exists a non-negative integer k such that $\mathrm{cl_T}(N) = N^{T_k}$, where N^{T_k} is given by (11).

We still have to show that $\mathrm{cl_T}(N) \in \mathrm{Mod}(T)$, i.e., for any $A \sqsubseteq B \in T$ we have $\mathrm{cl_T}(N) \models A \sqsubseteq B$. Suppose that $\mathrm{S}(A, \mathrm{cl_T}(N)) \geq \alpha$. Then, $\mathrm{cl_T}(N) = N \cup \{\beta \otimes B\}$. Obviously, $\mathrm{S}(B, N \cup \{\beta \otimes B\}) \geq \beta$, proving that $\mathrm{cl_T}(N) \in \mathrm{Mod}(T)$ which contains N. For any $X \in \mathrm{Mod}(T)$ such that $N \subseteq X$ we have to show that $\mathrm{cl_T}(N) \subseteq X$. This easily follows by the properties of closure operators and by Proposition 3. In fact, we have that $\mathrm{cl_T}(N) \subseteq \mathrm{cl_T}(X) = X$. □

Based on the previous result we present an algorithm for the computation of the closure $C_{\mathrm{Mod}(T)}(N)$ of a fuzzy attribute set $N \in \mathbf{L}^M$ provided $\mathbf{L}$ and M are finite.

Algorithm 1: Closure(N, T)

```
1 repeat
2     take $A \sqsubseteq B \in T$ such that $\mathrm{S}(A, N) \geq \alpha$ and $\mathrm{S}(B, N) < \beta$;
3     set $N$ to $N \cup \{\beta \otimes B\}$;
4 until forall $A \sqsubseteq B \in T$, ($\mathrm{S}(A, N) < \alpha$) or ($\mathrm{S}(A, N) \geq \alpha$ and $\mathrm{S}(B, N) \geq \beta$);
5 return $N$
```

If we choose more restrictive values for α and β, we have the following connection between fuzzy implications and fAD formulas:

Lemma 4. *Let* $\mathrm{Imp}(T) := \{A \Rightarrow B \mid$ *for all* $A \sqsubseteq B \in T\}$ *and* T *be a set of fAD formulas. If we choose* $\alpha = \beta = 1$*, then the following holds*

$$\mathrm{Mod}(\mathrm{Imp}(T)) \subseteq \mathrm{Mod}(T).$$

Proof. Omitted due to lack of space. □

Definition 4. *Two sets* T_1 *and* T_2 *of fAD formulas are called* ***equivalent****, written* $T_1 \equiv T_2$*, if for each* $\varphi_1 \in T_1$ *and* $\varphi_2 \in T_2$ *we have* $T_1 \models \varphi_2$ *and* $T_2 \models \varphi_1$*.*

Lemma 5. *Let* T_1 *and* T_2 *be sets of fAD formulas. Then, the following are equivalent:*
i) $\mathrm{Mod}(T_1) = \mathrm{Mod}(T_2)$*,*
ii) For any fAD formula φ *we have* $T_1 \models \varphi \Longleftrightarrow T_2 \models \varphi$*,*
iii) $T_1 \equiv T_2$*.*

Proof. Omitted due to lack of space. □

Now we are prepared to introduce non-redundant bases.

Definition 5. *A set* T_1 *of fAD formulas is called a* ***non-redundant base*** *of* T *if* $T \equiv T_1$ *and there is no* $T_2 \subset T_1$ *with* $T_2 \equiv T$*. A set* T_1 *of fAD formulas is called a* ***minimal base*** *of* T *if* $T \equiv T_1$ *and for each* T_2 *such that* $T \equiv T_2$*, we have* $|T_1| \leq |T_2|$*.*

Obviously, if T_1 is a minimal base of T, then T_1 is a non-redundant base of T. The converse implication is not true in general.

For a given set T of fAD formulas we may compute a non-redundant base as follows: First note that if $T_1 := T \setminus \{A \sqsubseteq B\}$ and $T_1 \models A \sqsubseteq B$, then $T \equiv T_1$. We may then remove fAD formulas $A \sqsubseteq B$ from T step-by-step until there is no $T_1 \subset T$ such that $T_1 \equiv T$. The computation of a non-redundant base with this method is quite laborious. In what follows, we present another connection between fuzzy attribute implications and fAD formulas which will considerably simplify this task.

Lemma 6. *Let T be a set of fAD formulas. We have* $\mathrm{Mod}(T) = \mathrm{Mod}(\mathrm{Imp}(T^*))$, *where*

$$\mathrm{Imp}(T^*) := \{\alpha \otimes A \Rightarrow \beta \otimes B \mid \forall A \sqsubseteq B \in T,\ \alpha, \beta \text{ thresholds of } T\}. \tag{13}$$

Proof. First note that an $\mathbf{L}$-set $N \in \mathbf{L}^M$ is a **model** of an attribute implication $A \Rightarrow B$ in a fuzzy setting if $||A \Rightarrow B||_N := \mathrm{S}(A, N)^* \to \mathrm{S}(B, N) = 1$.

Let $N \in \mathrm{Mod}(T)$ and $A \sqsubseteq B \in T$. We have two cases: i) $\mathrm{S}(A, N) \geq \alpha$ and $\mathrm{S}(B, N) \geq \beta$ both hold. Consider its first part. Then, for every attribute $m \in M$, we have $A(m) \to N(m) \geq \alpha$ which by the adjointness property gives us $\alpha \otimes A(m) \leq N(m)$ and therefore $\mathrm{S}(\alpha \otimes A, N) = 1$ and thus $\mathrm{S}(\beta \otimes B, N) = 1$. Hence, $\alpha \otimes A \Rightarrow \beta \otimes B||_N = \mathrm{S}(\alpha \otimes A, N)^* \to \mathrm{S}(\beta \otimes B, N) = 1^* \to 1 = 1$. ii) We have $\mathrm{S}(A, N) < \alpha$ which is equivalent to $\mathrm{S}(\alpha \otimes A, N) < 1$. Therefore, $||\alpha \otimes B \Rightarrow \beta \otimes B||_N = \mathrm{S}(\alpha \otimes A, N)^* \to \mathrm{S}(\beta \otimes B, N) = 0 \to \mathrm{S}(\beta \otimes B, N) = 1$. Cases i) and ii) show that N is a model of $\mathrm{Imp}(T^*)$.

For the converse inclusion let $N \in \mathrm{Mod}(\mathrm{Imp}(T^*))$. Then, we have

$$||\alpha \otimes A \Rightarrow \beta \otimes B||_N = \mathrm{S}(\alpha \otimes A, N)^* \to \mathrm{S}(\beta \otimes B, N) = 1 \tag{14}$$

for any fuzzy implication $A \Rightarrow B \in \mathrm{Imp}(T^*)$. Equation 14 holds if and only if one of the following situations appears: i) $(\mathrm{S}(\alpha \otimes A, N)^* = 1$ and $\mathrm{S}(\beta \otimes B, N) = 1)$ $\iff$ $(\mathrm{S}(A, N) \geq 1$ and $\mathrm{S}(B, N) \geq 1)$. ii) $\mathrm{S}(\alpha \otimes A, N)^* = 0$ is equivalent to $\mathrm{S}(\alpha \otimes A, N) < 1$ which further is equivalent to $\mathrm{S}(A, N) < \alpha$. The two cases prove $N \models_{\alpha,\beta} A \sqsubseteq B$. □

Thus, a fAD formula $A \sqsubseteq B$ with thresholds α, β is satisfied if and only if the corresponding implication from $\mathrm{Imp}(T^*)$, where * is the globalisation, holds with truth value 1. With this link between fAD formulas and fuzzy attribute implications we may easily compute a minimal base for any set T of fAD formulas. First, we build the set $\mathrm{Imp}(T^*)$ associated to T as given in (13). For this set we compute a minimal base of attribute implications $\mathcal{B}_{T^*}$. Finally, from $\mathcal{B}_{T^*}$ we obtain a minimal base of fAD formulas for T by

$$\mathcal{B}_T := \{A^\blacklozenge \sqsubseteq B^\blacklozenge \setminus A^\blacklozenge \mid A \Rightarrow B \in \mathcal{B}_{T^*}\},$$

where

$$A^\blacklozenge := \bigvee\{C \in \mathbf{L}^M \mid \alpha \otimes C = \alpha \otimes A\},\ B^\blacklozenge := \bigvee\{D \in \mathbf{L}^M \mid \alpha \otimes D = \alpha \otimes B\}.$$

Note that, unlike the crisp case, in the fuzzy setting a formal context does not have to have a unique stem base (see [12]). The uniqueness is ensured just in the case of the globalisation.

With the possibility of computing a non-redundant base, the user may review his choices and alter them conveniently.

4 Conclusion

We have presented two generalisations of crisp attribute dependency formulas into the fuzzy setting. Both variants allow the user to define a sort of order on the attributes. According to the entered constraints the user sees just a part of the concept lattice, namely the one containing the relevant concepts for him/her. Due to lack of space, the straightforward generalisation was just briefly presented in Remark 1.

The second approach was designed for compound attributes, such which incorporate more than one quality or specification. This time we required from the "interesting" concepts that if they contain all less important attributes with a threshold α, then they should also contain all more important attributes with a threshold β, were the thresholds are truth degrees such that $\alpha \leq \beta$. For such formulas, besides showing some of their properties, we focused mainly on the computation of their non-redundant bases.

Future work will focus on applying the method on real-world data and evaluating the outcomes by experts. Another research topic is the exploration of fAD formulas, where the user may alter the choices made without starting from scratch each time.

References

1. Belohlávek, R., Sklenar, V.: Formal concept analysis constrained by attribute-dependency formulas. In Ganter, B., Godin, R., eds.: ICFCA. Volume 3403 of Lecture Notes in Computer Science., Springer (2005) 176–191
2. Belohlávek, R., Vychodil, V.: Formal concept analysis with background knowledge: Attribute priorities. IEEE Transactions on Systems, Man, and Cybernetics, Part C **39**(4) (2009) 399–409
3. Glodeanu, C.: Attribute dependencies in a fuzzy setting. Technical Report MATH-AL-05-2012, TU Dresden (2012)
4. Hájek, P.: The Metamathematics of Fuzzy Logic. Kluwer (1998)
5. Belohlávek, R.: Fuzzy Relational Systems: Foundations and Principles. Volume 20 of IFSR Int. Series on Systems Science and Engineering. Kluwer Academic/Plenum Press (2002)
6. Hájek, P.: On very true. Fuzzy Sets and Systems **124**(3) (2001) 329–333
7. Biacino, L., Gerla, G.: An extension principle for closure operators. Journal of Mathematical Analysis and Appl. **198** (1996) 1–24
8. Belohlávek, R.: Fuzzy closure operators. Journal of Mathematical Analysis and Appl. **262** (2001) 473–489
9. Belohlávek, R., Funioková, T., Vychodil, V.: Fuzzy closure operators with truth stressers. Logic Journal of the IGPL **13**(5) (2005) 503–513
10. Pollandt, S.: Fuzzy Begriffe. Springer Verlag, Berlin Heidelberg New York (1997)
11. Belohlávek, R., Vychodil, V.: Fuzzy concept lattices constrained by hedges. JACIII **11**(6) (2007) 536–545
12. Belohlávek, R., Vychodil, V.: Attribute implications in a fuzzy setting. In: ICFCA. (2006) 45–60

How Relational Concept Analysis Can Help to Observe the Evolution of Business Class Models

A. Osman Guédi[1,2,3], A. Miralles[2,3], B. Amar[2], C. Nebut[3], M. Huchard[3], and T. Libourel[4]

[1]Université de Djibouti, Avenue Georges Clémenceau BP: 1904 Djibouti (REP)
[2]Tetis/Irstea, Maison de la télédétection, 500 rue JF Breton 34093 Montpellier Cdx 5
[3]LIRMM (CNRS et Univ. Montpellier), 161, rue Ada, F-34392 Montpellier Cdx 5
[4]Espace Dev, Maison de la télédétection, 500 rue JF Breton 34093 Montpellier Cdx 5

Abstract. The development of information systems follows a long and complex process in which various actors are involved. We report an experiment in which we observe the evolution of the analysis model of an information system through 15 successive versions. We use indicators on the underlying concept lattices built by applying Relational Concept Analysis (RCA) to each version. RCA is an extension of FCA which groups entities based on characteristics they share, including links to other entities. It here helps in analyzing their evolution. From this experience, we establish recommendations to monitor and verify the proper evolution of the analysis process.

Keywords: Relational Concept Analysis, Unified Modeling Language, UML, Evolution, Model analysis, Metric, Indicator

1 Introduction

In thematic domains, like environment and territories, the development of information systems often involves many actors and scientists with different (sometimes opposed) viewpoints on a complex and heterogeneous knowledge. The analysis is often conducted during sessions with a different team each time, interspersed with consolidation meetings to cross-check and merge various viewpoints on business concepts or on subsets of UML (Unified Modeling Language) models. Methods and tools are thus welcome to accompany the evolution of systems.

To study the evolution of a system, classical model indicators can be used such as the number of elements of various kinds (classes, methods, etc). However, those indicators do not reveal more complex evolutions such as the precision in the description of model elements, or the level of abstraction and factoring. We thus propose to base evolution analysis not only on classical indicators but also on indicators provided by the application of Formal and Relational Concept Analysis to the successive models. Indeed, Formal Concept Analysis [7] groups together similar business concepts, highlights and brings out new, more abstract, business concepts about which the scientists may not have thought, since the scientists are not necessarily interested in the overall model (experience shows that scientists focus their analytical efforts on the parts of the model they are familiar with or that is needed for their research). Furthermore, FCA removes duplications in a systematic way, by creating new abstractions (within a robust process that leads to a unique solution). The result of the FCA process for a class model of an information system is the equivalent of a normal (non redundant) form. FCA has been used over years for this purpose [9] and is used in an iterative manner in Relational Concept Analysis (RCA [12]) to take into account the richness of the relations between classes, attributes, methods, etc. Because FCA produces a normal form, it offers a framework for comparing the successive versions of the model, disregarding the potential lack of factoring and of abstraction.

In this paper, we report such observation on the evolution of the class model of an information system through 15 successive model versions of the system. We observe a set of indicators, and establish recommendations to use those indicators to monitor and verify the proper evolution of the analysis process. The model under study, *Pesticides*, is described in Section 2. Section 3 explains how lattices on class models can be built thanks to RCA. In Section 4, we introduce indicators on the normal forms and we observe the evolution of those indicators on *Pesticides*. We position our work relatively to the literature in Section 5. Section 6 concludes the paper and gives perspectives of this work.

2 Case Study : the business model *Pesticides*

The project under study is called Environmental Information System for Pesticides (EIS-Pesticides). It aims at defining an information system grouping together the knowledge and the information produced by two teams: the first one (Transfer team), is specialized in the study of the transfer of pesticides to the rivers and the second one (Practice team) mainly works on the agricultural practices of farmers. UML is used to capitalize the knowledge of the thematicians within an analysis model which will be transformed into the schema of the database. During this analysis phase, the adopted methodology consisted to archive the models after each major change. This section answers two questions: i) what was the *Pesticides* model "life" during the analysis phase? ii) what was the evolution of the numbers of model elements?

2.1 A brief history of the *Pesticides* model

We thus have 15 versions for the *Pesticides* model [16]. The V0 is the result of a first analysis of a set of documents and data from the Transfer Team. This V0 model is free of superclasses. The V1 version has been produced during the first analysis session with the Transfer team, where the composite design pattern has been used to organize the hydrographic entities but also those of the landscape. A model refactoring activity and a decomposition in three packages lead to the V3 : superclasses have been added and several associations have been removed. During the next meeting, the model of agricultural activity was detailed, producing a strong increase of the number of classes and consequently the number of model elements (V4 model).

As the model on the activity of metrology was not correct, it was entirely reanalyzed in the V5 model. A copy of the corresponding package was made in order to avoid the loss of concepts, thus producing a strong increase of the model elements. The V6 model results from business concept refinement. To obtain V7, the project leader has removed the remaining concepts wrongly introduced copying the activity model of metrology, thus the model elements number has strongly decreased. The V8 model results from the assignment of a type to all the attributes, and the specification of all the features of the associations (name, cardinalities, name roles). In the V9 model, a pictogram-based language [17, 18] was introduced for spatiality (point, line and polygon) and temporality (time point and time period) notions, it has resulted in the deletion of attributes and their substitution by stereotypes. The next versions (V10 to V14) result from classical analysis : refactoring, errors corrections and specification of some non-detailed items.

2.2 Evolution of the numbers of *Pesticides* model elements

In this section, we analyze the quantity of the included model elements on different versions of the *Pesticides* model. We study the number of classes (#Classes[1]), attributes (#Attributes), associations (#Associations) and the total. Figure 1 shows those metrics for the 15 versions.

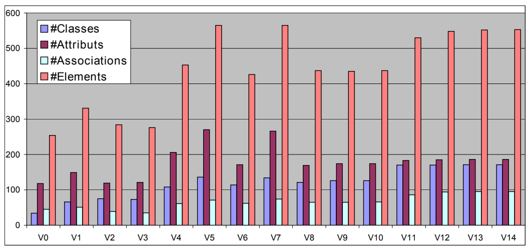


Fig. 1. Evolution of the different model elements

#**Classes**: The number of classes globally increases (except in a few steps): from about 50 classes in V0 to more than 170 classes in V14. The increase from V0 to V3 is explained by the deepening of V0. New business concepts continue to be added up to V4, as the analysis progresses. At V5, classes are duplicated, explaining the large increase. The decreasing between V6 and V7 corresponds to the cleaning-up done by the project leader. Two stable plateaus are observed

[1] #Classes means number of classes

from V8 to V10 and V11 to V14. The increase from V10 to V11 corresponds to a big design activity.

#**Attributes**: When the attribute number increases, this may indicate that business concepts are more precisely described. This is the case between V0 and V1. The introduction of superclasses (more general business concepts) in V2 may explain the decrease of the attribute number, because redundancies may have been removed. Large increase from V4 to V5 may be explained by the duplication of a package and decrease between V6 and V7 by the cleaning-up activity. From V10 to V11, a very slight increase indicates that actors have added some precision on the existing business concepts.

#**Associations**: This metric is an indicator of the relational description of the business concepts. A slight decrease is observed for the very first versions (V0 to V3) followed by a moderate growth in recent versions of the model. This reflects two work phases. In the first, the trainee model was completely restructured, and the decrease may be explained by the introduction of superclasses. Associations may have been factored out on those superclasses. Then, new points of view brought by the different actors may explain the growth.

3 Relational Concept Analysis and model normalization

Like FCA [10], the RCA framework can be used to produce normal forms for UML models that eliminate redundant descriptions, add all the implicit specialization relationships and highlight relevant abstractions. Such a normalization can be seen as the transposition of the normalization step [4] used for the design of relational databases. We illustrate these characteristics on the model of Figure 2 which contains attribute redundancies (*e.g.* `Device Type`) and associations which deserve to be generalized (*e.g.* `Groundwater Instrumentation` and `Rainfall Instrumentation`). An FCA approach applied to such a UML model would, for example, describe the classes by their attributes names, the associations by their roles names and attributes by their type name. The description is given in the form of binary tables called formal contexts. The result of FCA is a lattice of concepts: the entities taken into account in the formal context are grouped according to the properties they share. The discovered groups of entities sharing properties are called concepts. Concepts are formed by maximal sets of entities (the extent of the concept) sharing maximal sets of common properties (the intent of the concept). They are organized in a classification with a lattice structure. The obtained lattice enables to find higher level superclasses like `Measuring Device` which factorizes the attribute `Device Type`. But new discovered abstractions (*e.g.* on classes) cannot be exploited to discover other abstractions (*e.g.* on associations, etc.).

To go beyond, RCA extends FCA on a context family. This family, composed of formal contexts and relational contexts, is called Relational Context Family (RCF). RCA iterates on the RCF and builds several lattices, one for each context representing a model entity[2], that are then used to normalize the UML model. The normal form for the `Measuring station model` is shown in Figure 3. As the relational contexts encode the relationships between the various entities, we use them in particular to formalize (through roles) the UML notion of association between classes. The iterative procedure of RCA first finds the class `Measuring Device` (given by the expert), then the role of type `Measuring Device (Devices)`, and later the association `Instrumentation` (named by the expert) which generalizes the associations `Groundwater Instrumentation` and `Rainfall Instrumentation`. Due to the cardinality of the new association, the semantics of the model is changed. We should add an OCL constraint to allow only one instance of `Rain Gauge` and one instance of `Piezometer`. In this work, we use the lattices that are at the origin of these normal forms as a framework to better compare models and understand their evolution.

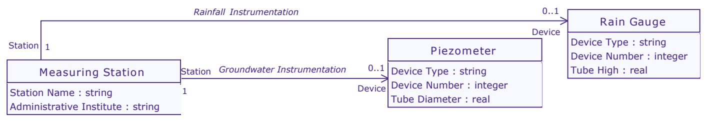


Fig. 2. Extract of the measuring station model

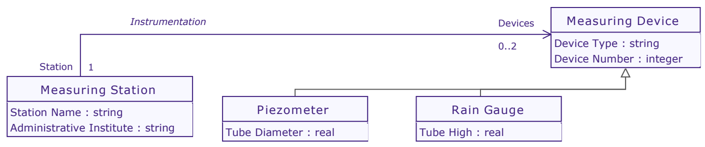


Fig. 3. Refactored model of the Measuring station model (see Fig. 2)

4 Evolution through the indicators on the lattices

4.1 Studied RCA configurations

We denote *RCA configuration* the choice of the elements, the features and the relations that have to be taken into account to produce the normal form. We here consider two configurations C1 and C2 composed of relations between the main modeling elements including classes, attributes and associations. In UML, navigability is a notion that applies to an association $R \subseteq C_{source} \times C_{target}$. If R is

[2] The contexts and lattices of this example are presented at the URL: `http://www2.lirmm.fr/~huchard/Documents/Papiers/RCA_Pesticide_model_example.pdf`

navigable from C_{source} to C_{target}, this means that from an object of C_{source}, you can reach (or query, or send a message to) an object of C_{target}. Furthermore, in our current analysis we select only the roles that are named because they have a stronger semantics. In the two defined configurations, the formal contexts only describe the modeling elements with their names. To preserve inheritance relationships, classes are also described by the names of their superclasses.

The RCA configuration C1 comprises the following elements: classes, attributes, operations, associations and roles (described by their names). The relational contexts of the RCF are as follows: $owns_1 \subseteq classes \times attributes$, $owns_2 \subseteq classes \times operations$, $owns_3 \subseteq classes \times roles$, $owns_4 \subseteq Associations \times roles$.

The RCA configuration C2 extends the previous one by adding a fifth relation $hasType_5 \subseteq roles \times classes$. This configuration leads to much more complex lattices. After several experiments, we observed that the better informative results are obtained at step 5, probably due to the structure of the UML meta-model[3]. Our experiments on the complete process (beyond step 5) are very difficult to interpret. It would be interesting to study this point.

As support tools for our experiments, we developed an UML profile in Objecteering[4], that makes use of the framework eRCA (Eclipse Relational Concept Analysis)[5] for the lattice construction. We applied the two RCA configurations on the 15 versions of the model *Pesticides*.

4.2 Lattice indicators evolution

In this section, we follow the evolution of *Pesticides* with information extracted from lattices used as a normalization framework. For each considered kind of UML model elements, we computed the following indicators to evaluate the unicity of concepts within the model and the increase in new abstract concepts:

1. The ratio of Merged concepts: #Merge/#Model Elements. Merged concepts have a proper extent that contains more than one element. They *merge* several formal objects which have the same description. Note that proper extents contain elements not present in the extent of sub-concepts.
2. The ratio of the New concepts: #New/#Model Elements. New concepts are the concepts whose proper extent is empty. They correspond to the factorization of formal attributes and no formal object is described exactly by their formal attribute set.

We report only the analyses on classes attributes and associations because the number of operations is small and roles behave more or less like associations. The indicators have been computed on lattices at the end of each RCA step for the 15 versions. Visually, the lattices of each version become more complex and indicators help us to go into a detailed analysis.

[3] To navigate from a class to another one via associations, five model elements are crossed: Class, EndAssociation, Association, EndAssociation and Class.

[4] http://www.objecteering.com/

[5] http://code.google.com/p/erca/

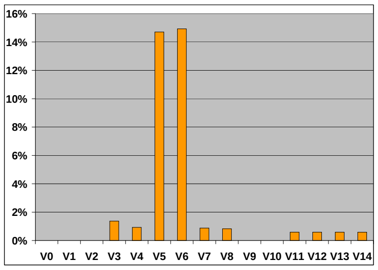


Fig. 4. C1-C2: #Merge/#Classes

Indicators on Classes

#**Merge**/#**Classes**: This indicator counts the number of Merged concepts in the class lattice and compares it to the class number in the model. These concepts group classes with same name and same description. This indicator has the same value in all the RCA steps in the two chosen configurations. In Figure 4, we observe two peaks in V5 and V6 while, for other versions, the indicator is low. Those peaks come from the existence of duplicated classes, e.g. in V5 we have three classes called *Active ingredient* that have the same attributes. Indeed, in V5 and V6, a package was duplicated during a working session. For versions V0, V1, V2, V9 and V10, we do not observe merged concepts: each class owns something that the others do not own. The more the ratio is high, the more we have identical classes. This ratio should alert the designer on the versions where it is high, especially if it does not decrease in the following versions. It is very different from the case where we add many classes.

#**New**/#**Classes**: This indicator counts the number of New concepts in the class lattice relatively to the class number in the model. The new concepts correspond to new class abstractions that would be needed to factorize the similar characteristics of the existing classes. The ratio differs for the two configurations and changes during the RCA steps. It reflects the missing factorizations rate. In the case of C1 (Fig. 5), we notice a progressive decrease although the class number increases. The evolution suggests that even if there are more and more classes, the abstraction level of the model improves. This is confirmed by the fact that the project leader regularly factorized attributes and associations on new added superclasses. The duplication done in V5-V6 degrades the abstraction level: the ratio increases in these steps. This suggests that in the duplicated classes there were missing factorizations.

Analyzing Figure 6 (step 5), we notice peaks suggesting a lack of factorization in V4, V5, V6 and V13, V14. In V4, this is explained by the addition of many classes that lack factorization (also visible in C1, but more evident here because of the inclusion of more information in the configuration). In V5 and V6, this is the effect of the package duplication. In V13, associations have been added, and the project leader a posteriori judges that they introduced missing class factorization. Reversely, a significant fall is observed between V1 and V2,

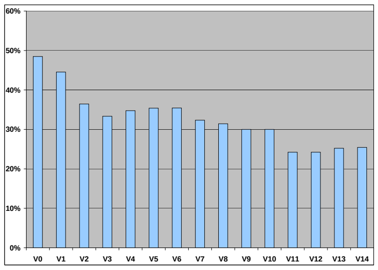


Fig. 5. C1: #New/#Classes

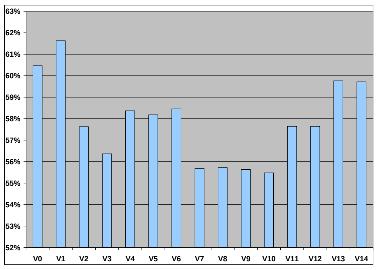


Fig. 6. C2-step 5: #New Concepts/#Classes

due to the refactoring of many associations inserted in the first version by the trainee. When the model is improved on the factorization of classes (resp. associations), the ratio should decrease in C1 (resp. in C2-step5). When it increases, the designer should be warned and (s)he should consider looking at duplications in attributes and associations that (s)he should factorize in relevant superclasses.

Indicators on Attributes

#Merge/#Attributes: This metric gives the ratio of Merged attribute concepts in the attribute lattice relatively to the attribute number (Fig. 7). These concepts group attributes with same name (in a model, this often corresponds to redundant attributes). A general decreasing tendency is observed, revealing less attribute redundancy and confirming that factorization has been improved. The ratio remains low. Peaks in V1 to V3 correspond to important additions of attributes with similar names. The peak at versions V5 and V6 corresponds to the mentioned package duplication. After V7, the attribute number slightly increases, but the ratio decreases, thus the attributes that are introduced do not introduce redundancy because either these are new properties or the designer improves the model by removing the repetition of property names.

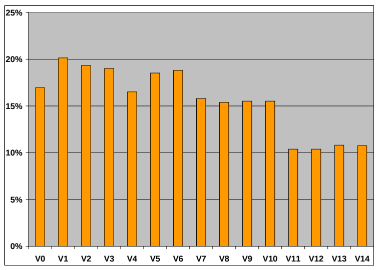


Fig. 7. C1-C2: Merge/Attributes

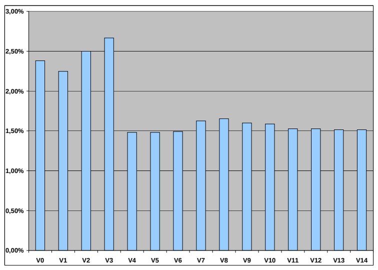


Fig. 8. C1-C2: New/Attributes

#New/#Attributes: This ratio expresses the proportion of new attribute concepts relatively to the attributes (Fig. 8). With the chosen configurations (C1 and C2), there are no new concepts in the attribute lattice. However, attributes participate massively in the creation of new class and association abstractions.

Indicators on Associations
#Merge/#Associations: This ratio gives the number of Merged association concepts relatively to the number of associations (Fig. 9). When decreasing tendencies are observed, the designer confirmed that the model was improved by association and role factorization. In V0 and V1 which not contain superclasses and V5 and V6 where a lot of associations are duplicated, the ratio of merged concepts is high because the factorization level is low.

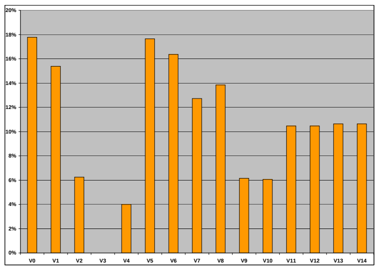


Fig. 9. C1-C2: #Merge/#Associations

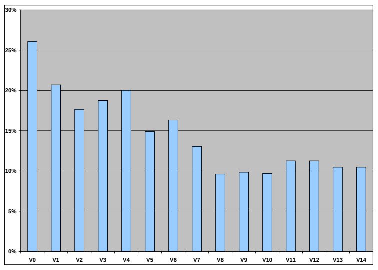


Fig. 10. C1: #New/#Associations

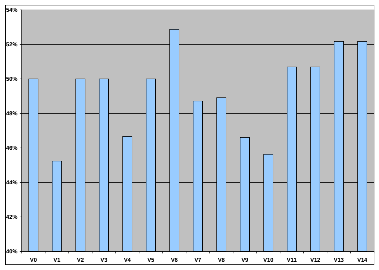


Fig. 11. C2-step 5: #New/#Associations

#New/#Associations: This ratio gives the number of New association concepts relatively to the number of associations. It expresses the potentially missing abstractions of associations. For C1 (Fig. 10), the global tendency in versions V0-V9 is decreasing, suggesting that factorization improves. At the end, it slightly grows, and analyzing the models, we noticed that the added associations needed factorization. For C2 (Fig. 11), we notice that V1 lacks many class factorizations, and comparatively not so many association factorizations. Actually, V1 version is a rather chaotic transition from the trainee model to the current model. From V2 to V4, there are very few associations, and classes and associations behave differently. They do not much influence each other. After V7, classes and associations have the same behavior, indicating a needed factorization growing at the end (V13 and V14).

4.3 Discussion

The evolution of the data encapsulation level is highlighted by the evolution of the number of classes. The evolution of attribute number might indicate the level of completion of the model. The relational aspect is shown by the evolution of the numbers of roles and associations. But these numbers of elements cannot indicate if we really completed the model or if we have introduced duplication and redundancies. This is where lattice-based indicators help. By comparing the results of indicators and the known "life" of the model, we can draw advice. The evolution of the number of merged concepts indicates if identical or badly described model elements have been introduced into a version. The evolution of new concept numbers gives a measure of the increasing of the abstraction level of the model or its degradation. The new class and association concepts also indicate the effort needed to reach a normalized model, by computing the number of abstractions necessary to factorize the entities.

5 Related Work

Several well known software metrics have been proposed for measuring the quality of inheritance [3, 15]. For example, DIT [3] measures the length of a longest path from the root to a leaf of the inheritance tree; NRM [15] counts the number of overridden methods (not inherited, not specialized). High inheritance use and overriding correspond to an increasing complexity, but also to an improvement of the abstraction level and of the opportunity to reuse methods and attributes in subclasses. Empirical evaluations [2, 11] of these metrics assess if they can predict faults in software. Here, we are not interested in detecting faults but in following the construction of a class model covering a domain. Counting the main modeling elements gives indications about the degree of reification of the model and the degree of completion of the model. Indicators built using the lattices assess the degree of reification, but more deeply analyze the abstraction level of reified entities, their structuring and their description.

In [5], the proposed metrics measure the quality of factoring in inheritance hierarchies by reference to an ideally factorized hierarchy obtained by constructing a Galois subhierarchy on flattened class description. Here, we observe identified categories of concepts in lattices (Merged and New concepts) because they are immediately understandable by the expert, who can exploit the resulting concepts to build new classes or associations in his/her model.

Software metrics are included in frameworks dedicated to software evolution analysis. In [14], an *evolution matrix* of classes is proposed. The size of classes (in LOC) is represented by a variable-size box representing the class in a specific version. Main evolution phases emerge (growth, stabilization, etc.) and specific behavior of classes (*e. g.* permanently large classes) are highlighted. Pingzer et al. [21] use Kiviat diagrams for visualizing metrics through several releases.

Other approaches use information on the releases [6], or on basic changes or code churns [20]. In [13], authors propose a meta-heuristic-based approach for determining an editing distance of minimal cost between two successive versions of a class model. This distance counts the design changes between the versions and is compared with traditional metrics for predicting defects. Here we want to understand evolution and identify its main phases. We use information on the model elements, and on the lattices to alert the designer on the lack of details or the lack of abstraction of the model. In the database domain, several approaches have been proposed for managing the evolution of object-oriented database schema [1, 19, 22]. They mainly define the operations used for schema evolution. An evolution combines a set of primitive evolution operations as well as global rules used for validating or invalidating the evolution. Here we propose a set of indicators for observing and understanding the evolution. We interpret results w.r.t. the known evolution history and we establish recommendations.

6 Conclusion and perspectives

In this paper, we presented an observation of the evolution of the business class model of an information system. Metrics on model elements and indicators on the lattices generated with RCA are systematically computed on the 15 versions of *Pesticides* model. The results of these metrics and indicators offer to the model designer a dashboard that can be used to monitor the process of analysis of its information system. Indicators based on RCA give a normalization framework for the monitoring, highlighting aspects of the evolution that are not captured by the model metrics. The observation of the evolution of the analysis process on 15 versions of the *Pesticides* model allowed recommendations to be extracted, that are confirmed by the story of the model.

As future work, we will implement traceability links to better monitor and understand the evolution of business concepts in the analysis process to help the designer. Beyond the evolution analysis, and following tracks of previous work, we would like to better control the new concepts that emerge from RCA to build a normalized UML model.

References

1. Banerjee, J., Kim, W., Kim, H.J., Korth, H.F.: Semantics and implementation of schema evolution in object-oriented databases. In: SIGMOD Conference. pp.

311–322. ACM Press (1987)
2. Briand, L.C., Morasca, S., Basili, V.: Property-based software engineering measurement. Transactions on Software Engineering 22(1), 68–86 (1996)
3. Chidamber, S.R., Kemerer, C.F.: A metrics suite for object oriented design. IEEE Transactions on Software Engineering 20(6), 476–493 (Jun 1994)
4. Codd, E.F.: A relational model for large shared data banks. In: Comm. of ACM 13(6) (1970)
5. Dao, M., Huchard, M., Libourel, T., Roume, C., Leblanc, H.: A new approach to factorization - introducing metrics. In: IEEE METRICS. pp. 227–236. IEEE Computer Society (2002)
6. Gall, H., Jazayeri, M., Riva, C.: Visualizing software release histories: The use of color and third dimension. In: ICSM. pp. 99–108 (1999)
7. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical Foundations. Springer-Verlag (1999)
8. Ganter, B., Stumme, G., Wille, R. (eds.): Formal Concept Analysis, Foundations and Applications, Lecture Notes in Computer Science, vol. 3626. Springer (2005)
9. Godin, R., Mili, H.: Building and maintaining analysis-level class hierarchies using galois lattices. In: OOPSLA. pp. 394–410 (1993)
10. Godin, R., Valtchev, P.: Formal concept analysis-based class hierarchy design in object-oriented software development. In: Ganter et al. [8], pp. 304–323
11. Gyimóthy, T., Ferenc, R., Siket, I.: Empirical validation of object-oriented metrics on open source software for fault prediction. IEEE Trans. Software Eng. 31(10), 897–910 (2005)
12. Huchard, M., Hacene, M.R., Roume, C., Valtchev, P.: Relational concept discovery in structured datasets. Ann. Math. Artif. Intell. 49(1-4), 39–76 (2007)
13. Kpodjedo, S., Ricca, F., Galinier, P., Guéhéneuc, Y.G., Antoniol, G.: Design evolution metrics for defect prediction in object oriented systems. Empirical Software Engineering 16(1), 141–175 (2011)
14. Lanza, M., Ducasse, S.: Understanding software evolution using a combination of software visualization and software metrics. L'OBJET 8(1-2), 135–149 (2002)
15. Lorenz, M., Kidd, J.: Object-Oriented Software Metrics: A Practical Guide. Prentice-Hall (1994)
16. Miralles, A., Pinet, F., Carluer, N., Vernier, F., Bimonte, S., Lauvernet, C., Gouy, V.: EIS Pesticides: an information system for data and knowledge capitalization and analysis. In: Euraqua-PEER Scientific Conference, 26/10/2011 - 28/10/2011. p. 1. Montpellier, FRA (2011)
17. Miralles, A., Libourel, T.: A new methodology to automate the transformation of gis models in an iterative development process. In: Advances in Modelling Agricultural Systems, vol. 25, pp. 19–36. Springer (2009)
18. Miralles, A., Pinet, F., Bédard, Y.: Describing spatio-temporal phenomena for environmental system development: An overview of today's needs and solutions. International Journal of Agricultural and Environmental Information Systems 1(2), 64–84 (2010)
19. Monk, S., Sc, S.M.B.: A model for schema evolution in object-oriented database systems. (1993)
20. Nagappan, N., Ball, T.: Use of relative code churn measures to predict system defect density. In: ICSE. pp. 284–292. ACM (2005)
21. Pinzger, M., Gall, H., Fischer, M., Lanza, M.: Visualizing multiple evolution metrics. In: SOFTVIS. pp. 67–75. ACM (2005)
22. Skarra, A.H., Zdonik, S.B.: The management of changing types in an object-oriented database. In: OOPSLA (1986)

Finding Errors in New Object Intents

Artem Revenko[1][2]* and Sergei O. Kuznetsov[2]

[1] Technische Universität Dresden
Zellescher Weg 12-14, 01069 Dresden, Germany
[2] National Research University Higher School of Economics
Pokrovskiy bd. 11, 109028 Moscow, Russia
artem_viktorovich.revenko@mailbox.tu-dresden.de,
skuznetsov@hse.ru

Abstract. The classification of possible errors in object intents is given and some possibilities of exploring them are discussed. An approach for finding some types of errors in new object intents is introduced. This approach is based on finding implications from an implication basis of the context that are not respected by a new object. It is noted that this approach may lead to a computationally intractable solution. An alternative approach based on computing closure of subsets of object intent is considered. The alternative approach allows one to find a polynomial time algorithm and to deal with inconsistent combination of attributes. This algorithm may also be used to prove object's correctness (existence) or to add missing attributes and remove unnecessary attributes from the object's intent. Experimental results are discussed.

Keywords: formal context, implication, error exploration

1 Introduction

The work is motivated by the idea of building a multi-user system based on methods of Formal Concept Analysis. Consider two well known multi-user data representation systems: QED and Wikipedia. The aim of the QED project ([1]) is to build a single, distributed, computerized repository that rigorously represents all important, established mathematical knowledge. For this purpose all the input data for QED project should be mathematically formalized. However, it is not always possible in practice. In Wikipedia one may input any data, but no reasoning is supported. To find inferences or consequences from data given in Wikipedia one should read it and process it by own means. However, Formal Concept Analysis offers methods for reasoning with not completely formalized data. Error finding tools are absolutely necessary for such multi-user systems. In Wikipedia the task is solved by moderations, in QED project all the data is checked by automatic provers. To our knowledge there is no FCA-based research on error detection.

* We thank Bernhard Ganter and Sergei Obiedkov for discussion and useful remarks.

In this paper we discuss possible error types in new object intents, choose two most important ones, and propose efficient methods for finding them. We provide two different approaches for revealing such errors: one based on computing implication system of the context and another one based on computing the closures of the subsets of the new object intent. Since computing closures may be performed much faster we improve and generalize this approach and finally obtain a procedure for finding all possible errors of considered types. This procedure may also be used to prove correctness of a given object. However, it is not possible to completely escape the necessity of checking some objects by hand. However, the procedure allows one to prove the correctness of input data checking by hand only maximal objects. We finish the paper by discussing experimental results.

In this paper we assume that we are given a context (possibly empty) with correct data and a number of new object intents that may contain errors. This data is taken from some data domain and we may ask an expert whose answers are always correct. However, we should ask as few questions as possible.
All sets and contexts we consider in this paper are assumed to be finite.

2 Main Definitions

Let G and M be sets. Let $I \subseteq G \times M$ be a binary relation between G and M. Triple $\mathbb{K} := (G, M, I)$ is called a *(formal) context*.
Set G is called a set of *objects*. Set M is called a set of *attributes*.

Consider mappings $\varphi\colon 2^G \to 2^M$ and $\psi\colon 2^M \to 2^G$: $\varphi(X) := \{m \in M \mid gIm \text{ for all } g \in X\}$, $\psi(A) := \{g \in G \mid gIm \text{ for all } m \in A\}$. For any $X_1, X_2 \subseteq G$, $A_1, A_2 \subseteq M$ one has

1. $X_1 \subseteq X_2 \Rightarrow \varphi(X_2) \subseteq \varphi(X_1)$
2. $A_1 \subseteq A_2 \Rightarrow \psi(A_2) \subseteq \psi(A_1)$
3. $X_1 \subseteq \psi\varphi(X_1)$ and $A_1 \subseteq \varphi\psi(A_1)$

Mappings φ and ψ define a *Galois connection* between $(2^G, \subseteq)$ and $(2^M, \subseteq)$, i.e. $\varphi(X) \subseteq A \Leftrightarrow \psi(A) \subseteq X$. Usually, instead of φ and ψ a single notation $(\cdot)'$ is used. $(\cdot)'$ is sometimes called a *derivation operator*. For $X \subseteq G$ the set X' is called the *intent* of X and is denoted $\operatorname{int}(X)$. Similarly, for $A \subseteq M$ the set A' is called the *extent* of A and is denoted $\operatorname{ext}(A)$.
Let $Z \in M \cup G$. $(Z)''$ is called the *closure* of Z in $\mathbb{K}$. Applying Properties 1 and 2 consequently one gets the *monotonicity* property: for any $Z_1, Z_2 \in G \cup M$ one has $Z_1 \subseteq Z_2 \Rightarrow Z_1'' \subseteq Z_2''$.
Let $m \in M, X \subseteq G$, then $\overline{m}$ is called a *negated* attribute iff $\overline{m} \in X'$ whenever $m \notin X'$.

An *implication* of $\mathbb{K} := (G, M, I)$ is defined as a pair (A, B), written $A \to B$, where $A, B \subseteq M$. A is called the *premise*, B is called the *conclusion* of implication $A \to B$. Implication $A \to B$ is *respected by a set of attributes* N if $A \nsubseteq N$ or $B \subseteq N$. Implication $A \to B$ holds (is valid) in $\mathbb{K}$ if it is respected by all int(g), $g \in G$, i.e. every object, that has all the attributes from A, also has all the attributes from B. Implications satisfy *Armstrong rules*:

$$\frac{}{A \to A}\ ,\quad \frac{A \to B}{A \cup C \to B}\ ,\quad \frac{A \to B, B \cup C \to D}{A \cup C \to D}$$

A *support* of an implication in context $\mathbb{K}$ is the set of all objects of $\mathbb{K}$, whose intents contain the premise and the conclusion of the implication. A *unit implications* is defined as an implication with only one attribute in conclusion, i.e. $A \to b$, where $A \subseteq M,\ b \in M$. Every implication $A \to B$ can be regarded as the set of unit implications $\{A \to b \mid b \in B\}$. One can always observe only unit implications without loss of generality.

An *implication basis* of a context $\mathbb{K}$ is defined as a set $\mathfrak{L}$ of implications of $\mathbb{K}$, from which any valid implication for $\mathbb{K}$ can be deduced by the Armstrong rules and none of the proper subsets of $\mathfrak{L}$ has this property.
A minimal implication basis is an implication basis minimal in the number of implications. A minimal implication basis was defined in [6] and is known as the *canonical implication basis.* In paper [5] the premises of implications from canonical base were characterized in terms of pseudo-intents. A subset of attributes $P \subseteq M$ is called a *pseudo-intent*, if $P \neq P''$ and for every pseudo-intent Q such that $Q \subset P$, one has $Q'' \subset P$. The canonical implication basis looks as follows: $\{P \to (P'' \setminus P) \mid P$ - pseudo-intent$\}$.

We say that an object g is *reducible* in a context $\mathbb{K} := (G, M, I)$ iff $\exists X \subseteq G :$ $g' = \bigcap\limits_{j \in X} j'$.

3 Classification of Errors

In this section we use the idea of *data domain dependency.* Usually objects and attributes of a context represent entities (e.g. physical objects, mathematical instances, goods and services, etc.). Dependencies may hold on attributes of such entities (e.g. if an object represents a convex quadrangle, the sum of all angles should be equal to π). However, such dependencies may not be implications of a context as a result of an error in object intents. Thereby, data domain dependencies are such rules that hold on data represented by objects in a context, but may erroneously be not valid implications of a context. We aim to restore valid dependencies and therefore correct errors.

Every object in a context is described by its intent. In the data domain there may exist dependencies between attributes. In this work we consider only dependencies that do not have negations of attributes in premises. As mentioned above there is no need to specially observe non-unit implications. Consider possible types of such dependencies ($A \subseteq M,\ b, c \in M$):

1. $A \to b$
2. $A \to \overline{b}$
3. $A \to b \vee c$

4. $A \to \Phi$, where Φ is any logical formula not considered above, for example, $\Phi = a \vee (b \wedge \overline{c})$

If we have no errors in a context, all the dependencies of Type 1 are deducible from implication basis. However, if we have not yet added enough objects in the context, we may get false consequence. Nevertheless, it is guaranteed that none of valid dependencies is lost, and, as we add objects without errors we reduce the number of false consequences from the implication basis.
The situation is different if we add an erroneous object. It may violate a dependency valid in the data domain. In this case, until we find and correct the error, we are not able to deduce all dependencies valid in the data domain from the implication basis, no matter how many correct objects we add afterwards.
Now we take a closer look at various possible cases. If a dependency of Type 3 holds in the data domain, there should be a restriction on reducible objects in the context. However, reducible objects do not change neither the closure system, nor the implication basis of a context. This case would be an interesting direction for further investigation, but now we do not consider this case.
In Type 4 formula Φ can be represented in conjunctive normal form. That is why we may hope that if it is possible to find and generalize solution for Type 3, we would be able to reveal dependencies of Type 4 as well.
Types 1 and 2 are most simple and common dependencies. In this work we try to find the algorithm to reveal these two types of dependencies and find corresponding errors.

4 Finding Errors

We introduce two different approaches to finding errors. The first one is based on inspecting the canonical basis of a context. When adding a new object to the context one may find all implications from the canonical basis of the context such that the implications are not respected by the intent of the new object. These implications are then output as questions to an expert in form of unit implications. If at least one of these implications is accepted, the object intent is erroneous. Since the canonical basis is the most compact (in the number of implications) representation of all valid implications of a context, it is guaranteed that minimal number of questions is asked and no valid dependencies of Type 1 are left out.
Although this approach allows one to reveal all dependencies of Type 1, there are several issues. The problem of producing the canonical basis with known algorithms is intractable. Recent theoretical results suggest that the canonical base can hardly be computed with better better worst-case complexity than that of the existing approaches ([3], [2]). One can use other bases (for example, see progress in computing proper premises [9]), but the algorithms known so far are still too costly and non-minimal bases do not guarantee that the expert is asked minimal sufficient number of questions.

However, since we are only interested in implications corresponding to an object, it may be not necessary to compute a whole implication basis. Here is the second approach. Let $A \subseteq M$ be the intent of the new object not yet added to the context. $m \in A''$ iff $\forall g \in G : A \subseteq g' \Rightarrow m \in g'$, in other words, A'' contains the attributes common to all object intents containing A. The set of unit implications $\{A \to b \mid b \in A'' \setminus A\}$ can then be shown to the expert. If all implications are rejected, no attributes are forgotten in the new object intent. Otherwise, the object is erroneous. This approach allows one to find errors of Type 1.

5 An Example

Consider the following example with convex quadrangles. The formal context in Fig. 1 contains convex quadrangles and their properties. The context is not full, i.e. not all possible convex quadrangles are considered, and some objects in the context are reducible (they do not bring new information in the implication basis of the context). Seven attributes are considered. Attributes "has equal legs" and "has equal angles" require at least two angles/legs of a quadrangle to be equal. Some dependencies on attributes are obvious, e.g., it is clear that if all angles are equal in a quadrangle, then this quadrangle definitely has equal angles.
Four errors are presented in Fig 2. Errors are added to the context in Fig. 1 one at a time. One should treat an error as an object to be added to the context.
The context without errors in Fig. 1 is denoted $\mathbb{K}$, $(\cdot)'$ is the corresponding derivation operator.
The context of errors in Fig. 2 is denoted by $\mathbb{K}_{\mathrm{e}}$, $(\cdot)^{\mathrm{e}}$ is the derivation operator for $\mathbb{K}_{\mathrm{e}}$.

Example 1. $\{$Erorr 2$\}^{\mathrm{e}}$ =$\{$has equal legs, has right angle, all legs equal, all angles equal$\}$
$\{$Erorr 2$\}^{\mathrm{e}''}$ =$\{$has equal legs, has right angle, all legs equal, all angles equal, has equal angles$\}$

The canonical basis of the context $\mathbb{K}$ in Fig. 1 looks as follows:

1. at least 3 different angles $\to$ at least 3 different legs
2. all angles equal $\to$ has equal angles, has equal legs, has right angle
3. all legs equal $\to$ has equal angles, has equal legs
4. has right angle, at least 3 different legs $\to$ at least 3 different angles
5. has equal angles, has equal legs, at least 3 different legs, all legs equal $\to$ has right angle, at least 3 different angles, all angles equal
6. has equal angles, has equal legs, at least 3 different legs, all angles equal, has right angle, at least 3 different angles $\to$ all legs equal
7. has right angle, has equal legs, all legs equal, has equal angles $\to$ all angles equal

Consider Error2. $\{$Erorr 2$\}^{\mathrm{e}}$ =$\{$has equal legs, has right angle, all legs equal, all angles equal$\}$
Using the first approach we find that this object does not respect Implications 2 and 3. It is easy to see that both implications are valid in data domain. Thereby,

Convex quadrangles	has equal legs	has equal angles	has right angle	all legs equal	all angles equal	at least 3 different angles	at least 3 different legs
Square	×	×	×	×	×		
Rectangle	×	×	×		×		
Quadrangle						×	×
Rhombus	×	×		×			
Parallelogram	×	×					
Rectangular trapezium		×	×			×	×
Quadrangle with 2 equal legs and right angle	×		×			×	×
Isosceles trapezium	×	×				×	
Rectangular trapezium with 2 equal legs	×	×	×			×	×
Quadrangle with 2 equal angles		×				×	×
Quadrangle with 2 equal legs	×					×	×
Quadrangle with 2 equal legs and 2 equal angles	×	×				×	×

Fig. 1. Context of convex quadrangles $\mathbb{K}$

Errors	has equal legs	has equal angles	has right angle	all legs equal	all angles equal	at least 3 different angles	at least 3 different legs
Error1	×			×		×	
Error2	×		×	×	×		
Error3		×	×	×	×	×	×
Error4	×	×		×			×

Fig. 2. Context of errors $\mathbb{K}_e$

the expert recognizes object as an error.
As {Erorr 2}$^{e''}$ ={has equal legs, has right angle, all legs equal, all angles equal, has equal angles}, the second approach yields the following implication: {has equal legs, has right angle, all legs equal, all angles equal} $\rightarrow$ {has equal angles}. This implication is also valid in data domain. Although this implication is less general than Implications 2 and 3, it is sufficient to indicate an error.

6 Improvements

Obviously, applying the derivation operator two times is much easier task than computing the canonical basis, and can be performed in polynomial time. However, the following case is possible. Let $A \subseteq M$ be the intent of the new object such that $\nexists g \in G : A \subseteq g'$. In this case $A'' = M$ and the implication $A \rightarrow A'' \setminus A$ has empty support. This may indicate an error of Type 2, because the object intent contains combination of attributes impossible in the data domain, but the object may be correct as well. An expert could be asked if the combination of attributes in the object intent is consistent in the data domain. For such a question the information already input in the context is not used. More than that, this question is not sufficient to reveal an error of Type 1.

Proposition 1. *Let $\mathbb{K} = (G, M, I), A \subseteq M$. The set*

$$\mathcal{I}_A = \{B \rightarrow d \mid B \in \mathcal{MC}_A, d \in B'' \setminus A \cup \overline{A \setminus B}\},$$

where $\mathcal{MC}_A = \{B \in \mathcal{C}_A | \nexists C \in \mathcal{C}_A : B \subset C\}$ and $\mathcal{C}_A = \{A \cap g' \mid g \in G\}$, is the set of all unit implications (or their non-trivial consequences with some attributes added in the premise) of Types 1 and 2 such that implications are valid in $\mathbb{K}$, not respected by A, and have not empty support.

Proof. Let $(E \rightarrow f) \in \mathcal{I}_A$. As $E = A \cap g'$ for some $g \in G$, $f \in g'$. Consider possible cases:

1. $f \in E'' \setminus A$. As follows from the definition of derivation operator, the implication is valid and $f \in g'$, i.e. at least g is in support of this implication. More than that, $E'' \setminus A \not\subseteq A$ and implication is not respected by A.
2. $f \in \overline{A \setminus E}$. Since E is a maximal intersection ($\nexists C \in \mathcal{C}_A : E \subseteq C$), there does not exist object $\hat{g} \in G$ such that $E \cup m \in \hat{g}'$, for any $m \in \overline{A \setminus E}$. This proves that implication is valid and at least g is in support of this implication. More than that, since $A \setminus E \subseteq A$ the implication is not respected by A.

Now let $E \rightarrow f$ be a valid implication not respected by A with a non-empty support. Then $E \in A, f \notin A$ and there exists $g \in G$ such that $E, f \in g'$. By construction there exists $B \in \mathcal{MC}_A$ such that $E \subseteq A \cap g' \subseteq B$. Consider possible cases:

1. $f \in M$. As implication is valid and not respected by A, we have $f \in (A \cap g')'' \setminus A$. From monotonicity property it follows that $f \in B''$. It shows that $(B \rightarrow f) \in \mathcal{I}_A$.

2. $f \in \overline{M}$. Let $f = \overline{v}$. As implication is valid, there does not exist $\hat{g} \in G$ such that $v \in \hat{g}'$. Then $v \in A \setminus B$ and $(B \to f) \in \mathcal{I}_A$. □

Proposition 1 allows one to find an algorithm for computing the set of questions to an expert revealing possible errors of Types 1 and 2. The pseudocode is pretty straightforward.

6.1 Pseudocode

```
inspect($\mathbb{K}$:=(G, M, I), A$\subseteq$M)
1. if A$''$=A:
    2. return $\emptyset$
3. Candidates = {object$'\cap$A | object$\in$G}
4. Candidates = {C$\in$Candidates |
                 $\nexists$B$\in$Candidates: C$\subseteq$B}
5. Result = $\emptyset$
6. for Candidate in Candidates:
   7. Result.add({Candidate $\to$ d |
                 d$\in$(Candidate$''$\A $\cup$ $\overline{\text{A}\setminus\text{Candidate}}$)})
8. return Result
```

A is the intent of the new object. In the third line we compute the set of all subsets that can produce the desired implication. In the fourth line we discard all the non-maximal elements. In lines 6 and 7 we compute closures and add the corresponding implications.

If we try to add a new object intent such that it is not contained in any object intent already in the context, we should ask a new question to the previous new object intent. Indeed, now there may exist new valid implications with nonempty support. However, it is easy to see that such a question is exactly the question to the new object intent with negated conclusion. Indeed, let $B \to c$ be the implication representing the question to the new object. If it is rejected and the new object is added to the context, then the new object respects the implication $B \to \overline{c}$. As the implication $B \to c$ was asked, there were no object intents in the context respecting implication $B \to \overline{c}$. That is why the implication $B \to \overline{c}$ was never asked before and should be asked now. Finding such implications does not require any time and guarantees independence of the order of adding new object intents.
It is worth noting that considering only implications with non-empty support is not always safe. On the one hand, it allows one to avoid questions not based on any input information. On the other hand, this consideration does not allow one to state that there are no errors in an object. However, it suffices to check only maximal object intents in the context at any point in time, because only they may contain combinations of attributes not occurring elsewhere in the context. So, in order to avoid doubling the work, one should not consider implications with empty support, checking maximal object intents "by hand" whenever it is needed to show that the context is free from errors of Types 1 and 2.

7 Results

For the sake of compactness in this section we present implications in non-unit form. The name `inspect_dg` is used to denote the function implementing the first described approach (involving the canonical basis).

7.1 Example

Inspecting Error1:

`inspect_dg`
  at least 3 different angles $\rightarrow$ at least 3 different legs
  all legs equal $\rightarrow$ has equal angles, has equal legs

`inspect`
  has equal legs, at least 3 different angles $\rightarrow$ at least 3 different legs, $\overline{\text{all legs equal}}$
  has equal legs, all legs equal $\rightarrow$ has equal angles, $\overline{\text{at least 3 different angles}}$

Both algorithms reveal possible errors in a similar manner, although there are obvious differences. In the output of `inspect_dg` the premises are smaller than in the output of `inspect`. The latter also reveals dependencies of Type 2. It is easy to see that all output implications hold in data domain. For example, if all legs are equal in a quadrangle, it should have equal angles and should not have 3 different angles. As a corollary this object should be recognized as an error.

Inspecting Error2:

`inspect_dg`
  all angles equal $\rightarrow$ has equal angles, has equal legs, has right angle
  all legs equal $\rightarrow$ has equal angles, has equal legs

`inspect`
  has right angle, has equal legs, all legs equal, all angles equal $\rightarrow$ has equal angles

In this example we are able to ask even less number of questions to an expert using `inspect` as with `inspect_dg`. This is the result of finding implications generated by maximal subsets of object's intent. Again, all implications are valid in data domain. The intent of Error2 occurs in the context (in the intent of Square), that is why we do not get any negated attributes in the output of `inspect`.

Inspecting Error3:

`inspect_dg`

all angles equal $\rightarrow$ has equal angles, has equal legs, has right angle

all legs equal $\rightarrow$ has equal angles, has equal legs

`inspect`

has equal angles, has right angle, at least 3 different legs, at least 3 different angles $\rightarrow$ $\overline{\text{all angles equal}}$, $\overline{\text{all legs equal}}$

has equal angles, has right angle, all legs equal, all angles equal $\rightarrow$ has equal legs, $\overline{\text{at least 3 different angles}}$, $\overline{\text{at least 3 different legs}}$

In the case of Error3 we get both implications from the output of `inspect_dg` combined in one implication with a bigger premise in the output of `inspect`. In addition we obtain several implications with negated attributes. It is easy to see that all implications hold in the data domain.

Inspecting Error4:

`inspect_dg`

has equal angles, has equal legs, at least 3 different legs, all legs equal $\rightarrow$ has right angle, at least 3 different angles, all angles equal

`inspect`

has equal angles, has equal legs, all legs equal $\rightarrow$ $\overline{\text{at least 3 different legs}}$

has equal angles, has equal legs, at least 3 different legs $\rightarrow$ $\overline{\text{all legs equal}}$

Error4 is a very special case where the corresponding implication from canonical basis has empty support. In the output of `inspect_dg` we obtain all questions possible for this intent. As discussed above these questions are not based on any information input so far. Even if we add attributes "at least 3 different angles" and "all angles equal" and reject the last implication we would not be able to recognize this object as an error. On the contrary `inspect` allows us to recognize errors of Type 2.

7.2 Experiment

Below the results of tests on a bigger data are presented. The tests were conducted as follows: all objects one by one are first separated from a context and then added as a new object; all the possible errors of Type 1 and 2 are found and output for this object.

FCA package for Python was used for implementation ([8]). For computing the canonical basis an optimized algorithm based on `Next Closure` was used ([7]). All tests described below were run on computer with Intel Core i7 1.6GHz processor and 4 Gb of RAM running Linux Ubuntu 11.10 x64.

In Fig. 3 the results of running both algorithms on random contexts are presented. For each context the number of objects is equal to 50. Parameter d represents the density of the context, i.e. the probability of having cross in the cross-table representing the relation. This result is presented in the semi-logarithmic scale as the growth of complexity of computing the canonical basis is nearly exponential in this example. It is easy to note that with the growth of the number of attributes and the density the difference between runtime of two algorithms grows as well.

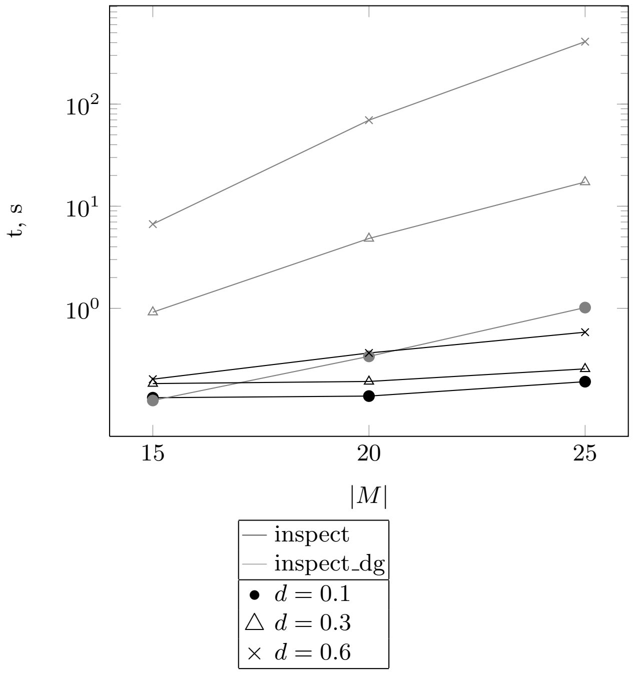


Fig. 3. Comparison of runtime on random contexts in semilog scale.

In Table 1 the results of running both algorithms on real data are presented. The data is taken from the UCI repository ([4]). In this tests algorithm `inspect` outperforms algorithm `inspect_dg` as well. Again, with the growth of the number of attributes the difference becomes more noticeable.

Context Name	$\|G\|$	$\|M\|$	inspect (s)	inspect_dg (s)
wine	178	68	4.674	13627.952
house-votes-84	435	18	1.048	64.735
SPECT	266	23	0.636	672.942

Table 1. Comparison of runtime on real data from UCI

8 Conclusion

An algorithm for finding errors of two types in new object intents is presented. As opposed to finding the canonical basis of the context the proposed algorithm terminates in polynomial time. Moreover, after checking only maximal object intents "by hand" it is possible to find all errors of two considered types (or prove their absence).

References

1. The qed project. http://mizar.org/qed/.
2. M. Babin and S. O. Kuznetsov. Recognizing pseudo-intent is conp-complete. *Proc. 7th International Conference on Concept Lattices and Their Applications, University of Sevilla*, pages 294–301, 2010.
3. Felix Distel and Barış Sertkaya. On the complexity of enumerating pseudo-intents. *Discrete Applied Mathematics*, 159(6):450–466, 2011.
4. A. Frank and A. Asuncion. UCI machine learning repository, 2010.
5. B. Ganter. Two basic algorithms in concept analysis. *Preprint-Nr. 831*, 1984.
6. J.-L. Guigues and V. Duquenne. Familles minimales d'implications informatives résultant d'un tableau de données binaires. *Math. Sci. Hum*, 24(95):5–18, 1986.
7. S. Obiedkov and V. Duquenne. Attribute-incremental construction of the canonical implication basis. *Annals of Mathematics and Artificial Intelligence*, 49(1-4):77–99, April 2007.
8. Nikita Romashkin. Python package for formal concept analysis. https://github.com/jupp/fca.
9. Uwe Ryssel, Felix Distel, and Daniel Borchmann. Fast computation of proper premises. In Amedeo Napoli and Vilem Vychodil, editors, *International Conference on Concept Lattices and Their Applications*, pages 101–113. INRIA Nancy – Grand Est and LORIA, 2011.

Adapting Fuzzy Formal Concept Analysis for Fuzzy Description Logics

Felix Distel

Institute of Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany, felix@tcs.inf.tu-dresden.de

Abstract. Fuzzy Logics have been applied successfully within both Formal Concept Analysis and Description Logics. Especially in the latter field, Fuzzy Logics have been gaining significant momentum during the last two years. Unfortunately, the research on fuzzy logics within the two communities has been conducted independently from each other, leading to different approaches being pursued. We show that if we look at a restricted variant of fuzzy formal concept analysis, then the differences between the two approaches can be reconciled. Moreover, an implicational base can be computed even when the identity hedge is used.

1 Introduction

In many applications one is forced to deal with vague knowledge, knowledge that does not fit into the binary world of classical logics. Among the countlessly many examples are the questions whether a country is large, or whether two cities are close to each other are difficult to answer with true or false. There are various degrees of size and proximity. Fuzzy Logics has successfully proposed to use a scale of truth degrees to describe vague knowledge. It has first been formalized for propositional logic [1] and has since been applied to many other logics and logic related formalisms. Among them are Formal Concept Analysis (FCA) [2] and Description Logics (DL) [3].

Whenever one applies Fuzzy Logics to an existing formalism, one is faced with several choices: Should the real unit interval be used for the set of truth degrees or a more complex lattice of truth degrees? How should the semantics of the conjunction be defined? Which parts of the existing theory should be replaced by their fuzzy counterparts and which should remain unchanged? These decisions have been made independently for fuzzy FCA and fuzzy DL.

In the crisp setting, a number of works have used FCA methods in DL. Some use it as a tool for efficiently computing concept hierarchies [4]. Others use it for ontology completion [5] and for exploring and learning from graph data [6, 7]. This work has been possible due to the close ties between FCA and DL. In FCA objects can be described using sets of attributes, and in DL individuals can be described using concept descriptions, in the easiest case conjunctions over concept names. Sets of attributes in FCA and conjunctions over concept names in DL share essentially the same semantics. While in the crisp case the similarities between fuzzy DL and fuzzy FCA are prominent, the situation is not so clear in the fuzzy variants of the respective theories. In fuzzy FCA one is allowed to use fuzzy sets of attributes. In fuzzy DL the same concept descriptions as in crisp DL are used. They are not fuzzy, only their semantics are. In Section 3 we identify such differences, that hinder the close cooperation that exists between the crisp variants of the two fields. We propose simple adjustments to avoid them. Generally speaking, one can say that the FCA community has been more ambitious and applied Fuzzy Logics to a much larger extent than fuzzy DL. Unfortunately, for this reason implication bases, which play an important role in the cooperation between crisp DL and crisp FCA, can no longer effectively be computed in the general case [8, 9].[1] We shall see in Section 4 that if we restrict expressivity of fuzzy FCA by considering only crisp sets of attributes, we can effectively compute bases.[2] Moreover, if the Gödel t-norm is used, this restricted version of fuzzy FCA is exactly the segment of fuzzy FCA whose semantics overlaps with fuzzy DL, presumably allowing synergies as in the crisp case.

The restriction to the Gödel t-norm is necessary, since fuzzy FCA uses weak conjunction for the semantics of attribute sets, while DL uses strong conjunction for its semantics. The two coincide only for the Gödel t-norm. From a current DL viewpoint, this is not a severe restriction, since up to now the Gödel t-norm is the only t-norm for which the standard DL reasoning tasks are known to be decidable [10].

2 Preliminaries

2.1 T-Norms, Hedges and Fuzzy Sets

Fuzzy Logics represent vague data while maintaining a well-defined semantics. Instead of using only the two values true and false a scale of *truth degrees* is used. In this work we consider only the most typical choice where truth degrees are values from the real unit interval [0, 1].

Fuzzy Logics provide several operators to define its semantics. A *t-norm* $\otimes$ is a binary operator $\otimes\colon [0,1] \times [0,1] \to [0,1]$ that is associative, commutative, monotone and has 1 as its unit. Every continuous t-norm gives rise to a binary operator $\Rightarrow\colon [0,1] \times [0,1] \to [0,1]$ that is the unique operator satisfying

$$z \leq x \Rightarrow y \text{ iff } x \otimes z \leq y \tag{1}$$

for all $z \in [0,1]$. The intuition is that the t-norm and the residuum be used to interpret conjunction and implication, respectively. Among the many continuous t-norms perhaps the simplest one, and the one we shall be interested in, is the Gödel t-norm. It is defined as $x \otimes y = \min\{x, y\}$ and its residuum is

$$x \Rightarrow y = \begin{cases} 1 & \text{if } x \leq y \\ y & \text{otherwise.} \end{cases}$$

[1] They can be computed if the globalization hedge is used. However, hedges do not exist in fuzzy DL and would even be problematic, as we shall see later.

[2] even if the globalization hedge is not used

A *hedge* $\cdot^*$ is a unary operator that is idempotent and satisfies $1^* = 1$, $a^* \leq a$, and $(a \Rightarrow b)^* \leq a^* \Rightarrow b^*$ for all $a, b \in [0,1]$. It is used for truth-stressing, i.e. to increase the contrast between 1 and the smaller truth values. A simple hedge is the *globalization*, defined as $1^* = 1$ and $a^* = 0$ for $a \neq 0$.

Fuzzy sets are a central idea of Fuzzy Logics. Given a set M a *fuzzy (sub-)set* T of M is a function $T\colon M \to [0,1]$, that maps each element of M to its membership degree in T. The *cardinality* of a fuzzy set T is defined as the cardinality of its support $\{x \in M \mid T(x) > 0\}$. Two fuzzy sets T_1 and T_2 can be compared pointwise by defining $T_1 \subseteq T_2$ iff $T_1(x) \leq T_2(x)$ for all $x \in M$. Alternatively, one can associate a subsethood degree with T_1 and T_2 by defining

$$S(T_1, T_2) = \inf_{x \in M} T_1(x) \Rightarrow T_2(x).$$

For finite fuzzy sets we use notation such as $\{{}^{0.5}/_{a}, {}^{1}/_{b}\}$ to denote the set that contains a with degree 0.5 and b with degree 1.

2.2 Formal Concept Analysis

The crisp setting We introduce crisp FCA in addition to fuzzy FCA, as we shall need the crisp version of the Duquenne-Guigues Base in the later sections. In crisp FCA [11], data is typically represented in the form of cross tables such as the one in Table 1. More formally, a *formal context* is a triple $\mathbb{K} = (G, M, I)$ where G is a set, called the set of *objects*, M is a set, called the set of *attributes*, and $I \subseteq G \times M$ is a binary relation, called the *incidence relation*. For sets $A \subseteq G$ and $B \subseteq M$ the derivation operators are defined as

$$A^{\uparrow} = \{m \in M \mid \forall g \in A\colon (g,m) \in I\},\ B^{\downarrow} = \{g \in G \mid \forall m \in B\colon (g,m) \in I\}. \quad (2)$$

The two derivation operators $\cdot^{\uparrow}$ and $\cdot^{\downarrow}$ form an antitone Galois-connection. An *implication* $A \to B$, where $A, B \subseteq M$, is said to hold in the context $\mathbb{K}$ if $A^{\downarrow} \subseteq B^{\downarrow}$. A set of attributes $U \subseteq M$ *respects* $A \to B$ iff $A \not\subseteq U$ or $B \subseteq U$. $A \to B$ *follows* from a set of implications $\mathcal{L}$ iff every set U that respects all implications from $\mathcal{L}$ also respects $A \to B$.

One way to structure the data in a formal context $\mathbb{K}$ is the *Duquenne-Guigues base* $\mathcal{DG}(\mathbb{K})$ [12]. $\mathcal{DG}(\mathbb{K})$ is a set of implication that is *sound* for $\mathbb{K}$, i.e. every implication from $\mathcal{DG}(\mathbb{K})$ holds in $\mathbb{K}$, *complete* for $\mathbb{K}$, i.e. every implication that holds in $\mathbb{K}$ follows from $\mathcal{DG}(\mathbb{K})$, and has *minimal cardinality* among all sound and complete sets of implications. A version that can handle background knowledge has been introduced in [13]. Given a sound set of implications $\mathcal{S}$ (the background knowledge) the *$\mathcal{S}$-Duquenne-Guigues base* $\mathcal{DG}_{\mathcal{S}}(\mathbb{K})$ is a set of implications such that $\mathcal{DG}_{\mathcal{S}}(\mathbb{K})$ is sound for $\mathbb{K}$, $\mathcal{S} \cup \mathcal{DG}_{\mathcal{S}}(\mathbb{K})$ is complete for $\mathbb{K}$ and $\mathcal{DG}_{\mathcal{S}}(\mathbb{K})$ has minimal cardinality [7]. The underlying mathematics of $\mathcal{DG}(\mathbb{K})$ and $\mathcal{DG}_{\mathcal{S}}(\mathbb{K})$ are not relevant for this work. It is, however, important, that both bases can effectively be computed for every finite context $\mathbb{K}$.

The Fuzzy Setting [2] In a fuzzy context $\mathbb{K} = (G, M, I)$ the incidence relation I is a fuzzy relation, i.e. a fuzzy subset of $G \times M$. The derivation operators are defined for fuzzy subsets A of G and fuzzy subsets B of M as follows:

$$A^{\uparrow}(m) = \inf_{g \in G} \big(A(g)^* \Rightarrow I(g,m)\big), \qquad B^{\downarrow}(g) = \inf_{m \in M} \big(B(m) \Rightarrow I(g,m)\big). \tag{3}$$

Notice, that the hedge $\cdot^*$ is used only for the derivation of fuzzy sets of objects. The operators $\cdot^{\uparrow}$ and $\cdot^{\downarrow}$ form a Galois connection with hedges.

In fuzzy FCA the implications are also allowed to be fuzzy. A *fuzzy implication* is a pair written as $A \to B$ where A and B are fuzzy subsets of M. Let U be a fuzzy subset of M. The *degree to which $A \to B$ holds in U* is defined as

$$\|A \to B\|_U = S(A, U)^* \Rightarrow S(B, U) \tag{4}$$

The *degree to which $A \to B$ holds in $\mathbb{K}$* is defined as $\|A \to B\|_{\mathbb{K}} = \min_{g \in G} \|A \to B\|_{I_g}$, where I_g is the fuzzy set to which each $m \in M$ belongs with degree $I(g,m)$. Let $\mathcal{L}$ be a fuzzy set of fuzzy implications. A set $U \subseteq M$ is called a model of $\mathcal{L}$ if $\|A \to B\|_U \geq \mathcal{L}(A \to B)$ holds for every fuzzy implication $A \to B$. We say that *$A \to B$ follows from $\mathcal{L}$ to degree q* if $\|A \to B\|_U \geq q$ for all models U of $\mathcal{L}$. There have been several works where the existence of bases for fuzzy implications has been considered [8, 9]. We shall not go into details, however, we would like to point out two things. First, it can be shown that it suffices to consider *crisp* sets $\mathcal{L}$ that contain *fuzzy* GCIs [14]. Second, in this setting an effective algorithm for computing a base is known only when globalization is used as the hedge [8].

2.3 Fuzzy Description Logics

For DL we only introduce the fuzzy version. The crisp version only occurs in a high-level description in Section 3.1. For a formal introduction of crisp DL we refer to [15]. DL is not just one formalism, but a family of many knowledge representation formalisms. The observations in this work hold for any fuzzy DL that provides for conjunction, i.e. virtually all of them. For brevity we only introduce the lightweight DL called $\mathcal{EL}$. In fuzzy $\mathcal{EL}$ (exactly like in crisp $\mathcal{EL}$) *concept descriptions* can be formed from a set of *concept names* $\mathcal{N}_C$ and a set of *role names* $\mathcal{N}_R$ using the constructors $\top$, $\sqcap$ and $\exists$. More formally, $\top$ and all concept names are concept descriptions, and if C and D are concept description and r is a role name then $C \sqcap D$ and $\exists r.C$ are also concept descriptions.

In fuzzy $\mathcal{EL}$ (in contrast to crisp $\mathcal{EL}$) fuzzy sets are used to interpret both concepts and roles. A fuzzy interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ satisfies

$$A^{\mathcal{I}} \colon \Delta^{\mathcal{I}} \to [0,1], \qquad r^{\mathcal{I}} \colon \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \to [0,1].$$

for all $A \in \mathcal{N}_C$ and all $r \in \mathcal{N}_R$. Fuzzy interpretations $\mathcal{I}$ are extended to complex concept descriptions by defining $\top^{\mathcal{I}}(x) = 1$ and

$$(C \sqcap D)^{\mathcal{I}}(x) = C^{\mathcal{I}}(x) \otimes D^{\mathcal{I}}(x), \qquad (\exists s.C)^{\mathcal{I}}(x) = \sup_{z \in \Delta^{\mathcal{I}}} s^{\mathcal{I}}(x,z) \otimes C^{\mathcal{I}}(z) \tag{5}$$

for all $x \in \Delta^{\mathcal{I}}$. Fuzzy GCIs are typically written as $\langle C \sqsubseteq D, q\rangle$, where C, D are concept descriptions and $q \in [0,1]$. The fuzzy GCI $\langle C \sqsubseteq D, q\rangle$ holds in the fuzzy interpretation $\mathcal{I}$ if all $x \in \Delta^{\mathcal{I}}$ satisfy $C^{\mathcal{I}}(x) \Rightarrow D^{\mathcal{I}}(x) \geq q$. The fuzzy interpretation $\mathcal{I}$ is a *model* of the set of fuzzy GCIs $\mathcal{T}$ if all fuzzy GCIs from $\mathcal{T}$ hold in $\mathcal{I}$. $\langle C \sqsubseteq D, q\rangle$ *is entailed by* $\mathcal{T}$ if it holds in all models of $\mathcal{T}$.

3 Comparison of the Two Formalisms

3.1 The Crisp Setting

Most existing works at the intersection of FCA and DL have in common that they associate FCA attributes and DL concept names. The objects are usually chosen to be domain elements of an interpretation [6, 7]. Other choices, such as selecting ABox individuals, usually require extending FCA theory, e.g. to allow for partial knowledge [5]. These choices are motivated by the following observation. Whether we compute the interpretation of the concept description $\mathsf{Large} \sqcap \mathsf{Populous} \sqcap \mathsf{Asian}$ or compute the derivation of the set of attributes $\{\mathsf{Large}, \mathsf{Populous}, \mathsf{Asian}\}$, the intuition in both cases is that we want to know which countries are large *and* populous *and* Asian.

To formalize this connection, for every interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ one can define its *induced context* $\mathbb{K}_{\mathcal{I}}$ whose set of objects is $\Delta^{\mathcal{I}}$, whose attributes are the concept names and where $x \in \Delta^{\mathcal{I}}$ and $A \in \mathcal{N}_C$ are incident iff $x \in A^{\mathcal{I}}$. For example, we can think of the context in Table 1 as being induced by an interpretation $\mathcal{I}$ whose domain are the world's 8 most populous countries, and where the concept names Populous, Large and Asian are interpreted as $\mathsf{Populous}^{\mathcal{I}} = \{\mathsf{China}, \mathsf{India}\}$, $\mathsf{Large}^{\mathcal{I}} = \{\mathsf{China}, \mathsf{Russia}, \mathsf{US}\}$, and $\mathsf{Asian}^{\mathcal{I}} = \{\mathsf{China}, \mathsf{India}, \mathsf{Indonesia}, \mathsf{Pakistan}, \mathsf{Russia}\}$.

In the induced context it holds for all sets $U \subseteq \mathcal{N}_C$ that $U^{\downarrow} = (\bigsqcap U)^{\mathcal{I}}$, further supporting the intuition that sets of attributes are treated like conjunctions over attributes. Similarly, for two sets of concept names $U, V \subseteq \mathcal{N}_C$ the GCI $\bigsqcap U \sqsubseteq \bigsqcap V$ holds in the interpretation $\mathcal{I}$ iff the implication $U \to V$ holds in the induced context of $\mathcal{I}$. Hence, the notions of dependencies also coincide in crisp FCA and crisp DL. One could even go so far to say that standard formal contexts and the very simple DL that only allows for conjunction are syntactic variants of each other.

3.2 The Fuzzy Setting

In this section, we analyze the differences between fuzzy FCA and fuzzy DL that hinder a close cooperation like it exists in the crisp setting. First, *the semantics of fuzzy FCA do not treat sets of attributes like conjunctions over concept names.* Remember that in the crisp setting the exact same semantics are used to compute the derivation of a set of attributes or the interpretation of a conjunction of concept names. This is not true in the fuzzy setting because the infimum (or minimum in the case of finite contexts) is used to interpret attribute sets (2) while

Table 1. Induced Context

	Large	Populous	Asian
Brazil			
China	×	×	×
India		×	×
Indonesia			×
Nigeria			
Pakistan			×
Russia	×		×
US	×		

Table 2. Induced Fuzzy Context

	Large	Populous	Asian
Brazil	0.5	0.14	0.0
China	0.56	1.0	1.0
India	0.19	0.9	1.0
Indonesia	0.11	0.18	0.76
Nigeria	0.05	0.13	0.0
Pakistan	0.05	0.13	1.0
Russia	1.0	0.11	0.75
US	0.58	0.23	0.0

the t-norm is used to interpret conjunctions (5).[3] In the case of the Gödel t-norm this is completely harmless, as the Gödel t-norm coincides with the minimum. For the other t-norms the difference is relevant.

As an example, assume that Table 2 is obtained from a fuzzy interpretation $\mathcal{I}$ by using the domain as objects, concept names as attributes and defining $I(x, A) = A^{\mathcal{I}}(x)$ (We could call it the *induced fuzzy context* of $\mathcal{I}$). If we use the Łukasiewicz t-norm, which is defined as $x \otimes y = \max\{0, x + y - 1\}$, then

$$(\mathsf{Populous} \sqcap \mathsf{Large})^{\mathcal{I}}(\mathsf{US}) = 0.23 \otimes 0.58 = 0$$

However, in fuzzy FCA with the Łukasiewicz t-norm we obtain

$$\{{}^{1}\!/\mathsf{Populous}, {}^{1}\!/\mathsf{Large}\}^{\downarrow}(\mathsf{US}) = \min\{1 \Rightarrow 0.23, 1 \Rightarrow 0.58\} = 0.23.$$

Thus, unlike in the crisp case, the fuzzy semantics differ even for crisp sets of attributes such as $\{{}^{1}\!/\mathsf{Populous}, {}^{1}\!/\mathsf{Large}\}$.

Second, we can observe that *in fuzzy DL concept descriptions on their own are not fuzzy.* The interpretations are fuzzy, the axioms are fuzzy, but the concept descriptions themselves are not. By contrast, in fuzzy FCA it is possible to use a fuzzy set of attributes to describe a class of objects.

To describe all countries that are (completely) huge and somewhat Asian, one can use a fuzzy set of attributes $\{{}^{1}\!/\mathsf{Large}, {}^{0.5}\!/\mathsf{Asian}\}$. Then in Table 2 the membership of Russia in the derivation $\{{}^{1}\!/\mathsf{Large}, {}^{0.5}\!/\mathsf{Asian}\}^{\downarrow}$ is 1. By contrast, in DL it is not possible to associate a truth degree with the concepts in a conjunction. The best approximation of the above attribute set in DL is the simple conjunction $\mathsf{Large} \sqcap \mathsf{Asian}$, which has, of course, a different semantics. In fact, Russia belongs to $(\mathsf{Large} \sqcap \mathsf{Asian})^{\mathcal{I}}$ only with degree 0.75.[4] In this respect fuzzy FCA is more expressive than fuzzy DL.

Finally, *fuzzy FCA typically uses hedges and fuzzy DL does not.* In principle, fuzzy FCA is more general here, since one could treat fuzzy DL as the special

[3] Some authors use two types of conjunction: a strong conjunction interpreted by the t-norm and a weak conjunction interpreted by the minimum. In this terminology, we could write that fuzzy DL uses strong conjunction while fuzzy FCA uses weak conjunction.

[4] The Gödel t-norm is used to emphasize that this is independent of the first problem.

case where identity is used as the hedge. In practice, if identity is used as the hedge, one cannot effectively compute a base in fuzzy FCA, at least not in the settings that have previously been considered.

On the other hand, using globalization in combination with crisp sets of attributes has practical limitations. Consider a fuzzy implication $A \to B$. Using the globalization as the hedge means, that all those counterexamples $g \in G$ are ignored that do not satisfy $S(A, I_g) = 1$. This is particularly problematic, if we only consider crisp left-hand sides A, since then

$$S(A, I_g) = \min_{m \in A}(1 \Rightarrow I(g,m)) = \min_{m \in A} I(g,m). \tag{6}$$

If for just one $m \in A$ the value $I(g,m)$ is not 1 then $S(A, I_g) < 1$ holds and the object g is ignored. For example, in Table 2 if we consider $A = \{1/\mathsf{Large}, 1/\mathsf{Populous}\}$ then all objects are ignored, i.e. any implication with A as its left-hand side holds. Presumably, in many applications values that differ from 1 are the rule rather than the exception, meaning that almost all objects will be ignored.

4 Bridging the Gap

In the previous section we have identified the three aspects in which fuzzy DL and fuzzy FCA disagree. We shall now consider a restricted subset of fuzzy FCA for which the semantics agree. Unfortunately, it is not possible to use strong conjunction instead of weak conjunction in fuzzy FCA, since the derivation operators would no longer form a Galois-connection. Instead, we caution that the following theory can only be applied directly in fuzzy DL with Gödel t-norm.

Since truly fuzzy implications have no equivalent in fuzzy DL, we only consider implications $A \to B$, where both sets A and B are crisp (from now on called *crisp implications*). A similar idea has been proposed under the name of "one-sided fuzzyness" in [16], with respect to concept lattices, not with respect to bases. Instead of trying to compute a base that is complete for all fuzzy implications we try to find a base that is complete only for crisp implications. In standard fuzzy FCA there is a result, that allows one to consider only crisp sets of fuzzy implications when searching for a base (Lemma 1 in [14]). Unfortunately, this result cannot be applied in our restricted setting. Instead of computing a crisp set containing fuzzy implications we compute a fuzzy set containing crisp implications:

Problem 1. Given a fuzzy context $\mathbb{K} = (G, M, I)$ compute a fuzzy subset T of $\{A \to B \mid A, B \subseteq M\}$ that is

- complete, i.e. for every implication $A \to B$, $A, B \subseteq M$, $\|A \to B\|_{\mathbb{K}} = q$ implies that $A \to B$ follows from T with degree q,
- sound, i.e. every implication $A \to B$ holds in $\mathbb{K}$ with degree at least $T(A \to B)$, and
- irredundant, i.e. no fuzzy set $U \subsetneq T$ is complete.

Furthermore, we use identity as the hedge, thereby ensuring both compatibility with DL and the use of all objects as potential counterexamples. These three restrictions – Gödel t-norm, identity as the hedge, only crisp implications – guarantee that $A \to B$ holds in the fuzzy induced context $\mathbb{K}$ of $\mathcal{I}$ to degree q iff $\langle \bigsqcap A \sqsubseteq \bigsqcap B, q\rangle$ holds in $\mathcal{I}$. This is analogous to the crisp case.

4.1 Axiomatization

In [14] an axiomatic system is presented that can be used to infer all fuzzy implications that follow from a crisp set of fuzzy implications. We present a similar system of deduction rules, which can be used to infer for each crisp implication the degree to which it follows from a fuzzy set of crisp implications.

Let $\mathcal{L}$ be a fuzzy subset of $\{A \to B \mid A, B \subseteq M\}$. Our axiomatic system consists of the following deduction rules, where q_1, q_2 are positive truth values. In each deduction step a new fuzzy subset $\mathcal{L}_{i+1}$ is obtained from the previous set $\mathcal{L}_i$, where $\mathcal{L}_0 = \mathcal{L}$. For all implications $E \to F$ we define $\mathcal{L}_{i+1}(E \to F) = \mathcal{L}_i(E \to F)$ unless mentioned otherwise in the rules.

(Refl) From $A \subseteq B$ and $\mathcal{L}_i(A \to B) < 1$ infer $\mathcal{L}_{i+1}(A \to B) = 1$
(Union) From $\mathcal{L}_i(A \to B) = q_1$, $\mathcal{L}_i(A \to C) = q_2$ and $\mathcal{L}_i(A \to B \cup C) < \min\{q_1, q_2\}$ infer $\mathcal{L}_{i+1}(A \to B \cup C) = \min\{q_1, q_2\}$
(Trans) From $\mathcal{L}_i(A \to B) = q_1$, $\mathcal{L}_i(B \to C) = q_2$ and $\mathcal{L}_i(A \to C) < q_1 \otimes q_2$ infer $\mathcal{L}_{i+1}(A \to C) = q_1 \otimes q_2$.

In each of the three rules the inferred implication obtains a membership degree that is smaller or equal to the membership degrees of the rules in the precondition. Since a rule can only be applied if the degree of the inferred implication strictly increases, no implication can ever be used in its own deduction implicitly or explicitly. There are only finitely many crisp implications and therefore the deduction process must terminate. We now want to show that the deduction system is *sound*, in the sense that if after a finite number k of deduction steps we can deduce $\mathcal{L}_k(A \to B) = q$ then $A \to B$ follows from $\mathcal{L}$ with at least degree q, and *complete* in the sense that if $A \to B$ follows from $\mathcal{L}$ with degree q then $\mathcal{L}_k(A \to B) = q$ can be deduced.

Lemma 1. *(Refl)–(Trans) is a sound and complete system of deduction rules.*

Proof. To prove soundness, we prove that each rule application does not change the models, i.e. that every model U of $\mathcal{L}_i$ is a model of $\mathcal{L}_{i+1}$. The converse that every model of $\mathcal{L}_{i+1}$ is a model of $\mathcal{L}_i$ is trivial, since $\mathcal{L}_i \subseteq \mathcal{L}_{i+1}$. Soundness of (Refl) is also trivial. *Soundness of (Union):* Assume that U is a model of $\mathcal{L}_i$. We define $\alpha = \min_{m\in A} U(m)$, $\beta = \min_{m\in B} U(m)$ and $\gamma = \min_{m\in C} U(m)$. From (6) and (4) we obtain

$$\|A \to B\|_U = \alpha \Rightarrow \beta,\ \|A \to C\|_U = \alpha \Rightarrow \gamma,\ \|A \to B \cup C\|_U = \alpha \Rightarrow \min\{\beta, \gamma\}.$$

Monotonicity of the residuum yields $\|A \to B \cup C\|_U = \min\{\alpha \Rightarrow \beta, \alpha \Rightarrow \gamma\} \geq \min\{q_1, q_2\}$. This proves that U is also a model of $\mathcal{L}_{i+1}$, which suffices to prove soundness of (Union). *Soundness of (Trans):* Since U is a model of $\mathcal{L}_i$ we obtain from the preconditions $\alpha \Rightarrow \beta \geq q_1$ and $\beta \Rightarrow \gamma \geq q_2$. Using (1) we obtain $\alpha \otimes q_1 \leq \beta$ and $\beta \otimes q_2 \leq \gamma$. From monotonicity of the t-norm we obtain $\alpha \otimes (q_1 \otimes q_2) \leq \gamma$. Using (1) again we get $q_1 \otimes q_2 \leq \alpha \Rightarrow \gamma = \|A \to C\|_U$. Hence, U is a model of $\mathcal{L}_{i+1}$, which proves soundness of (Trans).

Completeness: Let $X \to Y$ be an implication that follows (semantically) from $\mathcal{L}$ to degree q. Let $\mathcal{L}_k$ be the fuzzy set of implications obtained after exhaustively applying the deduction rules. To prove completeness it suffices to show that that $\mathcal{L}_k(X \to Y) \geq q$.

As a preliminary step, let us define the following fuzzy set $X^+(m) = \mathcal{L}_k(X \to \{m\})$ and show that it is a model of $\mathcal{L}$. Assume that X^+ is not a model of $\mathcal{L}$, i.e. $\|A \to B\|_{X^+} < \mathcal{L}(A \to B)$ for some implication $A \to B$. We use the notation $\alpha = \min_{m \in A} X^+(m)$ and $\beta = \min_{m \in B} X^+(m)$. Then $\|A \to B\|_{X^+} = \alpha \Rightarrow \beta < \mathcal{L}(A \to B)$, or equivalently by (1)

$$\alpha \otimes \mathcal{L}(A \to B) > \beta. \tag{7}$$

On the other hand $X^+(a) = \mathcal{L}_k(X \to \{a\}) \geq \alpha$ holds for all $a \in A$. Because the rules have been applied exhaustively to obtain $\mathcal{L}_k$ (Union) is not applicable to $\mathcal{L}_k$ and therefore $\mathcal{L}_k(X \to A) \geq \alpha$. Using a similar argument for (Trans) we obtain

$$\mathcal{L}_k(X \to B) \geq \mathcal{L}_k(X \to A) \otimes \mathcal{L}_k(A \to B) \geq \alpha \otimes \mathcal{L}(A \to B),$$

where we have exploited the fact that truth values can only increase when a rule is applied and therefore $\mathcal{L}(A \to B) \leq \mathcal{L}_k(A \to B)$. Finally, using (Refl) and (Trans) it follows that $\mathcal{L}_k(X \to \{b\}) \geq \alpha \otimes \mathcal{L}(A \to B)$ for all $b \in B$. This contradicts (7) and thus X^+ must be a model of $\mathcal{L}$.

Since $X \to Y$ follows from $\mathcal{L}$ to degree q it must hold that

$$q \leq \|X \to Y\|_{X^+} = \big(\min_{x \in X} X^+(x) \Rightarrow \min_{y \in Y} X^+(y)\big) = \min_{y \in Y} X^+(y).$$

Therefore $\mathcal{L}_k(X \to \{y\}) = X^+(y) \geq q$ for all $y \in Y$. Since (Union) cannot be applied to $\mathcal{L}_k$ we obtain $\mathcal{L}_k(X \to Y) \geq q$ which proves completeness. □

4.2 Stem Base

We now provide a practical approach for computing a finite base for the Gödel t-norm, the only t-norm for which the semantics of fuzzy FCA and fuzzy DL coincide. Assume that we are given a finite fuzzy context $\mathbb{K} = (G, M, I)$. Let $Q_\mathbb{K}$ be the set containing 1 and all truth degrees that occur in $\mathbb{K}$. Let $q_0 \in [0, 1]$ be a fixed truth degree. We define a crisp context $\mathbb{K}_{q_0} = (G_{q_0}, M, I_{q_0})$ as follows. For each $g \in G$ and each $q \in Q_\mathbb{K}$ with $q < q_0$ the set G_{q_0} contains an object g_q with

$$\{g_q\}' = \{m \in M \mid I(g, m) > q\},$$

i.e. g_q has exactly those attributes that g has with degree higher than q. As an example, consider a context $\mathbb{K}$ of South American Countries (Table 3).[5]

[5] The value for HighGDP is the fraction of the country's GDP per capita and the GDP per capita of Chile, the largest in South America. Similarly for the other values.

Algorithm 1 Computing a Minimal Base with Gödel t-Norm

```
$\mathcal{B} = \mathcal{L} = \emptyset$
for all $q \in Q_{\mathbb{K}}$ in decreasing order do
    $\mathcal{D} = \mathcal{DG}_{\mathcal{B}}(\mathbb{K}_q)$
    $\mathcal{B} = \mathcal{B} \cup \mathcal{D}$
    $\mathcal{L} = \mathcal{L} \cup \{{}^{q}/_{A \to B} \mid A \to B \in \mathcal{D}\}$
end for
return $\mathcal{L}$
```

Lemma 2. *$A \to B$ holds in $\mathbb{K}$ with at least degree q_0 iff $A \to B$ holds in $\mathbb{K}_{q_0}$.*

Proof. Assume that $A \to B$ holds in $\mathbb{K}$ with degree less than q_0. According to (4) and the definition of the Gödel-residuum this is equivalent to

$$\begin{aligned}
&\min_{g \in G} \|A \to B\|_{I_g} < q_0 \\
\iff &\exists g \in G\colon \left(\min_{a \in A} I(g,a) \Rightarrow \min_{b \in B} I(g,b)\right) < q_0 \\
\iff &\exists g \in G\colon \min_{b \in B} I(g,b) < q_0 \text{ and } \min_{a \in A} I(g,a) > \min_{b \in B} I(g,b) \\
\iff &\exists g \in G\colon \exists b \in B\colon I(g,b) < q_0 \text{ and } \forall a \in A\colon I(g,a) > I(g,b).
\end{aligned} \tag{8}$$

$I(g,b)$ is a truth degree from $Q_{\mathbb{K}}$ and $g_{I(g,b)}$ satisfies $(g_{I(g,b)}, b) \notin I_{q_0}$ and $(g_{I(g,b)}, a) \in I_{q_0}$ for all $a \in A$. Therefore, $\mathbb{K}_{q_0}$ contains a counterexample to $A \to B$, hence $A \to B$ does not hold in $\mathbb{K}_{q_0}$.

On the other hand if $A \to B$ does not hold in $\mathbb{K}_{q_0}$ then there must be some g_q, $q < q_0$ such that $A \subseteq \{g_q\}'$ and $B \not\subseteq \{g_q\}'$. By definition of $\{g_q\}'$ this is equivalent to $I(g_q, a) > q$ for all $a \in A$ and $I(g_q, b) \leq q$ for some $b \in B$. Since $q < q_0$ it holds that for this value b in particular $I(g_q, b) < q_0$ and $I(g_q, a) > I(g_q, b)$ for all $a \in A$. It then follows from (8) that $A \to B$ does not hold in $\mathbb{K}$ with at least degree q_0. □

Notice, that if $q_1 < q_0$ then $G_{q_1} \subseteq G_{q_0}$ and $I_{q_1} \subseteq I_{q_0}$. This observation, together with Lemma 2, suggests a levelwise approach as sketched in Algorithm 1. One starts with the largest value $q_{\max}$ in $Q_{\mathbb{K}}$ and computes the Duquenne-Guigues Base for $\mathbb{K}_{q_{\max}}$. The base serves two purposes. Its implications are added to the fuzzy set of implications $\mathcal{L}$ with degree q, and it serves as background knowledge in the next iteration. For the context from Table 3 Algorithm 1 yields the base $\{{}^{1}/_{\{\mathsf{Populous},\mathsf{Small}\} \to \{\mathsf{HighGDP}\}}, {}^{0.9}/_{\{\mathsf{Populous}\} \to \{\mathsf{HighGDP}\}}, {}^{0.2}/_{\emptyset \to \{\mathsf{HighGDP}\}}\}$.

Lemma 3. *Upon termination Algorithm 1 returns a fuzzy set of crisp implications $\mathcal{L}$ that is sound and complete for $\mathbb{K}_q$ and has minimal cardinality among all such sets.*

Proof. Soundness follows immediately from Lemma 8. To prove completeness, assume that $U \to V$ holds in $\mathbb{K}$ with degree $q \in Q_{\mathbb{K}}$ (notice that for the Gödel t-norm it always holds that $\|U \to V\|_{\mathbb{K}} \in Q_{\mathbb{K}}$). Then by Lemma 8 $U \to V$ holds

Table 3. South American Countries

	Populous	HighGDP	Small
Argentina	0.2	0.8	0.6
Bolivia	0.1	0.2	0.9
Brazil	1.0	0.9	0.0
Chile	0.1	1.0	0.9
Colombia	0.2	0.5	0.9
Ecuador	0.1	0.3	1.0
Guyana	0.0	0.2	1.0
Paraguay	0.0	0.2	1.0
Suriname	0.0	0.5	1.0
Uruguay	0.0	1.0	1.0
Venezuela	0.1	0.7	0.9

Table 4. $\mathbb{K}_1$

	Populous	HighGDP	Small
Argentina$_{0.6}$		×	
Argentina$_{0.2}$		×	×
Bolivia$_{0.2}$			×
Bolivia$_{0.1}$		×	×
Brazil$_{0.9}$	×		
Brazil$_{0.0}$	×	×	
Chile$_{0.9}$		×	
Chile$_{0.1}$		×	×
Colombia$_{0.5}$			×
Colombia$_{0.2}$		×	×
Ecuador$_{0.3}$			×
Ecuador$_{0.1}$		×	×
Guyana$_{0.2}$			×
Guyana$_{0.0}$		×	×
Paraguay$_{0.2}$			×
Paraguay$_{0.0}$		×	×
Suriname$_{0.5}$			×
Suriname$_{0.0}$		×	×
Uruguay$_{0.0}$		×	×
Venezuela$_{0.7}$			×
Venezuela$_{0.1}$		×	×

in $\mathbb{K}_q$. Consider $\mathcal{D}$, $\mathcal{V}$ and $\mathcal{L}$ after the iteration for q in Algorithm 1. Since $\mathcal{D} \cup \mathcal{B}$ is complete for $\mathbb{K}_q$ and $U \to V$ holds in $\mathbb{K}_q$ the implication $U \to V$ follows from $\mathcal{D} \cup \mathcal{B} = \{A \to B \mid \mathcal{L}(A \to B) \geq q\}$ in the crisp setting.

We show that then $U \to V$ follows to degree q from $\mathcal{L}$ in the fuzzy setting. Assume the contrary, i.e. that there exists a context $\bar{\mathbb{K}}$ for which $\mathcal{L}$ is sound, but in which $U \to V$ does not hold to degree at least q. By Lemma 8 this yields that $U \to V$ does not hold in $\bar{\mathbb{K}}_q$ while all implications from $\mathcal{D} \cup \mathcal{B} = \{A \to B \mid \mathcal{L}(A \to B) \geq q\}$ do hold in $\bar{\mathbb{K}}_q$. Because we have shown that $U \to V$ follows from $\mathcal{D} \cup \mathcal{B}$ in the crisp setting this is a contradiction. Hence $U \to V$ follows to degree q from $\mathcal{L}$, which proves completeness.

Assume that $\bar{\mathcal{L}}$ is another sound and complete fuzzy set of implications for $\mathbb{K}$. Then by Lemma 8 for each $q \in Q_{\mathbb{K}}$ the crisp set

$$\bar{\mathcal{L}}_q = \{A \to B \mid \mathcal{L}(A \to B) \geq q\}$$

must be sound and complete for $\mathbb{K}_q$. A simple induction over $q \in Q_{\mathbb{K}}$ can be used to show that $|\bar{\mathcal{L}}_q| \geq |\mathcal{L}_q|$ for all $q \in Q_{\mathbb{K}}$. For q maximal in $Q_{\mathbb{K}}$ the claim follows directly from minimality of the Duquenne-Guigues Base. For the induction step let $q \in Q_{\mathbb{K}}$ where $|\bar{\mathcal{L}}_{\bar{q}}| \geq |\mathcal{L}_{\bar{q}}|$ holds for the next larger value $\bar{q} \in Q_{\mathbb{K}}$. Both $\mathcal{L}_{\bar{q}}$ and $\bar{\mathcal{L}}_{\bar{q}}$ are sound and complete for $\mathbb{K}_{\bar{q}}$, in particular they have the same models. Thus, $\bar{\mathcal{L}}_q = \big(\bar{\mathcal{L}}_q \setminus \bar{\mathcal{L}}_{\bar{q}}\big) \cup \bar{\mathcal{L}}_{\bar{q}}$ and $\big(\bar{\mathcal{L}}_q \setminus \bar{\mathcal{L}}_{\bar{q}}\big) \cup \mathcal{L}_{\bar{q}}$ also have the same models, and are thus both sound and complete for $\mathbb{K}_q$. Minimality of the $\mathcal{L}_{\bar{q}}$-Duquenne-Guigues base implies $|\bar{\mathcal{L}}_q \setminus \bar{\mathcal{L}}_{\bar{q}}| \geq |\mathcal{DG}_{\mathcal{L}_{\bar{q}}}(\mathbb{K}_q)|$. This proves

$$|\bar{\mathcal{L}}_q| = |\big(\bar{\mathcal{L}}_q \setminus \bar{\mathcal{L}}_{\bar{q}}\big)| + |\bar{\mathcal{L}}_{\bar{q}}| \geq |\mathcal{DG}_{\mathcal{L}_{\bar{q}}}(\mathbb{K}_q)| + |\mathcal{L}_{\bar{q}}| = |\mathcal{L}_q|.$$

Since this holds for all $q \in Q_{\mathbb{K}}$ we get that $\mathcal{L}$ has minimal cardinality among all bases. □

5 Conclusion

We have restricted fuzzy FCA by allowing only crisp sets of attributes in the implications and using identity as the hedge. We have presented a sound and complete set of deduction rules for this restricted setting. For the Gödel t-norm the restricted setting corresponds semantically to fuzzy DL. Furthermore, we have presented a simple algorithm for computing a minimal base for the restricted setting. In the general setting this is only possible with globalization.

We do not claim, that this restriction of expressivity is the only feasible approach for reconciling the differences between the two fields. In future work it would be interesting to look at a kind of weighted conjunction in DL (imitating the semantics of fuzzy attribute sets). It would also be interesting to consider a version of fuzzy FCA that uses strong conjunction.

References

1. Hájek, P.: Metamathematics of Fuzzy Logic (Trends in Logic). Springer (2001)
2. Belohlavek, R.: Fuzzy Relational Systems: Foundations and Principles. Kluwer (2002)
3. García-Cerdaña, Á., Armengol, E., Esteva, F.: Fuzzy description logics and t-norm based fuzzy logics. Int. J. of Approx. Reasoning **51** (2010) 632–655
4. Sertkaya, B.: Computing the hierarchy of conjunctions of concept names and their negations in a description logic knowledge base using formal concept analysis. In: Cont. to ICFCA'06, Dresden, Germany (2006) 73–86
5. Baader, F., Ganter, B., Sattler, U., Sertkaya, B.: Completing Description Logic knowledge bases using Formal Concept Analysis. In: IJCAI'07. (2007)
6. Rudolph, S.: Exploring relational structures via FLE. In: Conceptual Structures at Work. Springer (2004) 233–233
7. Distel, F.: Learning Description Logic Knowledge Bases from Data Using Methods from Formal Concept Analysis. PhD thesis, TU Dresden (2011)
8. Belohlavek, R., Vychodil, V.: Attribute implications in a fuzzy setting. In: ICFCA'06. Springer (2006) 45–60
9. Belohlavek, R., Chlupova, M., Vychodil, V.: Implications from data with fuzzy attributes. In: AISTA'04. (2004)
10. Bobillo, F., Delgado, M., Gómez-Romero, J., Straccia, U.: Fuzzy description logics under Gödel semantics. Int. J. of Approx. Reasoning **50**(3) (2009) 494–514
11. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, New York (1997)
12. Guigues, J.L., Duquenne, V.: Familles minimales d'implications informatives résultant d'un tableau de données binaires. Math. Sci. Humaines **95** (1986) 5–18
13. Stumme, G.: Attribute exploration with background implications and exceptions. In: Data Analysis and Inf. Sys., Berlin, Springer (1996) 457ff
14. Belohlavek, R., Vychodil, V.: Axiomatizations of fuzzy attribute logic. In: IICAI'05. (2005)
15. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F., eds.: The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)
16. Krajči S.: Cluster based efficient generation of fuzzy concepts. Neural Network World **5** (2003), 521–530

Computing Functional Dependencies with Pattern Structures

Jaume Baixeries[1], Mehdi Kaytoue[2], and Amedeo Napoli[3]

[1] Departament de Llenguatges i Sistemes Informàtics. Universitat Politècnica de Catalunya. 08032 Barcelona. Catalonia.
`jbaixer@lsi.upc.edu`
[2] Université de Lyon. CNRS, INSA-Lyon, LIRIS. UMR5205, F-69621, France.
`mehdi.kaytoue@insa-lyon.fr`
[3] LORIA, B.P. 70239, Équipe Orpailleur, Bâtiment B, F-54506 Vandœuvre-lès-Nancy
`amedeo.napoli@loria.fr`

Abstract. The treatment of many-valued data with FCA has been achieved by means of scaling. This method has some drawbacks, since the size of the resulting formal contexts depends usually on the number of different values that are present in a table, which can be very large. Pattern structures have been proved to deal with many-valued data, offering a viable and sound alternative to scaling in order to represent and analyze sets of many-valued data with FCA.
Functional dependencies have already been dealt with FCA using the binarization of a table, that is, creating a formal context out of a set of data. Unfortunately, although this method is standard and simple, it has an important drawback, which is the fact that the resulting context is quadratic in number of objects w.r.t. the original set of data.
In this paper, we examine how we can extract the functional dependencies that hold in a set of data using pattern structures. This allows to build an equivalent concept lattice avoiding the step of binarization, and thus comes with better concept representation and computation.

Keywords: Association rules and data dependencies, attribute implications, data dependencies, pattern structures, formal concept analysis

1 Introduction and Motivation

In the relational database model there are different types of dependencies ([1,18]). Functional dependencies are among the most popular. The reason is that they are important in order to explain the normalization of a database scheme in the Relational Database Model. Functional Dependencies (FD's) have their own set of axioms ([5,18]), which, in turn, are also shared by other dependencies. For instance, implications share the same axioms as functional dependencies ([2]), which are basically reflexivity, augmentation and transitivity.

These axioms state how functional dependencies behave in the presence of a set of dependencies of the same kind. For instance, we can decide whether a set of functional dependencies Σ implies a single FD σ, that is, $\Sigma \models \sigma$

We can also find a minimal set of functional dependencies that implies a given set of them, that is, we can compute Σ' such that $\Sigma' \models \Sigma$, where Σ' is minimal. In this case, we also say that Σ' is the minimal generating set of Σ. These two problems have been studied in [1] and [16], and algorithms have been proposed. Yet, it is important to note that this calculation is performed starting from a set of dependencies, not a set of data.

In this paper, we aim at finding a characterization of functional dependencies that hold in a set of data using Formal Concept Analysis and pattern structures.

The lattice characterization of a set of Functional Dependencies is studied in [6,7,8,9], and the characterization with a formal context in [3,12]. This characterization is based on a *binarization*, which is the transformation of the original set of data into a binary context.

In fact, the primary concern when computing the characterization of a set of functional dependencies with FCA is that, generally, the dataset is many-valued, and not binary. This means that the set of data must be somehow transformed to obtain a binary context.

Applying conceptual scaling (without information loss) results either in a larger set of objects in the resulting formal context, or a larger set of attributes.

On another hand, pattern structures ([11,14]) have emerged as a valid alternative to work with non binary contexts and specially with numerical contexts, as well as to avoid the complexity drawbacks that are present in scaling.

Therefore, we have two different methods of computing the characterization of a set of functional dependencies that hold in a set of data:

1. Binarizing or scaling the original set of data, and obtaining a formal context.
2. Using pattern structures.

The purpose of this paper is twofold. On the one hand, we propose using pattern structures as a way to compute the characterization of a set of functional dependencies that hold in a set of data. The interest is to prove that pattern structures is a flexible mechanism that may encode the semantics of functional dependencies without adding further penalty to the resulting formal context. On the other hand, we aim at setting up a solid connection between the formalism of pattern structures and the finding of other different kinds of dependencies that may hold in a given set of data.

The paper is organized as follows. Firstly, the definitions of functional dependencies and their axioms are explained in Section 2 together with the scaling procedure allowing one to derive a formal context from a numerical dataset that characterize FD. Then, Section 3 presents the general formalism of pattern structures. Section 4 gives the core of this article: it shows how to define a pattern structure that holds the same concept lattice than with the introduced scaling. It follows experiments in Section 5 showing the interest of using pattern structures. Finally, the conclusion draws attention to perspectives of research.

2 Motivating Example

This example shows how functional dependencies can be extracted using FCA (this is based on [4]). The main idea behind this method is called *binarization*, and consists in transforming (implicitly) a many-valued set of data into a binary context. This transformation allows us to build a formal context.

Before explaining this process, we first introduce functional dependencies (FD's). Let $\mathcal{U}$ be a set of attributes, and let Dom be a set of values (a domain). For the sake of clarity, we assume that Dom is a numerical set. A tuple t is a function $t : \mathcal{U} \mapsto Dom$, and a table T is a set of tuples. Usually tables are presented as a matrix, as in the following example:

id	A	B	C	D
t_1	1	3	7	2
t_2	1	3	4	5
t_3	3	5	2	2
t_4	3	3	4	8

where the set of tuples (or objects) is $\{ t_1, t_2, t_3, t_4 \}$ and $\mathcal{U} = \{ A, B, C, D \}$ is the set of attributes.

Given a tuple $t \in T$, we say that $t(X)$ (for all $X \subseteq \mathcal{U}$) is the restriction of the tuple t in the attributes $X \subseteq \mathcal{U}$, this is the values of t in the attributes X. For instance, we have that $t_2(\{ A, C \})$ is $\{ 1, 4 \}$. We drop the set notation and say that $t_2(AC)$ is $\{ 1, 4 \}$.

Definition 1 ([18]). *Let T be a set of tuples, and $X, Y \subseteq \mathcal{U}$. A* **functional dependency (FD)** *$X \to Y$, holds in T if:*

$$\forall t_i, t_j \in T : t_i(X) = t_j(X) \Rightarrow t_i(Y) = t_j(Y)$$

For instance, we have that the functional dependency $C \to B$ holds in T, whereas the functional dependency $A \to B$ does not, because $t_3(A) = t_4(A)$ but $t_3(B) \neq t_4(B)$.

We are now ready to explain how to extract the set of functional dependencies that hold in a set of data, using FCA:

1. We define a formal context derived from the original many-valued data T.
2. We extract the implications that hold in the concept lattice associated to that context ([12]).
3. We see that the implications that hold in the concept lattice are the functional dependencies that hold in the original table T.

In order to define a formal context, we need to define first the set of objects:

$$G = \{ (t_i, t_j) \mid i < j \text{ and } t_i, t_j \in T \}$$

It corresponds to the set of all pairs of tuples from T (excluding symmetry and reflexivity). The relation of the context is defined as:

$$(t_i, t_j)\ I\ x \Leftrightarrow t_i(x) = t_j(x)$$

It is important to realize that the formal context $\mathbb{K} = (G, \mathcal{U}, I)$ depends entirely on the table T, since both G and $\mathcal{U}$ do, but this dependency is not explicitly shown in the definition of this context. According to the preceding case, we would have the formal context in Figure 1 and the corresponding concept lattice in Figure 2.

$\mathbb{K}$	A	B	C	D
(1,2)	×	×		
(1,3)				×
(1,4)		×		
(2,3)				
(2,4)		×	×	
(3,4)	×			

Fig. 1. Formal Context

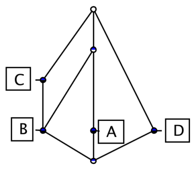


Fig. 2. Concept Lattice

We have created a new formal context out of a multi-valued table. We can see that the size of this context can be of the order of $\mathcal{O}(|T^2|)$ (where $|T|$ is the number of tuples of T), so it can be significantly bigger that the original set of data.

The following step is to compute the Duquenne-Guigues basis ([10]) of the concept lattice, i.e. the minimal set of implications such that all the implications that hold in the formal context can be derived from this set. In this case, this set consists of the following implications:

$$c \to b, abc \to d, bcd \to a$$

This is not the set of all implications that hold in the concept lattice of the formal context that we have defined. This is just a minimal set such that all the implications that hold in the lattice can be derived (by augmentation and transitivity) from this set. This means that, if Σ^* is the set of all implications that hold in the concept lattice, then $\Sigma \models \Sigma^*$.

Finally, we have to realize that the set of implications that hold in the concept lattice are syntactically the same as the set of functional dependencies that hold in T (this is shown in [4,12]). By *syntactically* we mean that whenever an implication $X \to Y$ holds in $\mathbb{K}$, then the functional dependency $X \to Y$ holds in T. Equivalently, the minimal generating set of functional dependencies that hold in T is the same as the Duquenne-Guigues basis of the concept lattice.

A known result of the binarization process precisely states that the set of implications that hold in the context is syntactically equivalent to the set of functional dependencies that hold in the original set of data [12].

3 Pattern Structures in Formal Concept Analysis

We assume the reader to be familiar with basic notions of formal concept analysis. We use the standard notations from [12]. Our interest lies in handling numerical data within FCA. Hence, we recall here the formalism of pattern structures that can be understood as a generalization towards complex data, i.e. objects taking descriptions in a partially ordered set.

A pattern structure is defined as a generalization of a formal context describing complex data [11]. Formally, let G be a set of objects, let $(D, \sqcap)$ be a meet-semi-lattice of potential object descriptions and let $\delta : G \longrightarrow D$ be a mapping associating each object with its description. Then $(G, (D, \sqcap), \delta)$ is a pattern structure. Elements of D are patterns and are ordered by a subsumption relation $\sqsubseteq$: $\forall c, d \in D$, $c \sqsubseteq d \Longleftrightarrow c \sqcap d = c$. A pattern structure $(G, (D, \sqcap), \delta)$ gives rise to two derivation operators $(\cdot)^{\square}$:

$$A^{\square} = \bigsqcap_{g \in A} \delta(g) \qquad for\ A \subseteq G$$

$$d^{\square} = \{g \in G | d \sqsubseteq \delta(g)\} \qquad for\ d \in (D, \sqcap).$$

These operators form a Galois connection between $(2^G, \subseteq)$ and $(D, \sqcap)$. Pattern concepts of $(G, (D, \sqcap), \delta)$ are pairs of the form (A, d), $A \subseteq G$, $d \in (D, \sqcap)$, such that $A^{\square} = d$ and $A = d^{\square}$. For a pattern concept (A, d), d is a pattern intent and is the common description of all objects in A, the pattern extent. When partially ordered by $(A_1, d_1) \leq (A_2, d_2) \Leftrightarrow A_1 \subseteq A_2 \quad (\Leftrightarrow d_2 \sqsubseteq d_1)$, the set of all concepts forms a complete lattice called pattern concept lattice. Existing FCA algorithms [15] can be used with slight modifications to compute pattern structures, in order to extract and classify concepts (details in [11,14]).

4 Finding FD with Partition Pattern Structures

As introduced in Section 2, an existing binarization allows to build a concept lattice from which a set of FDs can be characterized [12]. However, this formal context tends to be very large, even when the initial data are of reasonable size. We show here how the formalism of pattern structures can be instantiated to obtain an equivalent concept lattice. The so called *partition pattern structures* come with several advantages among which computation and interpretation of the resulting lattice.

4.1 Preliminaries on the partition lattice

Partition of a set. Given a set E, a partition over E is a set $P \subseteq \wp(E)$ s.t.:

- $\bigcup_{p_i \in P} p_i = E$
- $p_i \cap p_j = \emptyset$, for any $p_i, p_j \in P$ with $i \neq j$.

In other words, a partition covers E and is composed of disjoint subsets of E.

Equivalence relation. A partition P over a set E is an equivalence relation R_P on E. The 1-1-correspondence between P and R_P is given by $(e, e') \in R_P$ iff e and e' belongs to the same class of P [6,13]. For example, given $P = \{\{1, 2, 3\}, \{4\}\}$, one has the relation $R_P = \{(1, 2), (1, 3), (2, 3), (1, 1), (2, 2), (3, 3), (4, 4)\}$ (omitting symmetry for the sake of readability).

Ordering relation. A partition P_1 is finer than a partition P_2 (P_2 coarser than P_1), written $P_1 \sqsubseteq P_2$ if any subset of P_1 is a subset of a subset in P_2. For example,

$$\{\{1, 3\}, \{2\}, \{4\}\} \sqsubseteq \{\{1, 2, 3\}, \{4\}\}$$

Meet of two partitions. It is defined as the coarsest common refinement. In other words, it is the intersection of the respective equivalence relations:

$$\begin{aligned} \{\{1, 3\}, \{2, 4\}\} \sqcap \{\{1, 2, 3\}, \{4\}\} &= \{\{1, 3\}, \{2\}, \{4\}\} \\ \textit{or } \{(1, 3), (2, 4)\} \cap \{(1, 2), (1, 3), (2, 3)\} \end{aligned}$$

Join of two partitions. It is defined as the finest common coarsening. In other words, it is the transitive closure of the union of the respective equivalence relations.

$$\begin{aligned} \{\{1, 3\}, \{2\}\{, 4\}\} \sqcup \{\{1, 2\}, \{3\}\{4\}\} &= \{\{1, 2, 3\}, \{4\}\} \\ \textit{or transitive\_closure}(\{(1, 3), (4, 4)\} \cup \{(1, 2)(3, 3), (4, 4)\}) \end{aligned}$$

Finally, one should notice that the property $P_1 \sqcap P_2 = P_1 \Leftrightarrow P_1 \sqsubseteq P_2$ naturally holds (and the dual for join). Thus the set of all partitions over a set forms a lattice $(D, \sqcap, \sqcup)$ and can be used as a description space of a pattern structure.

4.2 Partition pattern structure

Consider a numerical table as a many-valued context (G, M, W, I) where G corresponds to the set of objects ("rows"), M to the set of attributes ("columns"), W the data domain ("all distinct values of the table") and $I \subseteq G \times M \times W$ a relation such that $(g, m, w) \in I$ also written $m(g) = w$ means that attribute m takes the value w for the object g [12]. In Table 1 (left), we have $D(4) = 8$.

We show how a partition pattern structure can be defined from a many-valued context (G, M, W, I) and show that its concept lattice is equivalent to the concept lattice obtained after binarization (see Section 2). Intuitively, formal objects of the pattern structure are the attributes of the numerical dataset. Then, given an attribute $m \in M$, its description $\delta(m)$ is given by a partition over G such that any two elements g, h of the same class take the same values for the attribute m, i.e. $m(g) = m(h)$. The result is given in Table 1 (middle). As such, descriptions obey the ordering of a partition lattice as described above. It follows that our initial numerical table (G, M, W, I) can be represented as a pattern structure $(M, (D, \sqcap, \sqcup), \delta)$ where M is the set of original attributes, and $(D, \sqcap, \sqcup)$ is the lattice of partitions over G. An example of concept formation is given as follows, starting from set $\{A, B\} \subseteq M$:

$$\begin{aligned}
\{A, B\}^{\square} &= \delta(A) \sqcap \delta(B) \\
&= \{\{1,2\},\{3,4\}\} \sqcap \{\{1,2,4\},\{3\}\} \\
&= \{\{1,2\},\{3\},\{4\}\} \\
\{\{1,2\},\{3\},\{4\}\}^{\square} &= \{m \in M | \{\{1,2\},\{3\},\{4\} \sqsubseteq \delta(m)\} \\
&= \{A, B\}
\end{aligned}$$

Hence, $(\{A, B\}, \{\{1,2\},\{3\},\{4\}\})$ is pattern concept. The resulting pattern concept lattice is given in Table 1 (left).

id	A	B	C	D
1	1	3	7	2
2	1	3	4	5
3	3	5	2	2
4	3	3	4	8

$m \in M$	$\delta(m) \in (D, \sqcap, \sqcup)$
A	$\{\{1,2\},\{3,4\}\}$
B	$\{\{1,2,4\},\{3\}\}$
C	$\{\{1\},\{2,4\},\{3\}\}$
D	$\{\{1,3\},\{2\},\{4\}\}$

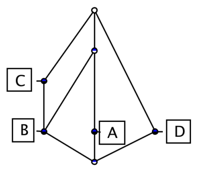


Table 1. The original data (left), the resulting pattern structure (middle) and its pattern concept lattice (right)

4.3 Formal context of the partition pattern structure

We showed that a numerical dataset can be described by a many-valued context (G, M, W, I) from which one can derive the formal context $(M, \mathcal{B}_2(G), I)$ [4] where $\mathcal{B}_2(G)$ represents any pair of objects, and $(m, (g, h)) \in I$ means that $m(g) = m(h)$. The resulting concept lattice can be used to extract FD [12]. A result is that both introduced structures $(M, \mathcal{B}_2(G), I)$ and $(M, (D, \sqcap), \delta)$ are equivalent, i.e. both collections of concepts are in 1-1-correspondence.

Proposition. Let (G, W, M, I) be a many-valued context. Let $(M, \mathcal{B}_2(G), I)$ be the formal context such as $(m, (g, h)) \in I \Leftrightarrow m(g) = m(h)$. Let $(M, (D, \sqcap, \sqcup), \delta)$ be the partition pattern structure where, for $m \in M$, $\delta(m)$ is the partition of G such that $p, q \in P$, $P \in \delta(m) \Leftrightarrow m(p) = m(q)$. Then, the following holds.
1. For any formal concept (R, S), there is one and only one pattern concept (C, d) such that $R = C$ and S is the equivalence class representation of d.
2. and vice-versa.

[4] Originally $(\mathcal{B}_2(G), M, I)$, but for sake of simplicity here, we permute formal objects and attributes. The results hold equally.

Proof. Consider both structures $(M, \mathcal{B}_2(G), I)$ and $(M, (D, \sqcap, \sqcup), \delta)$. They both hold the same set of "formal objects" M (attributes in the many-valued context). In the pattern structure, elements of M are described by a partition over G. In the formal context, elements are described by pairs of objects, that is, by definition, the representation of the same partitions. As such, elements of M are described in an equivalent way. Furthermore, intersections in both representations are equivalent too. Indeed, the meet operation between two partitions is known to be the intersection of their equivalence class representation. Since it is known that derivation operations are defined in pattern structures in the same way than in formal contexts, the proposition naturally holds.

Example. The pattern concept $(\{B\}, \{\{1,2,4\},\{3\}\})$ is equivalent to the formal concept $(\{B\}, \{(1,2),(2,4),(1,4)\})$.

From this example, one should remark that pattern structures offer more concise intent representation when the set of object becomes very large, i.e. storing a partition instead of all pairs of objects that are together in a same class of the partition. This leads us to the next section, where the attention is drawn to a computational comparison of both approaches.

5 Experiments

We showed how pattern structures can alternatively represent the formal context $(M, \mathcal{B}_2(G), I)$ by means of partition patterns. Both concept lattices are equivalent and thus can be used to characterize FD. To assess the usefulness of introducing partition pattern structures, we applied both methods to well known UCI datasets[5]. To compute with formal contexts, we wrote a simple procedure to scale the many-valued context into a formal context, and applied the (C++) closed itemset mining algorithm LCM (version 2 [17]). Whereas this algorithm only computes concept intents, it is known to be one of the most efficient for that task. To compute with pattern structures, we turned the many-valued context into a set of partitions over G (one for each attribute $m \in M$) and applied a slight (Java) modification of the algorithm CloseByOne [15]. Indeed, the latter can be easily adapted by changing the definition of both intersection and subsumption test, used for closures computation (a detailed explanation for another instance of pattern structures can be found in [14]). As such, this method computes pattern concepts, i.e. both pattern extent and intent.

Table 2 gives the details of the datasets and their derived formal context we experiment with. Note that in column $|\mathcal{B}_2(G)|$, formal objects (g, h) with empty description, i.e. $\{(g,h)\}' = \emptyset$ for any $g, h \in G$, are not taken into account. Table 3 gives the execution time of both methods. For pattern structures, execution times include the reading of the data, their process to a set of partitions and the CloseByOne execution. Concerning formal contexts, execution times include data reading and process with LCM, while the time to build the formal context is not taken into account. In both case, algorithms only output the number of

[5] `http://archive.ics.uci.edu/ml/datasets.html`

patterns. The experiments were carried out on an Intel Core i7 CPU 2.40 Ghz machine with 5 GB RAM.

	(G,M,W,I)		$(\mathcal{B}_2(G),MI)$		
Dataset	$\|G\|$	$\|M\|$	$\|\mathcal{B}_2(G)\|$	Avg. $\|g'\|$	Density
iris	150	5	4,363	1.38	27.56%
hepatitis	155	20	11,935	9.02	45.08%
glass	214	10	19,601	1.74	17.43%
imports-85	205	26	20,904	6.24	23.99%
balance-scale	625	5	143,236	1.67	33.35%
crx	690	16	236,633	5.53	43.54%
flare	1,066	13	567,645	8.79	67.60%
abalone	4,177	9	3,752,318	1.19	13.18%
krkopt-25%	7,013	7	20,115,505	1.84	26.26%
krkopt-50%	14,027	7	76,547,447	1.72	24.59%
krkopt-75%	21,040	7	171,199,419	1.66	24.22%
krkopt-100%	28,056	7	299,171,478	1.67	23.88%
adult-25%	8,140	15	33,124,730	6.32	42.14%
adult-50%	16,280	15	132,507,392	6.34	42.28%
adult-75%	24,320	15	295,709,848	6.34	42.20%
adult-100%	32,561	15	530,077,524	6.33	42.21%

Table 2. Datasets and their characteristics

From Table 3, it can be observed than computing with formal contexts is faster for the smallest datasets, even *abalone* that holds more than 3 millions of formal objects. However, with bigger datasets, from 20 to 530 millions of objects, partition pattern structures are the only able to compute the set of concepts. This holds for 7 numerical attributes already, and is bolder with 15. It is indeed already known that complexity of computing FD is highly related with the number of numerical attributes M.

As already suggested in [14,11] in different settings, the explanation is that when working with simple descriptions (i.e. vectors of bits), computing an intersection is more efficient than when working with more complex descriptions. Indeed, partitions are encoded in our algorithm as vectors of bitvectors (i.e. partitions) and both intersections or inclusion tests computation require to consider all pairs of sets between the two partitions in argument. Although we used optimizations avoiding an exhaustive computation between all pairs (by considering a lectic order on parts), those operations are more complex than standard intersections and inclusion tests between sets. However, we need to compute much less intersections, thus the following trade-off. Pattern structures perform better with larger datasets. Formal objects (numerical attributes) map to concise descriptions (partitions) whereas they map with the equivalence class of the same partitions in the case of formal contexts. Consequently, pattern structures are preferred to formal contexts when the number of possible pairs of objects that

	CloseByOne		LCMv2	
Dataset	Pattern concepts	Time (ms)	Concept intents	Time (ms)
iris	26	13	26	**4**
balance-scale	30	**30**	30	76
flare	4,096	**258**	4,096	650
glass	133	373	133	**41**
crx	9,528	4,367	9,528	**112**
abalone	252	5,887	252	**692**
hepatitis	95,576	11,178	95,576	**122**
imports85	205,623	228,877	205,623	**112**
krkopt-25%	126	**195**	126	5,441
krkopt-50%	126	**352**	N/A	N/A
krkopt-75%	126	**631**	N/A	N/A
krkopt-100%	126	**896**	N/A	N/A
adult-25%	10,881	**43,949**	N/A	N/A
adult-50%	12,398	**152,242**	N/A	N/A
adult-75%	13,133	**316,250**	N/A	N/A
adult-100%	13,356	**520,431**	N/A	N/A

Table 3. Comparing pattern structures and formal context representations. N/A means that the computation was intractable for memory issues.

agree for one or more attributes is high ($|\mathcal{B}_2(G)|$). Finally, let us recall that execution times for formal contexts do not include the scaling procedure time, since such procedure is highly dependent of I/O performances. We simply remark that conceptual scaling lengths more than 5 minutes for the dataset *adult-100%* resulting in a text-file of more than 10 giga-bytes (in the standard format of itemset mining algorithms: each line corresponds to an object described by the indexes of its attributes separated by a space).

To conclude, even with a simple Java implementation of CloseByOne (computing both extents and intents of pattern concepts), we gave here a proof of concept that pattern structures reveal themselves as a good trade off to overcome scaling difficulties, i.e. for computing the set of concepts whose lattice is equivalent to the one obtained after conceptual scaling.

6 Conclusions

We have presented a method to compute the characterization of a set of functional dependencies that hold in a set of many-valued data, based on formal concept analysis plus pattern structures. From this characterization, it is simply a matter of applying well-known algorithms to compute the minimal set of dependencies that imply the whose set (otherwise known as the Duquenne-Guigues basis). There was already methods to compute the characterization of those dependencies using FCA: One possibility is using conceptual scaling, which is the classical method to deal with many-valued data in FCA. This paper proposes to use pattern structures, because they have already been used successfully to deal with many-valued data ([14]).

The empirical results compare an algorithm based on scaling versus another based on pattern structures, and show that scaling is faster for small datasets, whereas pattern structures perform better for large datasets, precisely where the scaling-based algorithm is not able to compute the output. This indicates that this new paradigm is more scalable in terms of time and memory. Since datasets tend to become larges and contain more attributes, this scalability may be a much important feature than speed in small datasets.

The results in this paper present a new paradigm for computing a characterization of functional dependencies that outperforms algorithms based on the *classical* conceptual scaling, which shows the interest of pattern structures for dealing with many-valued data within FCA. We think that the results that have been presented open the possibility to adapt this pattern structures based framework to other kinds of dependencies, namely, multi-valued dependencies and similar constraints that may be found in different fields.

7 Acknowledgements

This research work has been supported by the Spanish Ministry of Education and Science (project TIN2008-06582-C03-01), EU PASCAL2 Network of Excellence, and by the Generalitat de Catalunya (2009-SGR-980 and 2009-SGR-1428) and AGAUR (grant 2010PIV00057) that allowed professor Napoli to visit the Universitat Politècnica de Catalunya.

References

1. S. Abiteboul, R. Hull, and V. Vianu. Foundations of Databases, 1995. Addison-Wesley.
2. J. Baixeries and J. L. Balcázar. Discrete Deterministic Data Mining as Knowledge Compilation. Proceedings of Workshop on Discrete Mathematics and Data Mining in SIAM International Conference on Data Mining, 2003.
3. J. Baixeries. A Formal Concept Analysis framework to model functional dependencies. Mathematical Methods for Learning, 2004.
4. J. Baixeries. Lattice Characterization of Armstrong and Symmetric Dependencies. Ph. Thesis, 2007.
5. C. Beeri, R. Fagin,and J. H. Howard. A Complete Axiomatization for Functional and Multivalued Dependencies in Database Relations. Proceedings of the 1977 ACM SIGMOD International Conference on Management of Data, Toronto, Canada, August 3-5, 1977.
6. N. Caspard and B. Monjardet. The Lattices of Closure Systems, Closure Operators, and Implicational Systems on a Finite Set: a Survey. Proceedings of the 1998 Conference on Ordinal and Symbolic Data Analysis (OSDA-98). Discrete Applied Mathematics, 2003.
7. A. Day. The Lattice Theory of Functional Dependencies and Normal Decompositions. International Journal of Algebra and Computation Vol. 2, No. 4 409-431. 1992.

8. J. Demetrovics, G. Hencsey, L. Libkin and I. Muchnik. Normal Form Relation Schemes: a New Characterization. Acta Cybernetica, 1992.
9. J. Demetrovics, L. Libkin and I. Muchnik. Functional Dependencies in Relational Databases: a Lattice Point of View. Discrete Applied Mathematics, 1992.
10. V. Duquenne and J.L. Guigues. Familles Minimales d'Implications Informatives Resultant d'un Tableau de Donées Binaires. Mathematics and Social Sciences, 1986.
11. B. Ganter and S. O. Kuznetsov. Pattern structures and their projections. In: Delugach, H., Stumme, G. (eds.) Conceptual Structures: Broadening the Base, Proceedings of the 9th International Conference on Conceptual Structures (ICCS 2001). pp. 129–142. LNCS 2120, Springer (2001)
12. B. Ganter and R. Wille: Formal Concept Analysis. Springer, Berlin (1999)
13. G. Grätzer. General Lattice Theory. Academic Press, 1978.
14. M. Kaytoue, Kuznetsov and A. Napoli. Revisiting Numerical Pattern Mining with Formal Concept Analysis. IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia.
15. S. O. Kuznetsov and S. Obiedkov: Comparing performance of algorithms for generating concept lattices. Journal of Experimental & Theoretical Artificial Intelligence 14(2/3), 189–216 (2002)
16. D. Simovici and R. Tenney. Relational Database Systems. Academic Press, 1995.
17. T. Uno, M. Kiyomi, and H. Arimura:Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. IEEE ICDM Workshop on Frequent Itemset Mining Implementations, 2004.
18. J. D. Ullman. Principles of Database Systems. Computer Science Press, 1982.

Computing minimal generators from implications: a logic-guided approach

P. Cordero, M. Enciso, A. Mora, M Ojeda-Aciego*

Universidad de Málaga. Spain.
pcordero@uma.es, enciso@lcc.uma.es, amora@ctima.uma.es, aciego@uma.es

Abstract. Sets of attribute implications may have a certain degree of redundancy and the notion of basis appears as a way to characterize the implication set with less redundancy. The most widely accepted is the Duquenne-Guigues basis, strongly based on the notion of pseudo-intents. In this work we propose the minimal generators as an element to remove redundancy in the basis.
The main problem is to enumerate all the minimal generators from a set of implications. We introduce a method to compute all the minimal generators which is based on the Simplification Rule for implications. The simplification paradigm allows us to remove redundancy in the implications by deleting attributes inside the implication without removing the whole implication itself. In this work, the application of the Simplification Rule to the set of implications guides the search of the minimal generators in a logic-based style, providing a deterministic approach.

1 Introduction

A *concept* is a general idea that corresponds to some kind of entities and that may be characterized by some essential features of the class. When Wille [18] conceived Formal Concept Analysis (FCA), he probably did not foresee the wide diffusion of his original idea, both in the theoretical and in the applied areas. Application areas of FCA ranges from data analysis, information retrieval, or data mining to knowledge representation or the semantic web.

The main goal of Formal Concept Analysis is to identify the relationships between sets of objects and sets of attributes from information in a binary table. These relationships establish a Galois connection which allows us to identify the concepts by using lattice theory. The construction of the lattice of concepts is known to be a hard problem due to its exponential complexity. For this task, probably the most cited method is the NEXTCLOSURE algorithm developed by Ganter [5].

Apart from building the concept lattice itself, one of the key problems is to extract the set of attribute implications which hold in the concept lattice. In

* Supported by grants P09-FQM-5233 of the Junta de Andalucía, and TIN09-14562-C05-01 and TIN2011-28084 of the Science and Innovation Ministry of Spain, co-funded by the European Regional Development Fund (ERDF).

real applications the size of the concept lattice is usually huge and is impossible for the user to visualize it. Belohlavek and Vychodil developed [1] a method of expressing implications by means of closure operators. The main effect is to filter-out outputs to the user. The study of implications allowed those authors to reduce the number of formal concepts extracted from the input data: *"Using background knowledge thus enables a focused extraction of knowledge and may considerably reduce the amount of information presented to the user"* [2].

The problem arises when the set of implications has a high degree of redundancy: by the existence of redundant implications or redundant attributes inside the implications. One desired goal in this area is to remove redundancy and obtain a minimal basis. The most widely approach comes from the notion of *Duquenne-Guigues Basis* [6] also called *stem base*. This basis is minimal w.r.t. the number of implications, i.e. if one of the implications is removed from the basis, there are non-redundant implications which are valid in the dataset and cannot be inferred, using the Armstrong's Axioms, from the new reduced basis.

Nevertheless, this notion of minimality and its associated redundancy may be improved. To illustrate this assertion, we present here an example appeared in [5, pp. 30 and 84] where Ganter presents a context of developing countries. In the example, 130 countries and six attributes are considered: Group of 77, Non-aligned, LLDC (Least Developed Countries), MASC (Most Seriously Affected Countries), OPEC (Organization of Petrol Exporting Countries) and ACP (African, Caribbean and Pacific Countries). Ganter builds the concept lattice from this context and provides the following Duquenne-Guigues basis:

$$\begin{array}{rcl}
\text{OPEC} & \rightarrow & \text{Group 77, Non-aligned} \\
\text{MASC} & \rightarrow & \text{Group 77} \\
\text{Non-aligned} & \rightarrow & \text{Group of 77} \\
\text{Group 77, Non-aligned, MASC, OPEC} & \rightarrow & \text{LLDC, ACP} \\
\text{Group 77, Non-aligned, LLDC, OPEC} & \rightarrow & \text{MASC, ACP}
\end{array}$$

Note, however, that in the last two implications, there still exist redundant attributes in the left hand side, whereas in the first and in the last implications the redundancy appears in the right hand sides. This implies that it is possible to provide an equivalent and simpler set of implications:

$$\begin{array}{rcl}
\text{OPEC} & \rightarrow & \text{Non-aligned} \\
\text{MASC} & \rightarrow & \text{Group 77} \\
\text{Non-aligned} & \rightarrow & \text{Group of 77} \\
\text{MASC, OPEC} & \rightarrow & \text{LLDC, ACP} \\
\text{LLDC, OPEC} & \rightarrow & \text{MASC}
\end{array}$$

Thus, we have an example in which the Duquenne-Guigues basis still can contain redundant information. In order to obtain the latter basis it is necessary to consider minimal generators instead of pseudo-intents.

Implications define a closure system, and for this reason, a closure system can be replaced by their equivalent implications. The relationship between closed sets and implications are the key of the attribute exploration techniques [8, 13].

In [12] we introduced a closure algorithm strongly based on the simplification logic [3]. Here, our closure operator is used to guide the search of all the minimal generators of the closed sets corresponding to a given set of implications.

Methods for obtaining generators of closed sets have been studied in [4,10,17]. Minimal generators [14–16] appear in the literature under different names in various fields. For instance, in relational databases they are called minimal keys. In [17], the authors emphasize the importance of studying minimal generators although "they have been paid little attention so far in the FCA literature".

We would like to provide, in a further step, an alternative definition of basis built around the notion of minimal generator, since our goal is to avoid redundancy *inside* the implications, instead of just minimizing the number of implications. As a preliminary step, we have to design a method to enumerate all the minimal generators and their corresponding closed sets.

This paper is structured in the following way: in Section 2 some preliminaries about formal concept analysis and the Simplification paradigm are presented. In Section 3 a method to enumerate all the minimal generators from an implication set is introduced and in subsection 3.2 the algorithm is modified to compute the non-trivial minimal generators. The paper ends with a section on conclusions and prospects for future work.

2 Preliminaries

2.1 Formal Concept Analysis

Intuitively, Formal Concept Analysis (FCA) provides methods to describe the relationship between a set of objects and a set of attributes. A **formal context** is a triple $\mathbf{K} := (G, M, I)$ where G is a set of objects, M is a set of attributes and $I \subseteq G \times M$ is a binary relation between G and M such that, for $o \in G$ and $a \in M$, $o\ I\ a$ means that the object o has the attribute a. From this triple two mappings can be defined. One of them $(\ )' \colon 2^G \to 2^M$ is defined for all $A \subseteq G$ as $A' = \{m \in M \mid g\ I\ m \text{ for all } g \in A\}$. The other one, $(\ )' \colon 2^M \to 2^G$ is defined for all $B \subseteq M$ as $B' = \{g \in G \mid g\ I\ m \text{ for all } m \in B\}$. Both mappings are denoted by the same symbol because no confusion arises. This pair of mappings is a Galois connection and both are antitone mappings ($A_1 \subseteq A_2$ implies $A'_2 \subseteq A'_1$ for all $A_1, A_2 \subseteq G$, and, for all $B_1, B_2 \subseteq M$, if $B_1 \subseteq B_2$ then $B'_2 \subseteq B'_1$)

The composition of the intent and the extent mappings, and vice versa, give us two closure operators $(\ )'' \colon 2^G \to 2^G$ and $(\ )'' \colon 2^M \to 2^M$. That is, both are extensive ($X \subseteq X''$), idempotent ($(X'')'' = X''$) and isotone (if $X_1 \subseteq X_2$ then $X''_1 \subseteq X''_2$). In order to make this work self-contained, the notion of closed set (as a fixpoint of a closure operator) is defined below:

Definition 1. *A* ***formal concept*** *is a pair (A, B) such that $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. Consequently, A and B are closed sets of objects and attributes, respectively called extent and intent.*

It is well-known that the set of formal concepts is a complete lattice, the **concept lattice associated to the context**, with the following partial ordering is considered:

$$(A_1, B_1) \leq (A_2, B_2) \text{ if and only if } A_1 \subseteq A_2 \text{ (or equivalently } B_1 \supseteq B_2\text{)}$$

Related to the notion of closed set, the minimal generator (mingen) and implication are defined as follows:

Definition 2. *Let* $\mathbf{K} = (G, M, I)$ *be a formal context and* $A \subseteq M$*. The set of attributes* A *is said to be a minimal generator (**mingen**) if, for all set of attributes* $X \subseteq A$ *if* $X'' = A''$ *then* $X = A$*.*

Attribute implication allows us to capture all the information which can be deduced from a context. They are expressions $A \to B$ being A and B sets of attributes. A context satisfies the implication $A \to B$ if every object that has all the attributes from A also has all the attributes from B.

Definition 3. *An (attribute) implication of a formal context* $\mathbf{K} = (G, M, I)$ *is defined as a pair* (A, B)*, written* $A \to B$*, where* $A, B \subseteq M$*. Implication* $A \to B$ *holds (is valid) in* $\mathbf{K}$ *if* $A' \subseteq B'$*.*

The set of all valid implications in a context satisfies the well-known Armstrong's axioms:

$$\texttt{[Ref]}\ \frac{A \supseteq B}{A \to B}, \quad \texttt{[Augm]}\ \frac{A \to B}{A \cup C \to B \cup C}, \quad \texttt{[Trans]}\ \frac{A \to B,\ B \to C}{A \to C}$$

An implication **basis** of $\mathbf{K}$ is defined as a set $\mathcal{L}$ of implications of $\mathbf{K}$ from which any valid implication for $\mathbf{K}$ can be deduced by using Armstrong rules. The goal is to obtain a implication basis with minimal size. This condition is satisfied by the so-called Duquenne-Guigues basis [6] or stem basis, which is built over the set of the pseudo-intents [5].

As we have illustrated in the introduction, the definition of the Duquenne-Guigues basis refers to minimality only in the size of the set of implicants, but redundant attributes use to appear in this kind of minimal basis. This situation comes from the use of pseudo-intents in the construction of the basis. To avoid redundant attributes, we propose to consider minimal generators in the left hand side of the implications. In this work we do not provide such alternative definition of minimal basis, but focus on the problem of enumerating all the minimal generators. This is the main goal of Section 3, but before we need to recall the basics of the simplification logic.

2.2 Simplification logic and closures

Armstrong's axiom system has influenced the design of several logics for functional dependencies [9,11], all of them built around the transitivity paradigm. In this section, we summarize the axiom system of Simplification Logic for Functional Dependencies $\mathbf{SL}_{FD}$. It avoids the use of transitivity and is guided by the idea of simplifying the set of functional dependencies by efficiently removing some redundant attributes [3].

To begin with, $\mathbf{SL}_{FD}$ logic considers reflexivity as axiom scheme

$$\texttt{[Ref]}\quad \frac{A \supseteq B}{A \to B}$$

together with the following inference rules called fragmentation, composition and simplification respectively.

$$\texttt{[Frag]}\ \frac{A{\to}B\cup C}{A{\to}B} \qquad \texttt{[Comp]}\ \frac{A{\to}B,\ C{\to}D}{A\cup C{\to}B\cup D} \qquad \texttt{[Simp]}\ \frac{A{\to}B,\ C{\to}D}{A\cup(C\smallsetminus B){\to}D}$$

The equivalence between $\mathbf{SL}_{FD}$ logic and Armstrong's Axioms and an efficient algorithm to compute the closure of a set of attributes were proposed in [12]. We have also conducted a comparison to other closure algorithms in the literature in order to prove the better performance of our logic-based system. The main advantage of $\mathbf{SL}_{FD}$ is that inference rules can be considered as equivalence rules, and that is enough to compute all the derivations (see [12] for further details).

Proposition 1. *Let A, B, C, D sets of attributes. In $\mathbf{SL}_{FD}$ logic, the following equivalences hold:*

1. $\{A{\to}B\} \equiv \{A{\to}B \smallsetminus A\}$
2. $\{A{\to}B, A{\to}C\} \equiv \{A{\to}B\cup C\}$
3. $A\cap B = \varnothing$ *and* $A\subseteq C$ *imply* $\{A{\to}B, C{\to}D\} \equiv \{A{\to}B, C\smallsetminus B{\to}D\smallsetminus B\}$

It is well-known that, given a basis of a context, the closure of attribute sets defined based on Armstrong's axioms coincides with the closure $()'' : 2^M \to 2^M$. On the other hand, Armstrong's axioms and the axiom system in $\mathbf{SL}_{FD}$ logic are equivalent.

Definition 4. *Let Γ be a set of implications and A be a set of attributes. The closure of A in $\mathbf{SL}_{FD}$ logic is defined as the maximum set of attributes A^+ such that $\Gamma \vdash A{\to}A^+$.*

Theorem 1. *Let $\mathbf{K} = (G, M, I)$ a formal context and Γ a basis for $\mathbf{K}$. For all $A\subseteq M$, the equality $A^+ = A''$ holds.*

As we have said, in [12] we present a novel algorithm to compute closures using $\mathbf{SL}_{FD}$Logic. The formula $\varnothing{\to}A$ is used as a seed by the reasoning method to render the closure A^+ of A (which coincides with A'') just by applying some operators based on the [`Simp`] inference rule. The algorithm is based on the following results:

Theorem 2. *Let A and B be sets of attributes and Γ be a set of implications.*

$$\Gamma\vdash A{\to}B \quad \textit{if and only if} \quad \{\varnothing{\to}A\}\cup\Gamma\vdash\{\varnothing{\to}B\}$$

The closure algorithm is based on the previous theorem and the systematic application of the following equivalences which are direct consequences of the simplification equivalence (item 3 in Proposition 1).

Proposition 2. *Let A, B and C be sets of attributes, then the following equivalences hold:*

- **Eq. I**: *If $B\subseteq A$ then $\{\emptyset{\to}A, B{\to}C\}\equiv\{\emptyset{\to}A\cup C\}$.*
- **Eq. II**: *If $C\subseteq A$ then $\{\emptyset{\to}A, B{\to}C\}\equiv\{\emptyset{\to}A\}$.*
- **Eq. III**: *Otherwise $\{\emptyset{\to}A, B{\to}C\}\equiv\{\emptyset{\to}A, B\smallsetminus A{\to}C\smallsetminus A\}$.*

Given a set of implications Γ and a set of attributes A, the execution of the $\mathbf{SL}_{FD}$closure method begins with the construction of the guide $\emptyset \to A$. The three equivalences are systematically applied to Γ, which produces a growth of the guide. When none of the three equivalences can be applied, in the guide we have the closure.

Method 1 *Let Γ be a set of implications and A a subset of attributes. The following method computes the closure A^+ of the set A w.r.t. Γ:*

1. *Build the guide as follows $\{\emptyset \to A\}$*
2. *While there exist $B \to C \in \Gamma$ such that $A \cap B \neq \varnothing$ or $A \cap C \neq \varnothing$ (being $\{\varnothing \to A\}$ the guide in this state of the execution), execute the corresponding equivalence:*
   - *If $B \subseteq A$ then remove $B \to C$ from Γ and change the guide $\{\emptyset \to A\}$ by $\{\emptyset \to A \cup C\}$.*
   - *If $C \subseteq A$ then remove $B \to C$ from Γ.*
   - *Otherwise substitute $B \to C$ by $B \smallsetminus A \to C \smallsetminus A$ in Γ.*
3. *If $\{\varnothing \to A\}$ is the guide (in this state) then return A.*

Example 1. Let $\Gamma = \{ab \to c, bc \to d, de \to f, ce \to f\}$ and $A = de$. The following table summarizes the execution of Method 1 to compute $A^+ = def$.

Guide	Γ			
$\emptyset \to de$	$ab \to c$	$bc \to d$	$de \to f$	$ce \to f$
$\emptyset \to de$	$ab \to c$	$bc \to \cancel{d}$	$\cancel{d}\cancel{e} \to f$	$c\cancel{e} \to f$
$\emptyset \to def$	$ab \to c$			$c \to \cancel{f}$
$\emptyset \to def$	$ab \to c$			

The new closure operator was proven to be faster than other closure algorithms [12]. Finally, it is possible to define a modification of the closure algorithm to return not only the closure but also the resulting set Γ. This algorithm will be denoted by `Cls`. So, considering the previous example,

$$\texttt{Cls}(de, \{ab \to c, bc \to d, de \to f, ce \to f\}) = (def, \{ab \to c\})$$

3 Computing all mingens from a basis

In this section, we present how the Simplification paradigm can be considered as a powerful tool to guide the search of all minimal generator sets. In the first method we present here, we will compute mingens (the set of all minimal generators) from a set of implicant set. The algorithm works by applying the $\mathbf{SL}_{FD}$Closure algorithm to the set of implications. This application provides a new candidate to be added to mingen and a smaller implications set which guides us in the search of new sets of attributes to be added to mingens.

The input of this algorithm is a set of attributes M and a set of implications Γ over the attributes in M. The output is the set of closed sets endowed with all the mingens that generate them, i.e.

$$\{\langle C, mg(C)\rangle \colon C \text{ is a closed set of attributes}\}$$

where $mg(C) = \{D \colon D \text{ is a mingen and } D^+ = C\}$.

For example, if $M = \{a, b, c\}$ and $\Gamma = \{a \to b, b \to a\}$ the output must be

$$\{\langle abc, \{ac, bc\}\rangle, \langle ab, \{a, b\}\rangle, \langle c, \{c\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}$$

It can be seen as the concept lattice in which we have added labels with the mingens to each closed set.

With this idea we are going to need operators that allow us to work with this kind of sets, i.e. sets of pairs $\langle A, B\rangle$ such that $A \subseteq M$ is an intent and $B \subseteq 2^M$ satisfies the following conditions:

1. $X \subseteq A$ for all $X \in B$.
2. $X, Y \in B$ and $X \subseteq Y$ imply $X = Y$.

This kind of sets are called labeled set of intents (LSI). Given two LSIs Φ and Ψ, we define the join of both, $\Phi \sqcup \Psi$, as the the minimum LSI that satisfies:

– If $\langle A_1, B_1\rangle \in \Phi$ and $A_1 \neq A_2$ for all $\langle A_2, B_2\rangle \in \Psi$ then $\langle A_1, B_1\rangle \in \Phi \sqcup \Psi$
– If $\langle A_1, B_1\rangle \in \Psi$ and $A_1 \neq A_2$ for all $\langle A_2, B_2\rangle \in \Phi$ then $\langle A_1, B_1\rangle \in \Phi \sqcup \Psi$
– If $\langle A, B_1\rangle \in \Psi$ and $\langle A, B_2\rangle \in \Phi$ then $\langle A, B_3\rangle \in \Phi \sqcup \Psi$ being B_3 the set of minimal elements of $B_1 \cup B_2$.

We define an operator, the trivial operator, that given M and Γ returns the following LSI:

$$trv(M, \Gamma) = \{\langle X, \{X\}\rangle \colon X \subseteq M \text{ with } A \not\subseteq X \text{ for all } A \to B \in \Gamma\}$$

For example, $trv(\{a, b, c\}, \{a \to b, b \to a\}) = \{\langle c, \{c\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}$.

We will also need a way to add a pair to an LSI; it is defined as follows:

$$\texttt{Add}(\langle C, \{D\}\rangle, \Phi) = \{\langle A \cup C, \{X \cup D \colon X \in B\}\rangle \colon \langle A, B\rangle \in \Phi\}$$

For example,

$$\texttt{Add}(\langle ab, \{a\}\rangle, \{\langle c, \{c\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}) = \{\langle abc, \{ac\}\rangle, \langle ab, \{a\}\rangle\}$$
$$\texttt{Add}(\langle c, \{c\}\rangle, \{\langle ab, \{a, b\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}) = \{\langle abc, \{ac, bc\}\rangle, \langle c, \{c\}\rangle\}$$

3.1 MinGen Algorithm

We already have all the tools needed to define the algorithm, which is introduced below, together with an illustrative example of its application.

Example 2. We want to compute

$$\texttt{MinGen}(\{a, b, c, d\}, \{a \to b, c \to bd, bd \to ac\})$$

Step 1.
$$\Phi = trv(\{a, b, c, d\}, \{a \to b, c \to bd, bd \to ac\} = \\ = \{\langle b, \{b\}\rangle, \langle d, \{d\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}$$

Algorithm 1: MinGen

```
Data: $M, \Gamma$
Result: $\Phi$
  begin
    if $\Gamma = \varnothing$ then
        $\Phi = trv(M, \Gamma)$
    else
        Let $\Phi = trv(M, \Gamma)$;
        foreach $A \to B \in \Gamma$ do
            Let $(A^+, \Gamma') = \texttt{Cls}(A, \Gamma)$;
            Let $\Phi := \Phi \sqcup \texttt{Add}(\langle A^+, \{A\}\rangle, \texttt{MinGen}(M \smallsetminus A^+, \Gamma')$;
    Return $\Phi$
```

Step 2. Considering $a \to b \in \Gamma$,

$$\texttt{Cls}(a, \Gamma) = (ab, \{c \to d, d \to c\})$$

$\emptyset \to a$	$a \to b$	$c \to bd$	$bd \to ac$
$\emptyset \to a$	$\cancel{a} \to b$	$c \to bd$	$bd \to \cancel{a}c$
$\emptyset \to ab$		$c \to \cancel{b}d$	$\cancel{b}d \to c$

and call to $\texttt{MinGen}(\{c, d\}, \{c \to d, d \to c\})$

Step 2.1. This label (2.1) points that we have passed to a lower level, i.e. we have made a recursive call to the procedure.

$$\Phi_1 = trv(\{c, d\}, \{c \to d, d \to c\} = \{\langle \varnothing, \{\varnothing\}\rangle\}$$

Step 2.2. Considering $c \to d \in \Gamma_1$,

$$\texttt{Cls}(c, \Gamma_1) = (cd, \varnothing)$$

$\emptyset \to c$	$c \to d$	$d \to c$
$\emptyset \to c$	$\cancel{c} \to d$	$d \to \cancel{c}$
$\emptyset \to cd$		

and $\texttt{MinGen}(\varnothing, \varnothing) = \{\langle \varnothing, \{\varnothing\}\rangle\}$.
And so

$$\begin{aligned}\Phi_1 &= \{\langle \varnothing, \{\varnothing\}\rangle\} \sqcup \texttt{Add}(\langle cd, \{c\}\rangle, \{\langle \varnothing, \{\varnothing\}\rangle\}) \\ &= \{\langle cd, \{c\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}\end{aligned}$$

Step 2.3. Considering $d \to c \in \Gamma_1$,

$$\texttt{Cls}(d, \Gamma_1) = (cd, \varnothing)$$

$\emptyset \to d$	$c \to d$	$d \to c$
$\emptyset \to d$	$c \to \cancel{d}$	$\cancel{d} \to c$
$\emptyset \to cd$		

and $\texttt{MinGen}(\varnothing, \varnothing) = \{\langle \varnothing, \{\varnothing\}\rangle\}$.
And so

$$\begin{aligned}\Phi_1 &= \{\langle cd, \{c\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\} \sqcup \texttt{Add}(\langle cd, \{d\}\rangle, \{\langle \varnothing, \{\varnothing\}\rangle\}) \\ &= \{\langle cd, \{c, d\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}\end{aligned}$$

Returning to the higher level,

$$\begin{aligned}\Phi &= \{\langle b, \{b\}\rangle, \langle d, \{d\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\} \sqcup \ \texttt{Add}(\langle ab, \{a\}\rangle, \{\langle cd, \{c, d\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}) \\ &= \{\langle abcd, \{ac, ad\}\rangle, \langle ab, \{a\}\rangle, \langle b, \{b\}\rangle, \langle d, \{d\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}\end{aligned}$$

Step 3. Considering $c \to bd \in \Gamma$,

$$\texttt{Cls}(bc, \Gamma) = (abcd, \varnothing)$$

$\emptyset \to c$	$a \to b$	$c \to bd$	$bd \to ac$
$\emptyset \to c$	$a \to b$	$\cancel{c} \to bd$	$bd \to a\cancel{c}$
$\emptyset \to bcd$	$a \to \cancel{b}$		$\cancel{b}\cancel{d} \to a$
$\emptyset \to abcd$			

and $\texttt{MinGen}(\varnothing, \varnothing) = \{\langle \varnothing, \{\varnothing\}\rangle\}$.

$$\begin{aligned}\Phi &= \{\langle abcd, \{ac, ad\}\rangle, \langle ab, \{a\}\rangle\langle b, \{b\}\rangle, \langle d, \{d\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\} \\ &\sqcup \texttt{Add}(\langle abcd, \{c\}\rangle, \{\langle \varnothing, \{\varnothing\}\rangle\}) \\ &= \{\langle abcd, \{c, ad\}\rangle, \langle ab, \{a\}\rangle, \langle b, \{b\}\rangle, \langle d, \{d\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}\end{aligned}$$

Note that $\{\langle abcd, \{ac, ad\}\rangle\} \sqcup \{\langle abcd, \{c\}\rangle\} = \{\langle abcd, \{c, ad\}\rangle\}$.

Step 4. Considering $bd{\to}ac \in \Gamma$,

$$\texttt{Cls}(bd, \Gamma) = (abcd, \varnothing)$$

and $\texttt{MinGen}(\varnothing, \varnothing) = \{\langle \varnothing, \{\varnothing\}\rangle\}$.

$\emptyset{\to}bd$	$a{\to}b$	$c{\to}bd$	$bd{\to}ac$
$\emptyset{\to}bd$	$a{\to}\cancel{b}$	$c{\to}\cancel{b}\cancel{d}$	$\cancel{b}\cancel{d}{\to}ac$
$\emptyset{\to}abcd$			

$$\begin{aligned}\Phi &= \{\langle abcd, \{c, ad\}\rangle, \langle ab, \{a\}\rangle, \langle b, \{b\}\rangle, \langle d, \{d\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}\\ &\sqcup \texttt{Add}(\langle abcd, \{bd\}\rangle, \{\langle \varnothing, \{\varnothing\}\rangle\})\\ &= \{\langle abcd, \{c, ad, bd\}\rangle, \langle ab, \{a\}\rangle\langle b, \{b\}\rangle, \langle d, \{d\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}\end{aligned}$$

Finally, the algorithm returns

$$\{\langle abcd, \{c, ad, bd\}\rangle, \langle ab, \{a\}\rangle\langle b, \{b\}\rangle, \langle d, \{d\}\rangle, \langle \varnothing, \{\varnothing\}\rangle\}$$

3.2 Non-trivial minimal generators.

Notice that the first method we have just introduced computes all minimal generators, including trivial mingens with non-relevant information.

In this subsection, we slightly modify the previous algoritm to produce only the relevant information (not trivial mingens), by considering that $trv(M, \Gamma)$ returns always only $\{\langle \varnothing, \{\varnothing\}\rangle\}$. This new algorithm is called L $\texttt{MinGen}_0$.

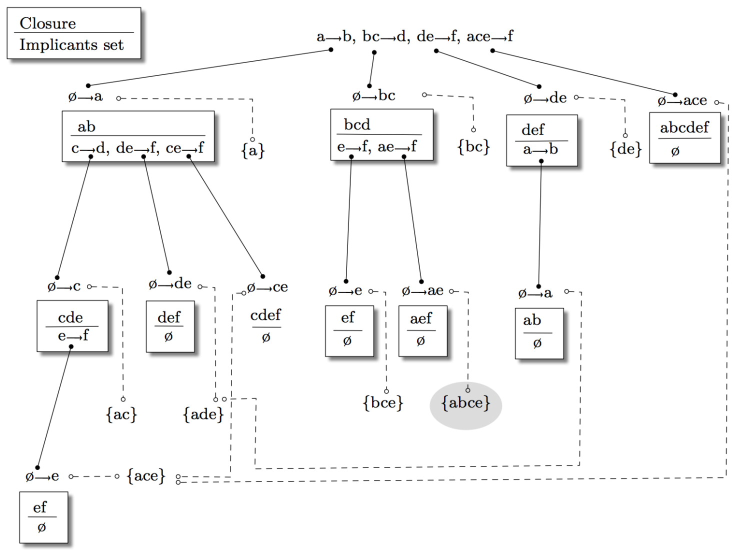


Fig. 1. $\mathbf{SL}_{FD}$ guides the construction of the MinGen set

Example 3. We want L to compute

$$\texttt{MinGen}_0(\{a,b,c,d,e,f\},\{a\to b, bc\to d, de\to f, ace\to f\})$$

The full execution of the method $\texttt{MinGen}_0$ is depicted in Figure 1. In Figure 2 we show the lattice of closed sets built with the non-trivial mingens provided by $\texttt{MinGen}_0$.

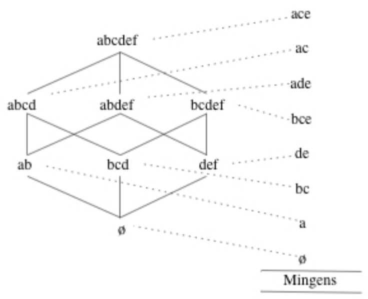


Fig. 2. Lattice of minimal generators

Step 1. $\Phi = \{\langle\varnothing,\{\varnothing\}\rangle\}$
Step 2. Considering $a\to b\in\Gamma$, compute

$$\texttt{Cls}(a,\Gamma) = (ab,\{c\to d, de\to f, ce\to f\})$$

Now we compute $\texttt{MinGen}_0(\{c,d,e,f\},\{c\to d, de\to f, ce\to f\})$
Step 2.1. $\Phi_1 = \{\langle\varnothing,\{\varnothing\}\rangle\}$
Step 2.2. Considering $c\to d\in\Gamma'$, compute $\texttt{Cls}(c,\Gamma) = (cd,\{e\to f\})$. Now we are going to compute $\texttt{MinGen}_0(\{e,f\},\{e\to f\})$.
Step 2.2.1 $\Phi_{1.1} = \{\langle\varnothing,\{\varnothing\}\rangle\}$
Step 2.2.2 Considering $e\to f\in\Gamma''$, $\texttt{Cls}(e,\Gamma'') = (ef,\varnothing)$. And so

$$\begin{aligned}\Phi_{1.1} &= \{\langle\varnothing,\{\varnothing\}\rangle\}\sqcup \texttt{Add}(\langle ef,\{e\}\rangle,\{\langle\varnothing,\{\varnothing\}\rangle\})\\ &= \{\langle ef,\{e\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\end{aligned}$$

Returning to the higher level,

$$\begin{aligned}\Phi_1 &= \{\langle\varnothing,\{\varnothing\}\rangle\}\sqcup \texttt{Add}(\langle cd,\{c\}\rangle,\{\langle ef,\{e\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\})\\ &= \{\langle cdef,\{ce\}\rangle,\langle cd,\{c\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\end{aligned}$$

Step 2.3. Considering $de\to f\in\Gamma'$, $\texttt{Cls}(de,\Gamma) = (def,\varnothing)$. Moreover, $\texttt{MinGen}_0(\{c\},\varnothing) = \{\langle\varnothing,\{\varnothing\}\rangle\}$

$$\begin{aligned}\Phi_1 &= \{\langle cdef,\{ce\}\rangle,\langle cd,\{c\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\sqcup \texttt{Add}(\langle def,\{de\}\rangle,\{\langle\varnothing,\{\varnothing\}\rangle\})\\ &= \{\langle cdef,\{ce\}\rangle,\langle def,\{de\}\rangle,\langle cd,\{c\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\end{aligned}$$

Step 2.4. Considering $ce\to f\in\Gamma'$, compute $\texttt{Cls}(ce,\Gamma) = (cdef,\varnothing)$. Moreover $\texttt{MinGen}_0(\varnothing,\varnothing) = \{\langle\varnothing,\{\varnothing\}\rangle\}$

$$\begin{aligned}\Phi_1 &= \{\langle cdef,\{ce\}\rangle,\langle def,\{de\}\rangle,\langle cd,\{c\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\sqcup \texttt{Add}(\langle cdef,\{ce\}\rangle,\{\langle\varnothing,\{\varnothing\}\rangle\})\\ &= \{\langle cdef,\{ce\}\rangle,\langle def,\{de\}\rangle,\langle cd,\{c\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\end{aligned}$$

Returning to the high level,

$$\begin{aligned}\Phi &= \{\langle\varnothing,\{\varnothing\}\rangle\} \sqcup \texttt{Add}(\langle ab,\{a\}\rangle,\{\langle cdef,\{ce\}\rangle,\langle def,\{de\}\rangle,\langle cd,\{c\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\})\\ &= \{\langle abcdef,\{ace\}\rangle,\langle abdef,\{ade\}\rangle,\langle abcd,\{ac\}\rangle,\langle ab,\{a\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\end{aligned}$$

Step 3. Considering $bc{\to}d \in \Gamma$, $\texttt{Cls}(bc,\Gamma) = (bcd,\{e{\to}f, ae{\to}f\})$. Now we compute $\texttt{MinGen}_0(\{a,e,f\},\{e{\to}f, ae{\to}f\})$.
Step 3.1. $\Phi_2 = \{\langle\varnothing,\{\varnothing\}\rangle\}$
Step 3.2. Considering $e{\to}f \in \Gamma'$, compute $\texttt{Cls}(e,\Gamma) = (ef,\varnothing)$ and $\texttt{MinGen}_0(\{a\},\varnothing) = \{\langle\varnothing,\{\varnothing\}\rangle\}$

$$\Phi_2 = \{\langle\varnothing,\{\varnothing\}\rangle\} \sqcup \texttt{Add}(\langle ef,\{e\}\rangle,\{\langle\varnothing,\{\varnothing\}\rangle\}) = \{\langle ef,\{e\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}$$

Step 3.3. Considering $ae{\to}f \in \Gamma'$, $\texttt{Cls}(ae,\Gamma) = (aef,\varnothing)$. Moreover $\texttt{MinGen}_0(\varnothing,\varnothing) = \{\langle\varnothing,\{\varnothing\}\rangle\}$

$$\begin{aligned}\Phi_2 &= \{\langle ef,\{e\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\} \sqcup \texttt{Add}(\langle aef,\{ae\}\rangle,\{\langle\varnothing,\{\varnothing\}\rangle\})\\ &= \{\langle aef,\{ae\}\rangle,\langle ef,\{e\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\end{aligned}$$

Returning to the high level,

$$\begin{aligned}\Phi = {} & \{\langle abcdef,\{ace\}\rangle,\langle abdef,\{ade\}\rangle,\langle abcd,\{ac\}\rangle,\langle ab,\{a\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\\ & \sqcup\texttt{Add}(\langle bcd,\{bc\}\rangle,\{\langle aef,\{ae\}\rangle,\langle ef,\{e\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\})\\ & = \{\langle abcdef,\{ace\}\rangle,\langle abdef,\{ade\}\rangle,\langle abcd,\{ac\}\rangle,\langle ab,\{a\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\\ & \sqcup\{\langle abcdef,\{abce\}\rangle,\langle bcdef,\{bce\}\rangle,\langle bcd,\{bc\}\rangle\}\\ & =\{\langle abcdef,\{ace\}\rangle,\langle abdef,\{ade\}\rangle,\langle bcdef,\{bce\}\rangle,\langle abcd,\{ac\}\rangle,\\ & \quad \langle bcd,\{bc\}\rangle,\langle ab,\{a\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\end{aligned}$$

Notice that the last application of the $\sqcup$ operator does not add the set $\{abce\}$ to mingen because it produces the same closed set than $\{ace\}$. Thus, in Figure 1 this leaf appears with gray color.
Step 4. Considering $de{\to}f \in \Gamma$, renders

$$\begin{aligned}\Phi = {} & \{\langle abcdef,\{ace\}\rangle,\langle abdef,\{ade\}\rangle,\langle bcdef,\{bce\}\rangle,\langle abcd,\{ac\}\rangle,\\ & \langle bcd,\{bc\}\rangle,\langle def,\{de\}\rangle,\langle ab,\{a\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\end{aligned}$$

Step 5. Considering $ace{\to}f \in \Gamma$, renders

$$\begin{aligned}\Phi = {} & \{\langle abcdef,\{ace\}\rangle,\langle abdef,\{ade\}\rangle,\langle bcdef,\{bce\}\rangle,\langle abcd,\{ac\}\rangle,\\ & \langle bcd,\{bc\}\rangle,\langle def,\{de\}\rangle,\langle ab,\{a\}\rangle,\langle\varnothing,\{\varnothing\}\rangle\}\end{aligned}$$

4 Conclusions and future works

In this work we have proposed the minimal generators as a basic notion to remove redundancy in sets of implications. To achieve this goal, the first step is to design a method to compute all minimal generators corresponding to a set of implications. The simplification paradigm is the key to design the MinGen and $\texttt{MinGen}_0$ algorithms. The application of $\mathbf{SL}_{FD}$closure algorithm guides the search of minimal generators.

As future work we will present a thorough study about the soundness, completeness, and complexity of the mingens algorithms, which have not been included here due to space limitations; moreover, a definition of basis center within the notion of minimal generators will be studied, together with a method to compute such a basis.

References

1. R. Belohlavek and V. Vychodil, Formal Concept Analysis with Constraints by Closure Operators, LNCS, 4068: 131–143, 2006.
2. R. Belohlavek and V. Vychodil, Formal Concept Analysis with Background Knowledge: Attribute Priorities. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 39: 399–409, 2009.
3. P. Cordero, M. Enciso, A. Mora, I.P, de Guzmán: SLFD logic: Elimination of data redundancy in knowledge representation, LNCS 2527: 141–150, 2002.
4. C. Frambourg, P. Valtchev, and R. Godin, Merge-based computation of minimal generators. Discrete Mathematics, LNCS 3596: 181–194, 2005.
5. B. Ganter, Two basic algorithms in concept analysis. Technische Hochschule, Darmstadt, 1984.
6. J.L. Guigues and V. Duquenne, Familles minimales d'implications informatives résultant d'un tableau de données binaires. Math. Sci. Humaines, 95, 5–18, 1986.
7. M. Hermann and B. Sertkaya, On the Complexity of Computing Generators of Closed Sets. ICFCA 2008: 158–168
8. S. O. Kuznetsov and A. Revenko, Attribute Exploration of Properties of Functions on Sets. Fundamenta Informaticae, 115 (4): 377–394, 2012.
9. C. Beeri, and R. Fagin and J.H. Howard, *A complete axiomatization for functional and multivalued dependencies in database relations*, Proc. ACM SIGMOD Conf., pp. 47-61, 1977.
10. A. Day, *The lattice theory of functional dependencies and normal decompositions*, International Journal of Algebra and Computation, 2: pp. 209–431, 1990.
11. R. Fagin, *Functional dependencies in a relational database and propositional logic*, IBM. Journal of research and development, 21 (6), (1977), pp. 534–544.
12. A. Mora, P. Cordero, M. Enciso, I.Fortes, Closure via functional dependence simplification, International Journal of Computer Mathematics, 89(4): 510–526, 2012.
13. G. Stumme, Attribute Exploration with Background Implications and Exceptions. Data Analysis and Information Systems. Statistical and Conceptual approaches. Proc. GfKl'95. Studies in Classification, Data Analysis, and Knowledge Organization 7: 457–469, 1996.
14. L. Szathmary and A. Napoli and S. O. Kuznetsov, ZART: A Multifunctional Itemset Mining Algorithm , Proc. of the 6th Intl. Conf. on Concept Lattices and Their Applications (CLA '08): 47–58, 2008.
15. L. Szathmary and P. Valtchev and A. Napoli and R. Godin, An Efficient Hybrid Algorithm for Mining Frequent Closures and Generators, Proc. of the 5th Intl. Conf. on Concept Lattices and Their Applications (CLA '07): 26–37, 2007.
16. L. Szathmary and P. Valtchev and A. Napoli and R. Godin, Efficient Vertical Mining of Frequent Closures and Generators, LNCS 5772: 393–404, 2009.
17. K. Nehmé, P. Valtchev, M. H. Rouane, R. Godin, On Computing the Minimal Generator Family for Concept Lattices and Icebergs, LNCS 3403: 192–207, 2005.
18. R. Wille, Restructuring lattice theory: an approach based on hierarchies of concepts. In I. Rival (ed.), Ordered sets, pp. 445-470, 1982.

Conceptual analysis of complex system simulation data for decision support: Application to aircraft cabin design

Nizar Messai[1], Cassio Melo[2], Mohamed Hamdaoui[2], Dung Bui[2], and Marie-Aude Aufaure[2]

[1] LI, University Franois Rabelais Tours, France
nizar.messai@univ-tours.fr
[2] MAS - Ecole Centrale Paris, Châtenay-Malabry, France
{cassio.melo,mohamed.hamdaoui,dung.bui,marie-aude.aufaure}@ecp.fr

Abstract. This paper presents a conceptual approach for decision support applied in a collaborative complex system design project. The approach takes advantage of the use of Similarity-based Formal Concept Analysis (SFCA) to classify, visualize, and explore simulation data in order to help system designers to identify relevant design choices. The approach is illustrated on an aircraft cabin design case study which concerns the simulation of different configurations of the ventilation system to study the passengers comfort in the cabin. The classification of simulation data with their corresponding comfort scores using SFCA allows to derive for each simulated input parameter the maximal interval of values which guarantee an acceptable comfort level. To evaluate the obtained results, the extracted intervals are then used as ranges of the input parameters for new simulations which confirmed the already obtained comfort levels and showed the convergence of the results.

1 Introduction

The design phase is one of the biggest challenges industrial companies are often facing in complex system production process. During the design process, several aspects must be studied to ensure the system performances as well as its compliance with the end-user requirements. These aspects are reflected by a set of design parameters that must be taken into account early from the design phase. Several simulations of design parameters are then performed before validating the design choices. In the case of complex systems, simulations produce large datasets to be carefully studied and analysed in order to identify optimal configurations. The analysis process simultaneously implies several criteria and is usually defined as multiple criteria decision analysis (MCDA) task which involves the comparison and ranking of a number of alternatives with respect to multiple, potentially conflicting, criteria, with the ultimate objective of identifying the best option from the available choices [1]. In this context, decision support tools are used to aid decision-makers to make rational choices. The range of decision support methods is very large and covers several application domains such as economy, industry, etc. [2, 3]. The choice of one method instead of another depends on several constraints related to data characteristics (data format, data size, etc.) or to the method it-self as generated result characteristics (visualization, analytics, etc.). In [1], a brief survey of the main decision support approaches in industrial contexts is given before introducing an approach based on rough sets for environmental decision support in industry. In the current work we are particularly interested in decision support approaches based on conceptual structures such as concept lattices derived based on Formal Concept Analysis (FCA) formalism [4]. These approaches are motivated by the richness of lattice structures as well as by their well-established formal properties. Indeed, a concept lattice is a conceptual representation of data that highlights its underlying structure and implicit relationships. Its usefulness for data analysis purposes is proved by the large number of derived approaches which used lattice structures as navigation support for information retrieval, as condensed representation of itemsets, implications and association rules for data mining, as a set of embedded decision trees for machine learning, prediction and decision support, etc. [5]. However, FCA method is usually limited by the rigidity of its input format (binary data). Some works have proposed to extend it to complex data [6–8], among them Similarity-based Formal Concept Analysis (SFCA) method which considers similarity to directly classify non-binary data into lattice structures called Many-Valued Concept Lattices (MV lattices) [8]. Besides extending FCA to complex data and avoiding loose of information in transformation phases, SFCA classification process produces MV lattices with different granularity levels which allows progressive data exploration [9].

In this paper we study the usefulness of MV lattices to provide a support for combining numerical values (quantitative) analysis together with qualitative analysis to aid decision makers in the process of complex system Design. More precisely we show how MV lattices can be used to highlight crucial information a designer may need to validate design choices. The proposed approach is applied to an industrial case study: the design of the ventilation system of an aircraft cabin. Simulation data for this case study are classified and analysed using MV lattices to identify relevant design configurations regarding the obtained passengers comfort in the cabin. The analysis is facilitated by two visualization techniques proposed in this work: score-based and interval-based visualization. The rest of the paper is organized as follows: Section 2 presents the context of the study and the aircraft cabin design test case. Section 3 recalls basic definitions of SFCA. Section 4 briefly describes the two visualization techniques to aid decision making in MV lattices. Section 5 details the application of SFCA to the aircraft cabin case, the obtained results and an evaluation of these results followed by the conclusion in Section 6.

2 Context of the Study: Collaborative Complex System Design

The present research work is part of the Complex System Design Lab (CSDL)[3] project which involves 27 industrial and academic partners and aims at providing a collaborative environment for complex system design. Since the simulation of design choices, which is one of the more strategic steps of complex system design, usually outputs large and high dimensional datasets, the CSDL platform should allow efficient analysis of such datasets to identify the right conception choices. In this project, simulations are

[3] http://www.systematic-paris-region.org/fr/projets/csdl

performed following a Design of Experiment procedure which consists in computing the systems outputs for a set of chosen design points of parameter space. Moreover, additional data dealing with the systems performances such as performance criteria, cost functions and constraints are computed. Then the objective of the analysis step is to identify interesting regions of the parameters space which correspond to the appropriate design configuration.

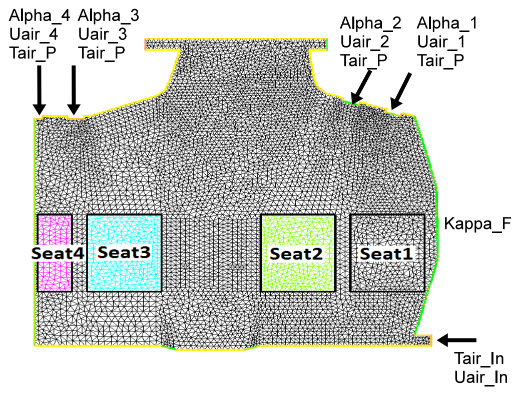


Fig. 1: Aircraft cabin schema and the main simulated input parameters

Parameter	Range
Alpha_1 (°)	[-90, -80]
Alpha_2 (°)	[-100, -45]
Alpha_3 (°)	[-135, -90]
Alpha_4 (°)	[-120, -80]
Uair_1 (m/s)	[0.2, 0.7]
Uair_2 (m/s)	[0.2, 0.7]
Uair_3 (m/s)	[0.2, 0.7]
Uair_4 (m/s)	[0.2, 0.7]
Uair_In (m/s)	[0.6, 3]
Tair_In (°C)	[22, 25]
Tair_P (°C)	[22, 25]
T_ext (°C)	[-65, -50]
Kappa_F (W/K)	[1 e^{-4}, 4 e^{-4}]

Table 1: Ranges of the simulated parameters

CSDL industrial partners have provided a use case which corresponds to a commercial aircraft cabin air control system. In this use case the goal is to identify relevant design configurations which ensure comfort conditions in terms of air temperature and velocity inside the cabin. Typical fields of temperature and velocity are obtained using the same fluid model as in [10] and the comfort design problem is parametrized by 13 continuous parameters each evolving in a range interval of possible values (see Table 1). These design parameters are: angles of air injection at 4 passengers' personal fan (Alpha_1..4), blown air speed at 4 passengers' personal fan (Uair_1..4), temperature of blown air at main inlet (Tair_In), temperature of blown air at 4 passengers' personal fan (Tair_P), blown air speed at main inlet (Uair_In), external temperature (T_ext), and fuselage thermal conductivity (Kappa_F). The mean values of temperature and velocity for each of the four passengers' seats (see Figure 1) have been computed to assess the passengers' comfort, which resulted in eight output criteria (two per passenger) related to the comfort. Moreover, a measure of the energy consumed by the air-conditioning system is also considered to estimate the price at which this comfort comes.

In the reminder of this paper we propose an approach based on SFCA for data analysis and decision support in industrial contexts through the classification and visualization of simulation datasets. The approach is illustrated on the previously introduced CSDL use case.

3 Similarity-Based Formal Concept Analysis Basic Notions

SFCA [8, 9] is a classification and data analysis method which extends FCA definitions and results to complex data represented by *many-valued contexts* [4]. Formally, an MV context is denoted by (G,M,W,I) where G is a set of objects, M is a set of attributes, W is a set of attribute values, and I is a ternary relation between G, M and W (i.e., $I \subseteq G \times M \times W$). $(g,m,w) \in I$ denotes the fact that "*the MV attribute m takes the value w for the object g*". This fact is also denoted by $m(g) = w$. Table 2 gives an example of MV context corresponding to a part of simulation results for the aircraft cabin test case (Figure 1). In this example, objects correspond to the first 5 simulations and the attributes correspond to a part of simulated input parameters, namely T_Ext, Tair_In, Tair_P, and Uair_In, and two output parameters T1 and V1 respectively corresponding to the air temperature and velocity in seat1.

Table 2: An example of Many-valued Context corresponding to a part of simulation results for seat1 in the aircraft cabin test case.

	T_Ext ($°C$)	Tair_In ($°C$)	Tair_P ($°C$)	Uair_In (m/s)	T1 ($°C$)	V1 ($10^{-2}m/s$)
1	-57.41	23.89	23.31	2.56	23.86	1.31
2	-56.68	22.71	23.11	1.95	22.34	3.47
3	-59.38	24.27	23.39	2.72	24.01	6.68
4	-51.11	22.06	23.95	1.78	21.42	2.38
5	-55.93	24.28	23.70	2.25	23.91	3.43

The basic intuition of SFCA is to group together objects which are sufficiently similar (i.e. have similar attribute values). Therefore, a set of objects A shares an attribute m when all values of m for the objects in A are similar. The similarity is defined in the common intuitive way: two values are similar when their difference is not significant. In the case of numerical data, computing similarity is straightforwardly given by a comparaison of the numerical values. Two values are said to be similar if their difference is less than a given similarity threshold θ which expresses the maximal variation allowed between two similar values. Formally, given a numerical MV context (G,M,I,W) and $w_i, w_j \in W$, $w_i \simeq w_j$ iff $|w_i - w_j| \leq \theta$. More generally, two intervals $[\alpha_1, \beta_1]$ and $[\alpha_2, \beta_2]$ are similar iff $max(\beta_1, \beta_2) - min(\alpha_1, \alpha_2) \leq \theta$. Given a similarity threshold θ, the set of all possible possible intervals of similar values that can be defined on W, denoted by $\mathfrak{I}_\theta$, is the set of intervals of the form $[w_i, w_j]$ such that $w_i, w_j \in W$ and $w_j - w_i \leq \theta$.

The choice of θ reflects the precision requirements to be considered during the data analysis process. Lower values of θ mean that only the closest values will be considered as similar whereas higher values of θ mean that more distant values can be considered as similar. Depending on considered datasets and on the analysis processes, it is possible to choose either the same similarity thresholds for all the context attributes or a separate threshold for each attribute. In the later case, θ denotes a vector $(\theta_i)_{0\leq i<|M|}$ of $|M|$ elementary thresholds respectively corresponding to the context attributes.

Based on the similarity of attribute values, SFCA extends the definition of attribute sharing between objects as follows. Given two objects $g_i, g_j \in G$ and an attribute $m \in M$ such that $m(g_i) = w_i$ and $m(g_j) = w_j$, g_i and g_j share m whenever $w_i \simeq w_j$. The interval of values $[min(w_i,w_j), max(w_i,w_j)]$ is called *similarity interval* of attribute m for objects g_i and g_j. Then g_i and g_j are said to share m w.r.t to $[min(w_i,w_j), max(w_i,w_j)]$ written as $(m,[min(w_i,w_j), max(w_i,w_j)])$. More generally, a set of objects A shares $(m,[\alpha,\beta])$ where $\alpha = min_{g\in A}(m(g))$ and $\beta = max_{g\in A}(m(g))$ whenever $\forall g_i, g_j \in A$, $m(g_i) \simeq m(g_j)$. In this case, A is said to be *valid* w.r.t. m and $[\alpha,\beta]$ is the similarity interval of m for A. In the same way, A shares a set of attributes B whenever A shares all attributes in B. In the MV context given in Table 2 objects 1 and 3 share attribute T1 for a similarity threshold $\theta_4 = 1$. However these two objects do not share attribute V1 for $\theta_5 = 1$.

A *many-valued concept* is defined as (i) maximal sets of objects having in common (ii) maximal sets of attributes with intervals of similar values. These sets are formally defined as follows.

(i) *Maximal valid sets of objects*: Given an attribute m and a set of objects A valid w.r.t. m. SFCA defines the set of reachable objects from A w.r.t. m as :

$$\mathfrak{R}(A,m) = \{g_i \in G \mid m(g_i) \simeq m(g),\ \forall g \in A\}$$

$\mathfrak{R}(A,m)$ is the maximal set containing all objects similar to those in A w.r.t. m. This set may not be valid w.r.t. m because of the non transitivity of "$\simeq$". The maximal valid set of objects containing A is the subset of $\mathfrak{R}(A,m)$ obtained by removing from $\mathfrak{R}(A,m)$ all pairs of objects which do not share m. Formally this set is defined as follows:

$$\mathfrak{R}_v(A,m) = \mathfrak{R}(A,m) \setminus \{g_i, g_j \in \mathfrak{R}(A,m) \mid m(g_i) \not\simeq m(g_j)\}$$

More generally, the maximal valid set containing A with respect to $B \subseteq M$ is:

$$\mathfrak{R}_v(A,B) = \bigcap_{m\in B} \mathfrak{R}_v(A,m)$$

(ii) *Maximal intervals of similar attribute values*: When $A \subseteq G$ shares an attribute $m \in M$, the largest interval of similar values of m for A is the interval of similar values of m for the objects in $\mathfrak{R}_v(A,m)$ obtained as follows:

$$\gamma(A,m) = [min_{g\in\mathfrak{R}_v(A,m)}(m(g)), max_{g\in\mathfrak{R}_v(A,m)}(m(g))]$$

Then A is said to share $(m,\gamma(A,m))$. For example, for $\theta_0 = 1$, the set of objects $\{1,3,5\}$ shares $(T1,[23.86,24.01])$.

Based on these maximal sets, SFCA defines the following derivation operators for $A \subseteq G$ and $B \subseteq M \times \mathfrak{I}_\theta$:

$$A^{\uparrow} = \{(m,\gamma(A,m)) \in M \times \mathfrak{I}_\theta \mid \gamma(A,m) \neq \emptyset\}$$

$$B^{\downarrow} = \mathfrak{R}_v(\{g \in G \mid \forall\ (m,[\alpha,\beta]) \in B,\ m(g) \simeq [\alpha,\beta]\}, B)$$

$A^{\uparrow}$ is the maximal set of MV attributes shared by all objects in A and $B^{\downarrow}$ is the maximal set of objects sharing all MV attributes in B. It has been shown in [8,9] that $^{\uparrow}$ and $^{\downarrow}$ form a Galois connection between $(G, \subseteq)$ and $(M \times \mathfrak{I}_\theta, \subseteq_\theta)$.

In the example of MV context in Table 2, and for $\theta = (10,10,10,10,1,1)$, we have: $\{1,3,5\}^{\uparrow} = \{(T1,[23.86,24.01]), (T\_Ext,[-59.38,-55.93]), (Tair\_In,[23.89,24.28]), (Tair\_P,[23.31,23.7]), (Uair\_In,[2.25,2.72])\}$
and
$\{(T1,[23.86,24.01]), (T\_Ext,[-59.38,-55.93]), (Tair\_In, [23.89,24.28]), (Tair\_P, [23.31,23.7]), (Uair\_In,[2.25,2.72])\}^{\downarrow} = \{1,3,5\}$.

MV concepts are then defined as pairs (A,B) where $A \subseteq G$ and $B \subseteq M \times \mathfrak{I}_\theta$ such that $A^{\uparrow} = B$ and $B^{\downarrow} = A$. A and B are respectively the extent and the intent of (A,B). For $\theta = (10,10,10,10,1,1)$ in the MV context given in Table 2, $(\{1,2,3\}, \{(T1,[23.86,24.01]), (T\_Ext,[-59.38,-55.93]), (Tair\_In,[23.89,24.28]), (Tair\_P,[23.31,23.7]), (Uair\_In,[2.25,2.72])\})$ is an example of MV concept.

The MV concepts of an MV context can be partially ordered based on the inclusion of their extents (and, dually, intents) and form the hierarchy structure called *MV concept lattice* and denoted by $\underline{\mathfrak{B}}_\theta(G,M,W,I)$. The Hasse diagram of the MV concept lattice of the MV context given in Table 2 for $\theta = (10,10,10,10,1,1)$ is shown in Figure 2. In this graphical representation, MV attributes in the form $(m,[\alpha,\beta])$ are written $m : [\alpha,\beta]$ for a better readability.

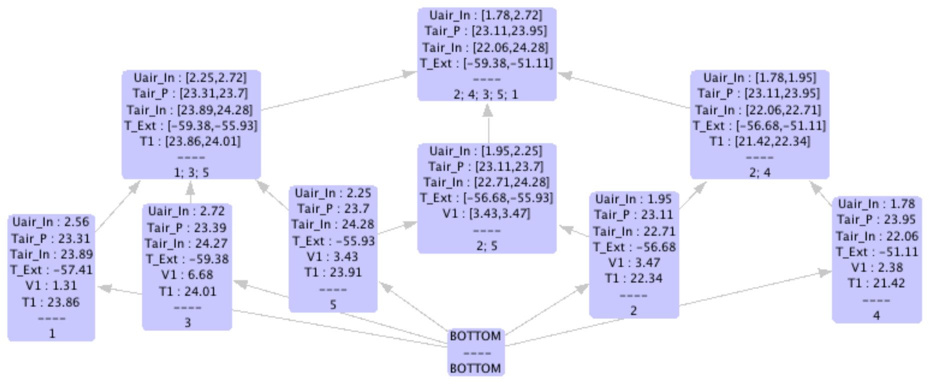


Fig. 2: The MV lattice $\underline{\mathfrak{B}}_\theta(G,M,W,I)$ corresponding to the MV context given in Table 2 for $\theta = (10,10,10,10,1,1)$.

4 Visualization of MV Concepts

Finding patterns within MV concepts may be assisted with the use of visualization techniques. In this work we propose two techniques to assist the analyst in determining the similarity threshold, filtering and selecting concepts of interest. The proposed techniques follow the so called "information-seeking mantra" - Overview first, zoom and filter, then details-on-demand [11].

First we propose a visualization technique that consists on the filtering and coloring of concepts based on user-defined scores. In Section 2, we show how certain intervals in the temperature and the air speed are used to determine comfort classes in the cabin according to international standards. A color gradient is assigned according to the "global score" of a concept (Figure 2). For instance, the score of "maximum comfort" is attributed when the scores of velocity and and temperature are equal to 2 and the color. This simplification is important because in Complex System Design the number of parameters are usually overwhelming to the analyst to overview and compare. Alternatively, user can filter concepts that are below a score threshold. This is particularly important during the extraction of comfort classes as explained in Section 2.

When the relevant concepts are identified, i.e. the comfort classes in our case, the analyst will take decisions based on concepts that fit best to his or her goals. To help the analyst to quick identify intervals in the concepts and compare with other concepts in the lattice, we created a new visualisation based on a "conceptual heat map" where each concept is depicted as an array of rectangles (Figure 3). Each rectangle represents an attribute, its color indicates the interval of the attribute value in a continuous color scale from blue to red. Its width is proportional to the size of the range. If an attribute is not present in the concept the corresponding rectangle is empty in order to keep the order of attributes consistent.

5 Applying SFCA to Aircraft Cabin Test Case

5.1 Using Similarity Threshold to Express Constraints to Guide Data Exploration Process

In the previous section, SFCA formalisation is given from a numerical data perspective. Such a formalisation can also be done on other data formats which makes SFCA approach generic and flexible. In this case, appropriate similarity measures need to be defined accordingly to handle complex data. Once similarity measures are defined, the application of SFCA follows the intuition of "grouping together similar data". As shown above, this operation also needs the definition of threshold which can be a subjective task strongly related to the data analysis process. The variation of similarity thresholds yields a variation in the obtained MV lattices in terms of number of MV concepts as well as in terms of concept granularity and it has been shown that both aspects are related [9].

In the current work, we are interested in identifying the input requirements which guarantee a convenient output situations based on a set of simulation data. This means that we first need to formulate "convenient" output situations in terms of constraints. Then, we use such constraints to guide the SFCA classification process in order to obtain the appropriate values of input parameters satisfying these constraints. In the previously shown example (Table 2 and Figure 2) the choice of the threshold $\theta = (10,10,10,10,1,1)$ follows the idea of defining constraints on the output parameters $T1$

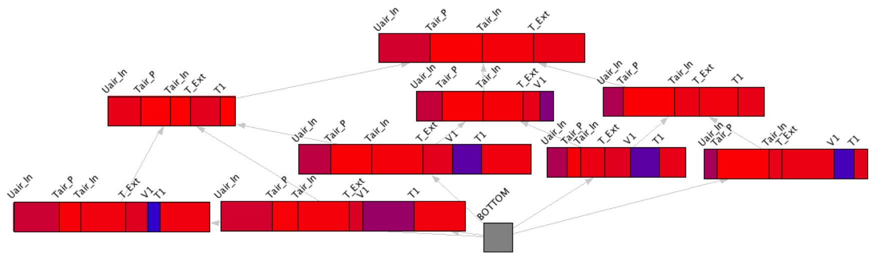


Fig. 3: The MV concept view for the lattice in Figure 2. Color indicates position in the range (from blue to red), width shows the length of the interval.

and $V1$ in order to extract the appropriate ranges of the input parameters T_Ext, $Tair\_In$, $Tair\_P$ and $Uair\_In$. Indeed, value of $\theta_i = 10$ defined as threshold for each input parameter exceeds the difference between the minimal and the maximal values of each of the four input parameters. Consequently, there is no effective constraint in grouping together these parameters while forming the MV concepts. However, the thresholds $\theta_i = 1$ for $T1$ and $V1$ mean that values of $T1$ (respectively $V1$) can be grouped together if and only if their difference is less than 1. This constraint is expressed to obtain a lattice formed by MV concepts where intervals of values of $T1$ and $V1$ are not larger than 1 and without any constraint on the other attributes. Having such a MV lattice one can directly read the maximal range of each input parameter which allows to obtain a temperature in given interval. For example, MV concept with extent $\{2,4\}$ in Figure 2 shows that values of $Uair\_In$, $Tair\_P$, $Tair\_In$ and T_ext respectively in intervals [1.78,1.95], [23.11,23.95], [22.6,22.71], and [-56.68,-51.11] ensure to obtain a value of T1 in [21.42,22.34]. Knowing intervals of temperature values corresponding to a comfort situation, it is then possible to deduce the required inputs to ensure such a comfort. Based on this idea we develop and apply a data analysis strategy to deal with the aircraft cabin design test case.

5.2 Simulation Dataset of the Aircraft Cabin Test Case

In the reminder of this paper we will consider the previously introduced aircraft design case study. The considered dataset corresponds to the simulation results of 100 randomly chosen configurations of design parameters (the 13 input parameters). 9 output criteria have been defined to assess the quality of each configuration in terms of passengers comfort and energy cost. The mean values of temperature and velocity of each of the four seats are computed which resulted in 8 criteria associated with the comfort of the passengers. The dissipated energy is computed based on the velocity as a measure of the loss of energy due to the fluid viscosity.

In order to quickly appreciate the comfort in each seat and hence simplify the dataset exploration and the experiments evaluation, comfort scores were computed for the values of the comfort output criteria (temperature and velocity). The scores are in a three points scale (0: uncomfortable, 1: acceptable, 2: comfortable) computed according to

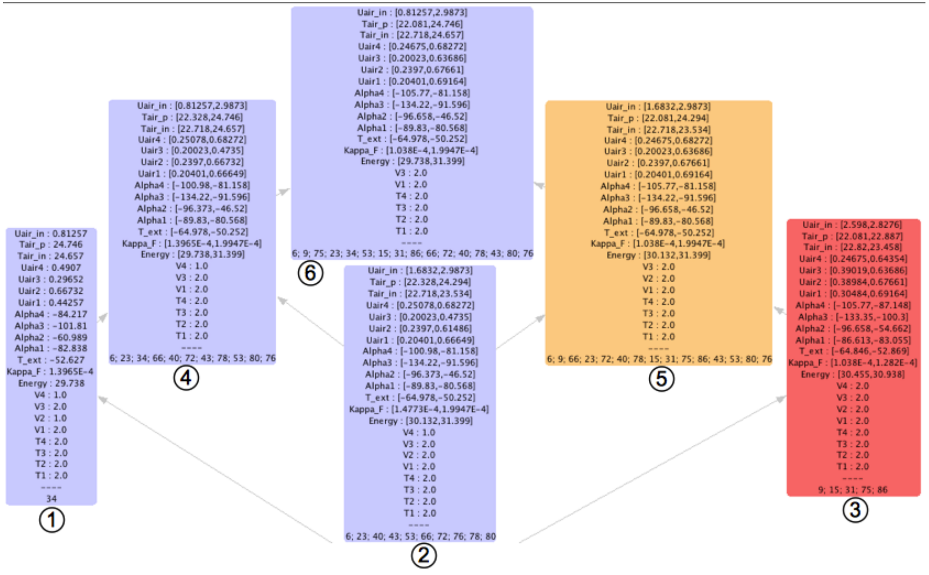
ANSI/ASHRAE Standards [12] as follows:

$$\text{score(T)} = \begin{cases} 0 \text{ if } T < 21 \text{ or } T > 24 \\ 1 \text{ if } 21 \leq T < 22.5 \text{ or } 23.5 < T \leq 24 \\ 2 \text{ if } 22.5 \leq T \leq 23.5 \end{cases} \quad \text{score(V)} = \begin{cases} 0 \text{ if } V > 1 \\ 1 \text{ if } 0.2 < V \leq 1 \\ 2 \text{ if } V \leq 0.2 \end{cases}$$

These scores are then used instead of their corresponding values of temperature and velocity in the classification process by SFCA.

5.3 Extracting Comfort Classes and Their Corresponding Design Parameters' Ranges

Our objective is to determine the design parameters that are important to qualify the experiments such that all of the four passengers are satisfied. We make the assumption that the temperature is more important than the velocity to define the thermal comfort of the passengers. Therefore, we focus our analysis on the experiments that offer the maximum comfort for the passengers from the temperature point of view only: we keep only the experiments such that the score for the temperature is 2 and we called this subset S_{silver}. Then, following the previously detailed strategy for thresholds choice, we applied SFCA to build the corresponding MV lattice shown on Figure 4.


Uair_in : 0.81257
Tair_p : 24.746
Tair_in : 24.657
Uair4 : 0.4907
Uair3 : 0.29652
Uair2 : 0.66732
Uair1 : 0.44257
Alpha4 : -84.217
Alpha3 : -101.81
Alpha2 : -60.989
Alpha1 : -82.838
T_ext : -52.627
Kappa_F : 1.3965E-4
Energy : 29.738
V4 : 1.0
V3 : 2.0
V2 : 1.0
V1 : 2.0
T4 : 2.0
T3 : 2.0
T2 : 2.0
T1 : 2.0

34
1
Uair_in : [0.81257,2.9873]
Tair_p : [22.328,24.746]
Tair_in : [22.718,24.657]
Uair4 : [0.25078,0.68272]
Uair3 : [0.20023,0.4735]
Uair2 : [0.2397,0.66732]
Uair1 : [0.20401,0.66649]
Alpha4 : [-100.98,-81.158]
Alpha3 : [-134.22,-91.596]
Alpha2 : [-96.373,-46.52]
Alpha1 : [-89.83,-80.568]
T_ext : [-64.978,-50.252]
Kappa_F : [1.3965E-4,1.9947E-4]
Energy : [29.738,31.399]
V4 : 1.0
V3 : 2.0
V1 : 2.0
T4 : 2.0
T3 : 2.0
T2 : 2.0
T1 : 2.0

6; 23; 34; 66; 40; 72; 43; 78; 53; 80; 76
4
Uair_in : [0.81257,2.9873]
Tair_p : [22.081,24.746]
Tair_in : [22.718,24.657]
Uair4 : [0.24675,0.68272]
Uair3 : [0.20023,0.63686]
Uair2 : [0.2397,0.67661]
Uair1 : [0.20401,0.69164]
Alpha4 : [-105.77,-81.158]
Alpha3 : [-134.22,-91.596]
Alpha2 : [-96.658,-46.52]
Alpha1 : [-89.83,-80.568]
T_ext : [-64.978,-50.252]
Kappa_F : [1.038E-4,1.9947E-4]
Energy : [29.738,31.399]
V3 : 2.0
V1 : 2.0
T4 : 2.0
T3 : 2.0
T2 : 2.0
T1 : 2.0

6; 9; 75; 23; 34; 53; 15; 31; 86; 66; 72; 40; 78; 43; 80; 76
6
Uair_in : [1.6832,2.9873]
Tair_p : [22.328,24.294]
Tair_in : [22.718,23.534]
Uair4 : [0.25078,0.68272]
Uair3 : [0.20023,0.4735]
Uair2 : [0.2397,0.61486]
Uair1 : [0.20401,0.66649]
Alpha4 : [-100.98,-81.158]
Alpha3 : [-134.22,-91.596]
Alpha2 : [-96.373,-46.52]
Alpha1 : [-89.83,-80.568]
T_ext : [-64.978,-50.252]
Kappa_F : [1.4773E-4,1.9947E-4]
Energy : [30.132,31.399]
V4 : 1.0
V3 : 2.0
V2 : 2.0
V1 : 2.0
T4 : 2.0
T3 : 2.0
T2 : 2.0
T1 : 2.0

6; 23; 40; 43; 53; 66; 72; 76; 78; 80
2
Uair_in : [1.6832,2.9873]
Tair_p : [22.081,24.294]
Tair_in : [22.718,23.534]
Uair4 : [0.24675,0.68272]
Uair3 : [0.20023,0.63686]
Uair2 : [0.2397,0.67661]
Uair1 : [0.20401,0.69164]
Alpha4 : [-105.77,-81.158]
Alpha3 : [-134.22,-91.596]
Alpha2 : [-96.658,-46.52]
Alpha1 : [-89.83,-80.568]
T_ext : [-64.978,-50.252]
Kappa_F : [1.038E-4,1.9947E-4]
Energy : [30.132,31.399]
V3 : 2.0
V2 : 2.0
V1 : 2.0
T4 : 2.0
T3 : 2.0
T2 : 2.0
T1 : 2.0

6; 9; 66; 23; 72; 40; 78; 15; 31; 75; 86; 43; 53; 80; 76
5
Uair_in : [2.598,2.8276]
Tair_p : [22.081,22.887]
Tair_in : [22.82,23.458]
Uair4 : [0.24675,0.64354]
Uair3 : [0.39019,0.63686]
Uair2 : [0.38984,0.67661]
Uair1 : [0.30484,0.69164]
Alpha4 : [-105.77,-87.148]
Alpha3 : [-133.35,-100.3]
Alpha2 : [-96.658,-54.662]
Alpha1 : [-86.613,-83.055]
T_ext : [-64.846,-52.869]
Kappa_F : [1.038E-4,1.282E-4]
Energy : [30.455,30.938]
V4 : 2.0
V3 : 2.0
V2 : 2.0
V1 : 2.0
T4 : 2.0
T3 : 2.0
T2 : 2.0
T1 : 2.0

9; 15; 31; 75; 86
3

Fig. 4: MV lattice generated on S_{silver}

By analyzing this MV lattice, it turns out that S_{silver} can be described using three distinct main comfort classes: (i) "*Maximum comfort*" (concept $n°3$): maximum score for both temperature and velocity for the 4 seats, (ii) "*Intermediate comfort*" (concepts $n°2$ and 5): the score for velocity for seat 4 is not maximum, and (iii) "*Poor comfort*" concept (concept $n°2$, 4, and 6): the score for velocity for seats 2 and 4 is not maximum.

These three comfort classes can be directly read on Figure 4. Each class is given by one MV concept in the lattice and the order defined on the MV concepts also holds for the comfort classes : Concepts 3, 5 and 6 corresponding respectively to maximum, intermediate, and poor comfort classes. The colors linked to concept scores in Figures 4 and 3 allow to quickly identify the concepts corresponding to the most relevant design configurations and to extract ranges of variation of the corresponding design parameters. These extracted ranges are given in Table 3.

Table 3: Extracted ranges of the 13 comfort design problem parameters for different classes of comfort.

Parameter	Range	"Maximum comfort" range	"Intermediate comfort" range	"Poor comfort" range
Alpha_1 (°)	[-90, -80]	[-86.61, -83.05]	[-89.83, -80.56]	[-89.83, -80.56]
Alpha_2 (°)	[-100, -45]	[-96.65, -54.66]	[-96.65, -46.52]	[-96.65, -46.52]
Alpha_3 (°)	[-135, -90]	[-133.35, -100.3]	[-134.22, -91.59]	[-134.22, -91.59]
Alpha_4 (°)	[-120, -80]	[-105.77, -87.14]	[-105.77, -81.15]	[-105.77, -81.15]
Uair_1 (m/s)	[0.2, 0.7]	[0.30, 0.69]	[0.20, 0.69]	[0.20, 0.69]
Uair_2 (m/s)	[0.2, 0.7]	[0.38, 0.67]	[0.23, 0.67]	[0.23, 0.67]
Uair_3 (m/s)	[0.2, 0.7]	[0.39, 0.63]	[0.20, 0.63]	[0.20, 0.63]
Uair_4 (m/s)	[0.2, 0.7]	[0.24, 0.64]	[0.24, 0.68]	[0.24, 0.68]
Uair_In (m/s)	[0.6, 3]	[2.59, 2.82]	[1.68, 2.98]	[0.81, 2.98]
Tair_In (°C)	[22, 25]	[22.82, 23.45]	[22.71, 23.53]	[22.71, 24.65]
Tair_P (°C)	[22, 25]	[22.08, 22.88]	[22.08, 24.29]	[22.08, 24.74]
T_ext (°C)	[-65, -50]	[-64.84, -52.86]	[-64.97, -50.25]	[-64.97, -50.25]
Kappa_F (W/K)	[$1e^{-4}$, $4e^{-4}$]	[$1.03e^{-4}$, $1.28e^{-4}$]	[$1.03e^{-4}$, $1.99e^{-4}$]	[$1.03e^{-4}$, $1.99e^{-4}$]

5.4 Evaluating the Extracted Design Parameters' Ranges Through New Simulations

In order to evaluate the obtained results, we considered the extracted ranges of input parameters corresponding to the maximum comfort class for new simulations to check wether the output values of temperature and velocity belong to the same comfort class. We performed 12 new simulations for which input parameters take randomly chosen values in the ranges of the maximal comfort class previously extracted. All of the twelve simulations resulted in maximal comfort values for temperature and velocity mean values for each of the four seats. That is, in each seat, the mean value of temperature is between 22.5°C and 23.5°C and the mean value of velocity is less than 0.2. The obtained values are in the following ranges: T1: [22.57, 23.16], T2: [22.64, 23.30], T3: [22.84, 23.40], T4: [22.76, 23.29], V1: [0.013, 0.041], V2: [0.016, 0.062], V3: [0.001, 0.004], and V4: [0.148, 0.2].

The ranges of the input parameters for the new simulations extracted from this lattice are shown in Table 4 (column: New range).

Compared to the ranges previously obtained for the maximum comfort class (also given in Table 4, column: Maximum comfort range), the new ranges are almost the same as the maximum comfort ranges for all the input parameters. As the new simulations resulted in values of temperature and velocity in full compliance with the maximum comfort class values, this means that the simulations converge to the ranges obtained for the maximum comfort class. These ranges can then be confirmed by the system designer as one possible optimal solution for the multidimensional problem of passengers comfort in the aircraft cabin.

We proceeded in the same way to evaluate the extracted ranges of the intermediate comfort class. The twelve simulations performed for values of input parameters randomly chosen in the ranges corresponding to the intermediate comfort class have produced values of temperature in the interval [22.5,23.5]. However velocity values in seat 4 resulted in values in the interval [0.2,1] for eleven simulations which correspond to intermediate comfort. These results also confirm the ones previously obtained. In addition, input parameter ranges are included or equal to the ones obtained in the previous case which means that the simulations converge as in the case of maximum comfort class.

The evaluation of the extracted parameters ranges for the poor comfort class in the same way produced similar results and confirmed the convergence of the simulations to the solution extracted from the MV lattice given in Figure 4.

These results show the usefulness of the presented approach for supporting a system designer to improve and validate the choice for a design solution of a complex system based on simulation results. In addition, the genericity and the flexibility of SFCA formalism makes it possible to adapt the presented approach and apply it to similar problems.

6 Conclusion

In this paper we presented an approach based on conceptual structures to support complex system designers in the identification of relevant design configurations. The approach takes advantage of SFCA formalism to study the thermal comfort of 4 passengers in an aircraft cabin whose ventilation system is parametrized by 13 design parameters. The design problem takes into account 9 decision criteria defining the passengers comfort and energy consumption of the air ventilation system. A dataset of 100 randomly chosen simulated configurations is considered for the aircraft case study. The use of SFCA following a smart analysis strategy through the choice of appropriate similarity thresholds allows to directly deduce the ranges of values of input parameters which guarantee different levels of passengers comfort. In addition, an adapted visualisation of MV lattices delivers a tractable and easy way to read them easing the design and decision making process. The obtained results are then evaluated by new simulations which converged to the same solutions in terms of passengers comfort as well as in terms of input parameters ranges.

Table 4: Extracted ranges of the 13 comfort design problem parameters for the maximum comfort class and for the new simulations.

Parameter	"Maximum comfort" range	New range
Alpha_1 (°)	[-86.61, -83.05]	[-86.27, -83.30]
Alpha_2 (°)	[-96.65, -54.66]	[-95.38, -56.82]
Alpha_3 (°)	[-133.35, -100.3]	[-131.51, -103.58]
Alpha_4 (°)	[-105.77, -87.14]	[-103.08, -87.87]
Uair_1 (m/s)	[0.30, 0.69]	[0.33, 0.68]
Uair_2 (m/s)	[0.38, 0.67]	[0.39, 0.60]
Uair_3 (m/s)	[0.39, 0.63]	[0.39, 0.63]
Uair_4 (m/s)	[0.24, 0.64]	[0.26, 0.59]
Uair_In (m/s)	[2.59, 2.82]	[2.61, 2.82]
Tair_In ($°C$)	[22.82, 23.45]	[22.86, 23.42]
Tair_P ($°C$)	[22.08, 22.88]	[22.14, 22.88]
T_ext ($°C$)	[-64.84, -52.86]	[-64.27, -54.27]
Kappa_F (W/K)	[$1.03e^{-4}$, $1.28e^{-4}$]	[$1.07e^{-4}$, $1.27e^{-4}$]

References

1. K. Aviso, R. Tan, and A. Culaba, "Application of rough sets for environmental decision support in industry," *Clean Technologies and Environmental Policy*, vol. 10, pp. 53–66, 2008.
2. J. Figueira, S. Greco, and M. Ehrgott, *Multiple Criteria Decision Analysis: State of the Art Surveys*. Springer, 2005.
3. M. Ehrgott, J. Figueira, and S. Greco, *Trends in Multiple Criteria Decision Analysis*. Springer, 2010.
4. B. Ganter and R. Wille, *Formal Concept Analysis*, mathematical found. ed. Springer, 1999.
5. B. Ganter, G. Stumme, and R. Wille, Eds., *Formal Concept Analysis, Foundations and Applications*, ser. LNCS, vol. 3626. Springer, 2005.
6. S. Ferré and O. Ridoux, "A logical generalization of formal concept analysis," in *ICCS 2000*, ser. LNCS, vol. 1867. Springer, pp. 371–384.
7. B. Ganter and S. O. Kuznetsov, "Pattern structures and their projections," in *ICCS 2001*, ser. LNCS, vol. 2120. Springer, pp. 129–142.
8. N. Messai, M.-D. Devignes, A. Napoli, and M. Smaïl-Tabbone, "Many-valued concept lattices for conceptual clustering and information retrieval," in *18th European Conference on Artificial Intelligence ECAI 2008*, vol. 178. IOS Press, pp. 127–131.
9. N. Messai, M.-D. Devignes, A. Napoli, and M. Tabbone, "Using domain knowledge to guide lattice-based complex data exploration," in *ECAI 2010*, vol. 215. IOS Press, pp. 847–852.
10. D. Bui, M. Hamdaoui, and F. de Vuyst, "Reduced-order modeling of parametrized finite element solutions by POD-ISAT technique. application to aircraft air control system," in *COUPLED PROBLEMS 2011*.
11. D. A. Keim, F. Mansmann, J. Schneidewind, H. Ziegler, and J. Thomas, "Visual analytics: Scope and challenges," December 2008, springer, Lecture Notes In Computer Science (lncs).
12. ASHRAE, "ASHRAE Standard, ANSI/ASHRAE Standard 55-2004: thermal environmental conditions for human occupancy," Tech. Rep., 2004.

A Hybrid Classification Approach based on FCA and Emerging Patterns - An application for the classification of biological inhibitors

Yasmine Asses[1], Aleksey Buzmakov[1,2], Thomas Bourquard[1], Sergei O. Kuznetsov[2], and Amedeo Napoli[1]

[1] LORIA (CNRS - Inria NGE - U. de Lorraine), Vandoeuvre les Nancy, France
[2] National Research University Higher School of Economics, Moscow, Russia

Abstract. Classification is an important task in data analysis and learning. Classification can be performed using supervised or unsupervised methods. From the unsupervised point of view, Formal Concept Analysis (FCA) can be used for such a task in an efficient and well-founded way. From the supervised point of view, emerging patterns rely on pattern mining and can be used to characterize classes of objects w.r.t. a priori labels. In this paper, we present a hybrid classification method which is based both on supervised and unsupervised aspects. This method relies on FCA for building a concept lattice and then detects the concepts whose extents determines classes of objects sharing the same labels. These classes can then be used as reference classes for classifying unknown objects. This hybrid approach has been used in an experiment in chemistry for classifying inhibitors of the c-Met protein which plays an important role in protein interactions and in the development of cancer.

Keywords: Formal Concept Analysis, supervised classification, unsupervised classification, emerging patterns, pattern mining

1 Introduction

In this paper, we present a classification approach based on a combination of knowledge discovery methods which are all interconnected. This approach has to guide two processes, classification and prediction, for analyzing the c-Met receptor protein, a molecule showing an abnormally elevated expression in cancer disease [1]. Activation of this receptor can be inhibited by different biochemical compounds (inhibitors). We collected a group of 100 molecules ("complete set of inhibitors") which are known to be c-Met inhibitors. Inhibitors act on c-Met through a "binding pocket" and an associated "binding mode". We know the binding modes for 30 inhibitors of the dataset (so called "training set"). According to the spatial regions involved in the binding pocket, three main binding modes have been determined: "Type-1", "DFG-out", and "C-Helix-out" (the names are given w.r.t. spatial configuration of proteins). The "Type-1" binding mode is very mixed, probably meaning that it should be divided into more specialized modes. Chemists are working on the definition of a fourth binding mode, close to "Type-1" and termed as "Type-1bis".

To ensure the best and adapted inhibition, it is important to know the binding mode of an inhibitor, and this can only be done through chemical experiments, which are long and expensive. Thus, two main questions arise here:

- Is it possible to classify the complete inhibitor set of 100 molecules according to the functionality (based on functional groups) and particular substructures detected in the 30 molecules of the training set?
- Is it possible then to predict the binding mode or "class" of the 70 inhibitors based on the classification of the complete inhibitors set?

For answering the two questions, we introduce a combined classification/prediction process involving supervised and unsupervised classification within the framework of FCA, graph mining and the so-called "Jumping Emerging Patterns" (JEPs). More precisely, we first want to classify a set of molecules (of the training set) according to their structure and their functionality (the functionality determines the behavior of a molecule during reaction and is linked to special substructures called functional groups). For analyzing the structures of the molecules in the training set, we consider molecules as graphs and apply graph mining techniques [2, 3] to extract frequent substructures. Then, these substructures are used as attributes in a formal context where objects are molecules of the training set. This formal context is "augmented" in the sense that each molecule in the training set has a "type" or a "class" according to its binding mode. A concept lattice is built from the formal concept. Moreover, the class information is used for characterizing the concepts whose extents include objects of a single class or binding mode. The intents of these particular concepts are JEPs. Closed JEPs have already been studied in the framework of FCA (see [4–6], where they are called JSM-hypotheses). The set of all JEPs forms a "disjunctive version space" which was related to FCA in [7].

The last step involves a "hierarchical agglomerative clustering" process. Based on the knowledge of JEPs and of functional groups, inhibitors are represented as vectors where components are filled with functional groups and JEPs (55 components where 42 are functional groups and 13 are JEPs). The cosine similarity is used for building a dendrogram which is used for explaining the "proximity" of some inhibitors and for predicting the binding mode of inhibitors for which this information is still unknown.

This classification process which calls for a variety of knowledge discovery methods is totally original and is designed for solving a real-world problem. Here, an original combination of supervised and unsupervised classification works in relation with graph mining and clustering. This shows also the flexibility of the FCA process to be combined with other classification methods for giving actual and substantial results. Experiments are still running but preliminary results have been reached and show that the approach should be continued and improved.

The paper is organized as follows. In Section 2 a motivating example is introduced. Then Section 3 describes the classification flow. Section 4 introduces the main definitions on FCA, JEPs and how they are extracted. Section 5 details

	H	CAD	OH	P	AAE	F	O=
319	x	x			x	x	x
320	x	x		x		x	x
L5G		x	x				x
ZZY	x			x	x	x	

(a) Formal Context

Molecule	Binding Mode
319	DFG-out
320	DFG-out
L5G	Type-1
ZZY	Type-1

(b) Molecule Binding Modes

Table 1: Running Example. In 1a, objects (the rows) are molecules; attributes (the columns) are functional groups. A cross in the cell (i, j) means that the molecule i includes the functional group j as a substructure. In 1b the last column designates the "class" of an object, i.e. the binding mode of the molecule.

the preparation of the molecular data that are processed with FCA. The clustering method and its application are following. The main results are discussed in Section 7 before the conclusion of the paper.

2 Running Example

Formal Concept Analysis (FCA) is briefly introduced hereafter. FCA is based on a formal context which is a triple (G, M, I), where G is a set of objects, M is a set of attributes and $I \subseteq G \times M$ is a relation between G and M [8].

A running example is shown in Table 1. Molecules are objects which are described by substructures, corresponding to attributes. The selection of these particular substructures is discussed later.

Concept	A Set of Molecules (Extent)	A Set of Substructures (Intent)
C_0		H, CAD, OH, P, AAE, F, O=
C_1	ZZY	H, P, AAE, F
C_2	319	H, CAD, P, F, O=
C_3	320	H, CAD, AAE, F, O=
C_4	L5G	CAD, OH, O=
C_5	319, 320	H, CAD, F, O=
C_6	320, ZZY	H, P, F
C_7	319, ZZY	H, AAE, F
C_8	319, 320, L5G	CAD, O=
C_9	319, 320, ZZY	H, F
C_{10}	319, 320, L5G, ZZY	

Table 2: A set of formal concepts w.r.t context on Table 1a.

For every set of molecules A it is possible to find the maximal set of substructures B, included into every molecule from A. This operation is denoted as $(\cdot)'$ with $B = A'$. For example, molecules 319 (BMS_WO/2005/117867_24) and ZZY (UCB_Celltech_azaindole) include the following substructures: H (Halogen), AAE (Alkyl Aryl Ether) and F (Figure 2a). Dually, for every set of substructures B it is possible to find the maximal set of molecules, including all substructures from B, denoted by $A = B'$. Substructures CAD (Carboxilic Acid Derivative) and O= (Figure 2b) are included into molecules 319 , 320 (molecule BMS_WO/2006-/004636_132) and L5G (Amgen_WO/2008/008539_123). The attribute P stands for substructure Primary Amine while the attribute OH stands for OH-Compound.

The pairs (A, B) –where A is a set of molecules and B is a set of substructures– such that $A' = B$ and $A = B'$ are called "formal concepts". The set A is the extent and the set B is the intent of the concept. The whole set of formal concepts for the running example is given in Table 2.

Formal concepts are partially ordered w.r.t. inclusion of set of objects or of set of attributes: $(A_1, B_1) \leq (A_2, B_2)$ iff $A_1 \subseteq A_2$ or dually $B_2 \subseteq B_1$. This partial ordering gives rise to a concept lattice. Figure 1 shows the concept lattice related to the running example, where reduced notation is used. There are many algorithms for computing formal concepts and the associated concept lattice [9–11].

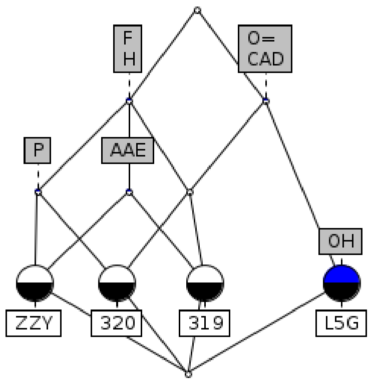


Fig. 1: The FCA-lattice for the context on Table 1.

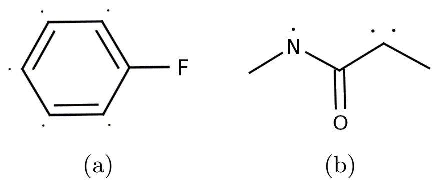


Fig. 2: Some substructures for running example in Table 1.

Additional information associated with the molecules is given in Table 1b. The table indicates the binding mode of the molecule with the c-Met protein. This additional column will allow us to process the molecule in a supervised way.

Among concepts in Table 2, it is possible to select concepts whose extent contains only molecules of the same class, e.g. C_1, C_2, C_3, C_4, C_5. The sets of substructures in the intents of these concepts are considered as JEPs and they describe sets of molecules with the same binding mode.

It can be noticed that concept C_5 is more general than concepts C_2 and C_3 since the extent of C_5 includes a wider set of molecules and its intent includes a narrower set of substructures than in extents and intents of C_2 and C_3. As the extent of C_5 only contains molecules of the same type ("DFG-out"), it can be inferred that the substructures in the intent of C_5 characterize this binding mode in a "general and sufficient" way. Accordingly, we are interested in the most general concepts able to describe the binding modes. Here, we obtain concepts C_1, C_4, C_5, and their intents correspond to the most general JEPs.

Since every most general JEP is likely a characteristic of a binding mode, it is worth including these JEPs into molecule descriptions for any clustering or classification purposes. Molecules of the running example can be clustered as shown in Figure 3. This figure should be read as follows: molecules `319` and `320` are close to each other, and are forming a cluster. This cluster is close to molecule `ZZY` and thus molecules `319`, `320`, and `ZZY` are forming a cluster at the next level. Finally, the four molecules are agglomerated into one larger cluster. This clustering process shows the "proximity" of each molecule w.r.t. the binding mode. In this way, clustering can be used to predict the binding mode of an unknown molecule.

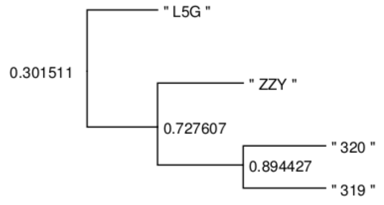


Fig. 3: The clustering result for the context on Table 1a.

3 The Classification Flow

A typical supervised classification task involves a database divided into a training set and a test set. The training set and the test set are sets of objects with their descriptions, where every object of the training set is labeled with a given class. Then a supervised classification method searches for rules in the training set, which can classify objects of the test set.

In our case, the database consists of public and known inhibitors of the c-Met protein. Here we consider 100 molecules, 30 in the training set and 70 in the test set (some molecules are shown in Figure 5). As indicated in the introduction, inhibitors can interact with the c-Met protein w.r.t. three different binding modes, plus one hypothetical binding mode under study [1]. Thus, in this work, four binding modes were used for labeling molecules in the training set. The objective is then to predict the binding modes of the molecules lying in the test set.

Figure 4 depicts the global classification flow. The first step is to choose the way how a molecule should be described. One way is to take into account

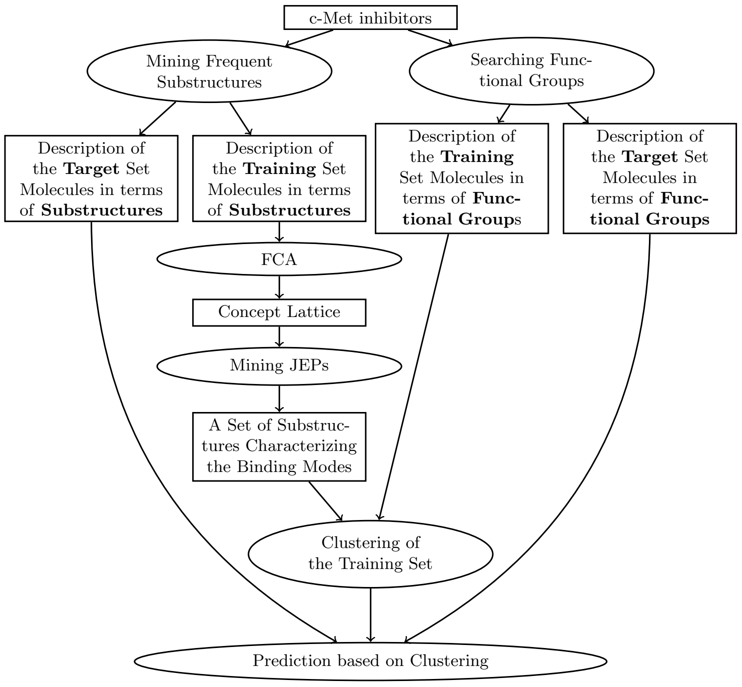

c-Met inhibitors
Mining Frequent Substructures
Searching Functional Groups
Description of the Target Set Molecules in terms of Substructures
Description of the Training Set Molecules in terms of Substructures
Description of the Training Set Molecules in terms of Functional Groups
Description of the Target Set Molecules in terms of Functional Groups
FCA
Concept Lattice
Mining JEPs
A Set of Substructures Characterizing the Binding Modes
Clustering of the Training Set
Prediction based on Clustering

Fig. 4: Diagram of the Classification Flow.

domain knowledge and to consider a molecule as a set of functional groups that are involved into interactions. But some other substructures are also involved into interactions, which are detected as follows:

1. Molecules from a dataset are considered as graphs, where vertexes correspond to atoms and edges to bonds between atoms.
2. A graph miming method is used to find all frequent subgraphs, i.e. subgraphs that belong to a significant part of molecules in the dataset.
3. A formal context is built in the following way:
   - Molecules are considered as objects.
   - Extracted substructures are considered as attributes.
   - A molecule m and a substructure s are related iff the molecule m includes s as a substructure.
4. JEPs (the sets of attributes that characterize only objects of the same class) are extracted from the formal context.

In the supervised classification task, the extracted substructures are used with functional groups to cluster molecules and to predict the binding mode of molecules in the test set.

4 Jumping Emerging Pattens (JEPs)

JEPs were introduced as a means for classification in itemset mining [12, 13], but the underlying idea had appeared and had been studied much earlier, e.g., within the framework of disjunctive version spaces [14, 15] or JSM-hypotheses. Consider an "augmented context", i.e. a context (G, M, I) taken with an additional "class attribute" giving "class information", i.e. the class of each object in G. For a concept (A, B) the set of attributes B is a JEP if every object in A is of the same class. In Table 1, the set of attributes {`F`, `O=`} is a JEP because objects `319` and `320` including these attributes are of the same class "DFG-out".

Since a JEP characterizes a class of objects, it can be used for analyzing this class and for guiding a clustering method. Usually, the set of attributes associated with a single object is trivially a JEP, but there are especially interesting JEPs characterizing a class of objects. The set of JEPs can be partially ordered w.r.t. the subset relation: if there are two JEPs J_1 and J_2 such that J_1 is a subset of J_2, then J_1 is more general, since it describes all the objects described by J_2 and some other objects. For example, the JEP J_1 ={`H`, `CAD`, `F`, `O=`} is more general than the JEP J_2 ={`H`, `CAD`, `P`, `F`, `O=`} since J_2 describes object `320`while J_1 describes objects `320`and `319`.

Relying on the JEP definition, the intent of a formal concept is a JEP if all objects in the concept extent are in the same class. Thus it is possible to compute the set of concepts for a given context and then to extract the JEPs by checking the class of objects in the concept extents. Moreover, the most general JEPs can be selected for further analysis and for clustering.

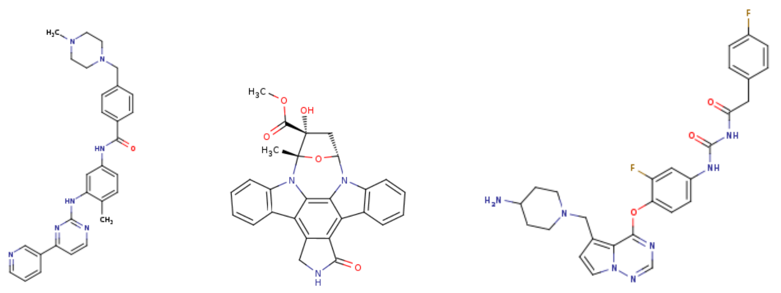
5 Graph Mining

A molecule is a complex structure composed of atoms connected by bonds, that can be considered as a graph. Vertexes of the molecule graph correspond to the atoms of the molecule and are labeled with atom names. The edges of the molecule graph are labeled with types of bonds between the corresponding atoms. For applying FCA and for finding a set of JEPs, a molecular graph can be described as as a set of subgraphs. Then, a formal context can be built with G as a set of molecules, M as a set of subgraphs or substructures and I the relation meaning that a molecule g has a substructure m. The problem now is to find "valid" and "interesting" substructures.

One way to select valid and interesting substructures is to search for frequent subgraphs –that often appear in molecular graphs– using graph mining. For a set of graphs G and a frequency threshold F_{min}, a graph s is frequent iff s is a subgraph of at least F_{min} graphs from G, i.e. $|\{g \in G \mid s \subseteq g\}| \geq F_{min}$.

For example, considering the set of molecular graphs G in Figure 5 and $F_{min} = 3$, the subgraphs "N-H" and "O=C" are frequent as they occur in all molecular graphs while subgraph "C-OH" only occurring in graph (b) (Figure 5b) and subgraph "F-C" only occurring in graph (c) (Figure 5c) are not frequent.

For discovering frequent subgraphs, different graph mining algorithms may be applied [2, 3]. Here we used `gSpan` and set $F_{min} = 10$ for the dataset of 100



(a) Imatinib or Gleevec®   (b) K-252a   (c) CKK

Fig. 5: Examples of molecules from database.

molecular graphs. This frequency threshold is sufficiently low to have a set of specific subgraphs characterizing every molecule, and it is sufficiently high to obtain feasible processing time.

The set of mined subgraphs can be divided into groups, where a group consist of a set of subgraphs appearing in the same set of molecular graphs. Thus, the group forms an equivalence class and can be represented by only one subgraph. Furthermore, the largest subgraphs preserve the sufficient information on substructures related to binding modes. In the present experiment, around 10^6 frequent subgraphs were extracted, then divided into 10^4 groups.

It can be noticed that if there are two frequent subgraphs g_1 and g_2 such that $g_1 \subseteq g_2$ then every closed JEP containing the subgraph g_2 contains the subgraph g_1. Thus, if a JEP contains g_2, there is no need to consider g_1.

6 Hierarchical Agglomerative Clustering (HAC)

Here we describe a hierarchical agglomerative clustering process (see [16]) based on the extracted JEPs and background knowledge on functional groups. Molecules are described by vectors having 55 components, including 42 functional groups[3] and 13 JEPs. The 13 JEPs are selected as the most representative for the molecules in the training set. Each attribute of the vector therefore corresponds either to a chemical functional group or to a substructures of a JEP with value set to 1 when this chemical function/substructure is present and *null* otherwise. The choice of a proper similarity is crucial for ensuring the quality of the clustering. Here, the cosine similarity was chosen according to the results of several specialized studies [17, 18]. If m_1 and m_2 are the description vectors of two molecules, then $((\boldsymbol{m_1}, \boldsymbol{m_2})$ denotes the scalar product of two vectors):

[3] The functional groups were extracted thanks to the specialized algorithm "Checkmol" http://merian.pch.univie.ac.at/ nhaider/cheminf/cmmm.html.

$$sim_{cos}(\boldsymbol{m_1}, \boldsymbol{m_2}) = \frac{(\boldsymbol{m_1}, \boldsymbol{m_2})}{|\boldsymbol{m_1}| \cdot |\boldsymbol{m_2}|} \tag{1}$$

The "centroid" of a cluster of molecules C, denoted by $\boldsymbol{centr}(C)$, is calculated as follows:

$$\boldsymbol{centr}(C) = \frac{1}{|C|} \sum_{m_i \in C} \boldsymbol{m_i} \tag{2}$$

Similarity between two clusters or between a molecule and a cluster is calculated with the same formula (1) by substituting the cluster C with its centroid $\boldsymbol{centr}(C)$.

The HAC clustering is a bottom-up process working as follows. For every molecule a unique cluster is created. Actually, all these clusters will be progressively merged until only one unique cluster remains. Considering at some step the set of remaining clusters $\mathfrak{C} = \{C_1, C_2, .., C_k\}$, a new cluster C_{k+1} is created by merging the two clusters C_i and C_j maximizing the similarity measure between them. The new cluster is added to the set of clusters while C_i and C_j are deleted from $\mathfrak{C}$. Finally, the process stops when only one cluster remains, $|\mathfrak{C}| = 1$.

$$(C_i, C_j) = \underset{C_i, C_j \in \mathfrak{C}, C_i \neq C_j}{argmax} \; sim_{cos}(\boldsymbol{centr}(C_i), \boldsymbol{centr}(C_j)) \tag{3}$$

$$C_{k+1} := C_i \cup C_j \tag{4}$$

$$\mathfrak{C} := \mathfrak{C} \cup \{C_{k+1}\} \setminus \{C_i, C_j\} \tag{5}$$

The result of HAC is shown on a dendrogram (see Figures 3 and 6). Each "vertex" of the dendrogram corresponds to a merging step of the algorithm. The number attached to the vertex represents the similarity between the two clusters at the lower level. The correlation between chemical similarities and binding modes is discussed below.

7 Results and Discussion

After applying graph mining on the set of molecules, a formal context including 30 objects (molecules) and 10^4 attributes (substructures) was built. The cardinalitiy of the sets of most general JEPs for the different binding modes are distributed as follows:

- 35 JEPs for Type-1 binding mode;
- 1 JEP for DFG-out binding mode;
- 1 JEP for C-Helix-out binding mode;
- 3 JEPs for Type-1bis binding mode.

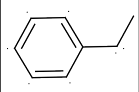
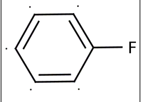
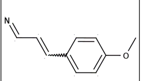
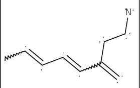
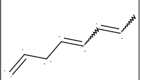
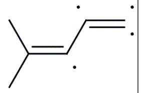
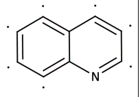
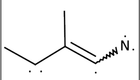
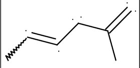
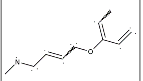
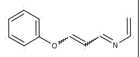
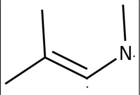
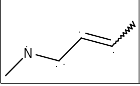
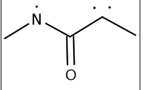
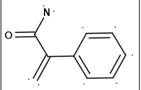
Type-1	DFG-out	C-Helix-out	Type-1bis
			
			
			
			

Table 3: Examples of the result JEPs. Every column corresponds to one JEP. Only some of structures for every JEP were exemplified. According to the dataset, all the molecules including all the substructures of the second (for example) column are of DFG-out binding mode. These sets of substructures belongs to disjoint sets of molecules.

Examples of extracted JEPs for different binding modes are shown in Table 3. Substructures associated with the most general JEPs were used in the description of a molecule for the clustering. A molecule was described by a set of functional groups and by the set of JEPs extracted by the mining process. The resulting dendrogram is shown in Figure 6. Two small clusters (0.485071 and 649934) are covering Type-1 molecules, while one small cluster (0.673565) is covering DFG-out molecules and a quite large cluster (0.681201) is covering DFG-out molecules as well as C-Helix-out molecules and one Type-1 molecule. The C-Helix-out molecules may appear in that cluster since they are quite similar while this Type-1 molecule share some chemical properties with the DFG-out molecules. It should be noticed that the dendrogram shows a better cohesion for molecules of the same binding mode and a better separation between molecules of different binding modes, than the dendrogram built from molecules only described by functional groups, and this is mainly due to JEPs substructures,

An extended dendrogram will be tested for the set of 100 molecules to check whether the class of some unknown molecule can be determined with respect to its "proximity" to other molecules in the dendrogram.

8 Conclusion

In this paper, we have shown how to classify a set of molecules with respect to their structure. Actually, we combined two classification aspects: supervised and unsupervised classifications. First, molecules are represented by molecular graphs and graph mining is applied to these graphs for extracting interesting substructures. Then, FCA is applied on an "augmented" context where there

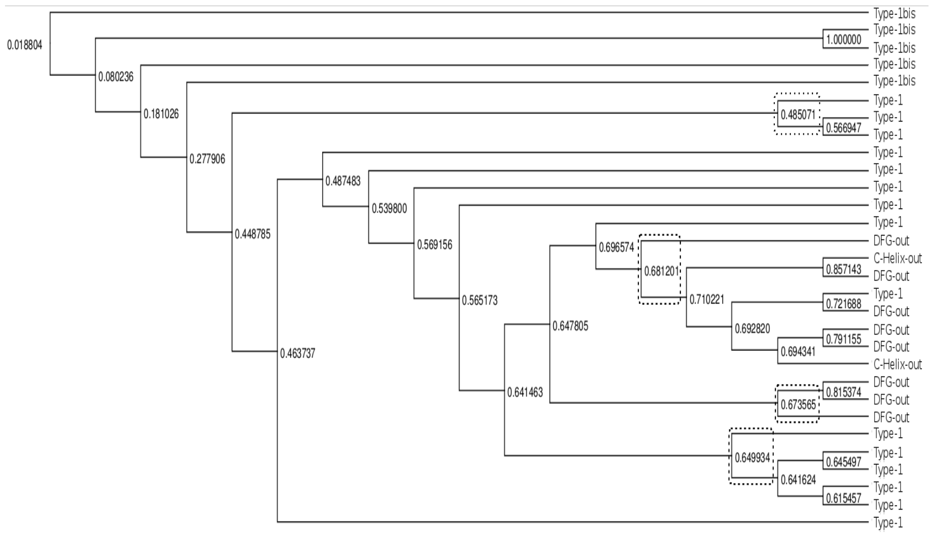


Fig. 6: The clustering result for 30 known molecules, described by functional groups and JEPs.

is a "class" information for objects. This allows one to extract JEPs. Then, JEPs and functional groups are used to cluster the molecules and to obtain a dendrogram showing the proximity between molecules.

In future work, we are going to use the dendrogram for prediction with the supervision of chemists. There are several ways to do that. One way to classify an unknown molecule is to find the closest cluster of molecules sharing the same label. Another way is to cluster all the molecules (complete set of inhibitors) and then to predict the class of unknown molecules, using information in the common class of molecules in the same cluster.

Another direction of the future work is to study and apply a knowledge-based evaluation function in graph mining, for selecting the relevant subgraphs from the chemical point of view instead of frequent ones. Finally, following [19], we are planning to combine a similarity-based approach with FCA, to avoid the clustering step and to build a lattice including all interesting information.

References

1. Asses, Y., Leroux, V., Tairi-Kellou, S., Dono, R., Maina, F., Maigret, B.: Analysis of c-met kinase domain complexes: A new specific catalytic site receptor model for defining binding modes of ATP-Competitive ligands. Chemical Biology & Drug Design **74**(6) (2009) 560–570
2. Yan, X., Han, J.: gSpan: graph-based substructure pattern mining. In: Proceedings of ICDM'02. (2002) 721 – 724
3. Nijssen, S., Kok, J.: The gaston tool for frequent subgraph mining. Electronic Notes in Theoretical Computer Science **127** (March 2005) 77–87

4. Ganter, B., Kuznetsov, S.: Formalizing hypotheses with concepts. In Ganter, B., Mineau, G., eds.: Conceptual Structures: Logical, Linguistic, and Computational Issues. Volume 1867 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2000) 342–356
5. Blinova, V.G., Dobrynin, D.A., Finn, V.K., Kuznetsov, S.O., Pankratova, E.S.: Toxicology analysis by means of the JSM-method. Bioinformatics **19**(10) (2003) 1201–1207
6. Kuznetsov, S.O., Samokhin, M.V.: Learning closed sets of labeled graphs for chemical applications. In: Proceedings of ILP. (2005) 190–208
7. Ganter, B., Kuznetsov, S.: Hypotheses and version spaces. In Ganter, B., de Moor, A., Lex, W., eds.: Conceptual Structures for Knowledge Creation and Communication. Volume 2746 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2003) 83–95
8. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. 1st edn. Springer-Verlag New York, Inc., Secaucus, NJ, USA (1997)
9. Ganter, B.: Two basic algorithms in concept analysis. In Kwuida, L., Sertkaya, B., eds.: Formal Concept Analysis. Volume 5986 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (1984) 312–340
10. Kuznetsov, S.O.: A fast algorithm for computing all intersections of objects in a finite semi-lattice. Automatic documentation and Mathematical linguistics **27**(5) (1993) 11–21
11. Merwe, D., Obiedkov, S., Kourie, D.: AddIntent: a new incremental algorithm for constructing concept lattices. In Goos, G., Hartmanis, J., Leeuwen, J., Eklund, P., eds.: Concept Lattices. Volume 2961. Springer Berlin / Heidelberg (2004) 372–385
12. Dong, G., Li, J.: Efficient mining of emerging patterns: discovering trends and differences. In: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '99, New York, ACM (1999) 43–52
13. Poezevara, G., Cuissart, B., Crémilleux, B.: Extracting and summarizing the frequent emerging graph patterns from a dataset of graphs. Journal of Intelligent Information Systems **37** (July 2011) 333–353
14. Sebag, M.: Delaying the choice of bias: A disjunctive version space approach. In: Proceedings of the 13 th International Conference on Machine Learning, Morgan Kaufmann (1996) 444–452
15. Nikolaev, N.I., Smirnov, E.N.: Stochastically guided disjunctive version space learning. In: Proceedings of the 12th European Conference on Artificial Intelligence, John Wiley & Sons, Ltd. (1996)
16. Murtagh, F.: A survey of recent advances in hierarchical clustering algorithms. The Computer Journal **26**(4) (November 1983) 354–359
17. Qian, G., Sural, S., Gu, Y., Pramanik, S.: Similarity between euclidean and cosine angle distance for nearest neighbor queries. In: Proceedings of 2004 ACM Symposium on Applied Computing, ACM Press (2004) 1232–1237
18. Yamagishi, M., Martins, N., Neshich, G., Cai, W., Shao, X., Beautrait, A., Maigret, B.: A fast surface-matching procedure for protein–ligand docking. Journal of Molecular Modeling **12**(6) (2006) 965–972
19. Kaytoue, M., Assaghir, Z., Napoli, A., Kuznetsov, S.O.: Embedding tolerance relations in formal concept analysis: an application in information fusion. In: Proceedings of the 19th ACM international conference on Information and knowledge management. CIKM '10, New York, NY, USA, ACM (2010) 1689–1692

The dependence graph of a lattice

Karell Bertet

L3I - Université de La Rochelle - av Michel Crépeau - 17042 La Rochelle
kbertet@univ-lr.fr

Abstract: In this paper, we introduce the dependence graph of a lattice defined on the set of its join-irreducible elements. This graph, issued from the dependence relation on a lattice, is a nice structure encoding together the minimal generators and the canonical direct basis of a lattice. Then, we propose a new generation algorithm.

Keywords: lattice, closed set lattice, dependence graph, implicational system, closure system, canonical direct basis, minimal generators

1 Introduction

From the year 2000, the increasing interest into Formal Concept Analysis (FCA)[7] in various domains of computer science, such as data-mining and knowledge representation, as well as the fields of ontology or databases, has brought to light the structure of *concept lattice*. A concept lattice can be introduced as a directed acyclic graph with the lattice property, defined from data described by a *binary table* object x attribute, named a *context*. The nodes of the graph are concepts - a concept is a maximal subset of objects possessing common attributes. This lattice composed of concepts connected by a generalization- specialization relation, supplies a very intuitive representation of the data.

FCA's increasing importance has many reasons. For one, new scientific areas have recently begun to incorporate computer technologies on a large scale through data-bases, whence a large production of data and the need for handling them. Secondly, the increasing power of computers permits the automatization of tasks that might have, in the worst case, exponential time-space costs. Among them, the typical problems from FCA related to the representation of a lattice that could have exponential size relative to the size of the original data.

In data mining, the problem of classification is naturally related to the notion of concept from FCA. Unsupervised classification consists in grouping objects that have close attributes while separating those having distant attributes; supervised classification groups objects having a same label (called class), while it distinguishes those having different labels. The notion of concept is then repeatedly used in applications to classify data, in a supervised or unsupervised way. Dependencies between attributes are classically represented by rules. Associative rules, well-known in data mining, can be either exact or approximate rules. For relational data-bases, systems of rules are known under the name of functional dependencies. The combinatorial explosion of the number of rules and the growing volume of data to be handled have made the use of concise representations, called bases, necessary.

In *lattice theory*, a fundamental representation theorem establishes that every lattice is the concept lattice of its *binary table* [1] defined from irreducible elements of the lattice - particular elements that are not a join or a meet of other elements. A second and less known representation theorem states that every finite lattice is isomorphic to a lattice of closed sets, where the closure operator can be defined by a basis of rules defined on the join-irreducible elements of the lattice.

There can be many equivalent bases, where two bases are equivalent if they give rise to the same closed set lattice. The canonical direct basis is the only one that is direct, meaning that the closure of an arbitrary set can be computed by just one application of the rules to the set, and that, at the same time, is minimal among the direct systems. Moreover, it has been shown in [2] that this basis is equivalent to five other bases whose rules are called *minimal functional dependency* in the domains of relational databases [10], and *proper implications* in the data-mining area research [13] whose premises are minimal generators of the lattice.

In this paper, we introduce the *dependence graph* as a representation of a lattice defined on the set of its join-irreducible elements. This graph, issued from the *dependence relation* on a lattice [1], is a nice structure encoding together the minimal generators and the canonical direct basis of a lattice. Representation of a lattice in the form of an edge-labeled graph was first suggested in [11]. This OD-Graph is closely associated to the *D-relation* on the set of join-irreducibles of a lattice, subset of the dependence relation, that was crucial in the study of free and lower bounded lattices [6].

After a first section of definitions, we define the dependence graph of a lattice. In the last section, we discuss about existing generation algorithms of the dependence graph, and we propose a new generation algorithm.

2 Definitions

Lattice. In lattice theory, the structure of lattice can been introduced either as an algebraic structure provided with two operators named lower and upper bounds, or as an ordered structure defined by the existence of particular elements called upper and lower bounds [3].

More formaly, a lattice is an order relation $\leq$ on a set S (i.e. a reflexiv, antisymmetic and transitiv relation) where every couple of elements has a a join and a meet. The *meet* (resp. *join*) of x and y, denoted $x \wedge y$ (resp. $x \vee y$), is the unique greatest lower bound (resp. least upper bound) of x and y.

Meet and join elements are defined in a more general but identical manner for a subset $X \subseteq S$: the meet of X, noted $\vee X$, is the unique greatest element of the predecessors of X, while the join of X, noted $\wedge X$, is the unique least element of the successors of X. As a direct consequence, any lattice admits a unique maximal element called *top* and denoted $\top$ or 1, and a unique minimal element called *bottom* and denoted $\perp$ or 0.

The *strict order relation* of any order relation $\leq$, denoted $<$, is an antisymmetric, transitive and irreflexive relation defined by $x < y$ if $x \leq y$ and $x \neq y$. It corresponds to the reflexive reduction of $\leq$. The *cover relation* of $\leq$, denoted $\prec$, is an antisymmetric relation defined by $x \prec y$ if $x < y$, and there is no z so that $x < z < y$. We then say that y *cover* x. It corresponds to the reflexive and transitive reduction of $\leq$. The *Hasse diagram* is a graphical representation of an order where only the arcs of the cover relation $\prec$ are represented since reflexivity and transitivity edges can be deduced.

Irreducibles elements. An element of a lattice is called *reducible* if it corresponds to a meet and a join of two distinct elements. Otherwise, it is called *irreducible*. More precisely, an element j is called a *join-irreducible* if for any subset X of elements, $j = \vee X$ implies that $j \in X$. An element m is called *meet-irreducible* if for any subset X of elements, $m = \wedge X$ implies that $m \in X$. The set of join-irreducibles of a lattice L is usually denoted J_L, and the set of meet-irreducibles M_L. In particular, we have $\bot = \vee\emptyset$ and $\top = \wedge\emptyset$ implying that $\bot$ is not a join-irreducible, and $\top$ is not a meet-irreducible.

A nice characterization establishes that an element is a join-irreducible if, and only if, it covers only one element, denoted j^-, while an element is a meet-irreducible if, and only if, it is covered by only one element, denoted m^+.

Any element $x \in S$ of a lattice L is the join of its predecessors, and the meet of its successors. The latticial property implies a reduction to join-irreducible predecessors and meet-irreducible successors:

$$x = \vee J_x = \vee\{y \in J_L \ : \ y \leq x\} \tag{1}$$

$$x = \wedge M_x = \wedge\{y \in M_L \ : \ y \geq x\} \tag{2}$$

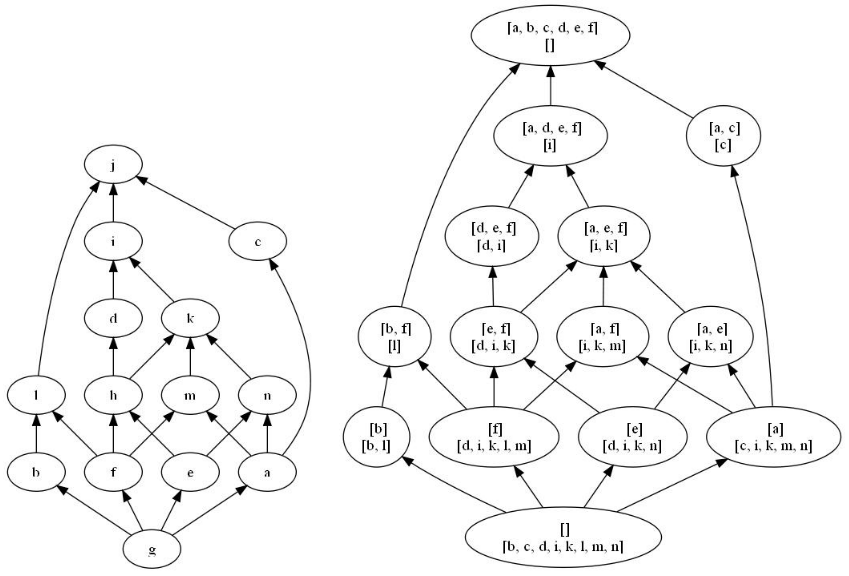
Therefore, irreducible elements are enough to build the lattice in its entirety, using either join irreducibles for reconstruction by upper bound, or meet-irreducibles for reconstruction by lower bound. Moreover, J_L and M_L are minimal set allowing reconstruction.

Minimal generators. Consider one element $x \in S$. Although x is the join of J_x, J_x is not always the minimal subset to define x as a join. A minimal subset to obtain x as a join, including in J_x, is named a *basis*, a *minimal generating set* or a *minimal generator* for x. More formally, a *minimal generator* of x is a subset B of J_x such that $x = \vee B$ and B is inclusion-minimal, i.e for all $A \subset B$, then $x \neq \vee A$. The family $\mathcal{B}_x$ of minimal generators of x is then:

$$\mathcal{B}_x = \{B \subseteq J_x \ : \ x = \vee B \text{ and } x \neq \vee A \text{ for all } A \subset B\} \tag{3}$$

The dual observation for M_x is valid for a reconstruction of x as meet. The number of minimal generators of x can be exponential in the cardinality of J_x in the worst case.

Consider the example of lattice in Figure 1(a). Six elements possess a single incoming arc, forming all join-irreducibles ; the meet-irreducibles, characterized



(a) A lattice (b) The isomorphic lattice with the sets J_x and M_x inside each node x

Fig. 1. Example of lattice

by a single outcoming arc, are heigth:

$$J = \{a, b, c, d, e, f\} \tag{4}$$

$$M = \{b, c, d, i, k, l, m, n\} \tag{5}$$

These irreducible elements are used to describe more precisely the elements of the lattice (see Figure 1) where node of each element x contains its join-irreducible predecessors J_x and its meet-irreducible successors M_x. Minimal generators are given in Table 1. One can observe that each join-irreducible possesses itself as unique minimal generator ; the top element possesses 4 minimal generators.

x	a	b	c	d	e	f	g
J_x	a	b	ac	def	e	f	$\emptyset$
$\mathcal{B}_x$	$\{a\}$	$\{b\}$	$\{c\}$	$\{d\}$	$\{e\}$	$\{f\}$	$\{\emptyset\}$

x	h	i	j	k	l	m	n
J_x	ef	adef	abcdef	aef	bf	af	ae
$\mathcal{B}_x$	$\{ef\}$	$\{ad\}$	$\{ab, bc, bd, dc\}$	$\{aef\}$	$\{bf\}$	$\{af\}$	$\{ae\}$

Table 1. Minimal generators of the lattice in Figure 1(a)

3 Dependence graph and canonical direct basis of a lattice.

Dependence graph. The dependence graph of a lattice is defined on the set J_L of join-irreducible elements.It is an edge-labeled directed graph whose edges corresponds to the dependence relation δ of a lattice [1], and are labeled by some minimal generators, thus a size that can be exponential in the cardinality of J_L. More precisely, the dependence graph of a lattice L is a pair (δ, ω) where:

- δ is the *dependence relation* [1] defined on J_L by $j \delta j'$ if there exists $x \in S$ such that $j \not\leq x$, $j' \not\leq x$ and $j < j' \vee x$. We note $j \delta_x j'$.
- ω is a label of the edges defined on $\mathcal{P}(J_L)$, for each relation $j \delta j'$, by:

$$\omega(j, j') = \{\text{minimal generators of } x \ : \ j \delta_x j' \text{ and } x \text{ minimal in the lattice}$$

A pair $(j, j') \in \delta$ can be denoted either $j \delta_x j'$ or $j \delta_B j'$, with B minimal generator of x. Figure 2 represents the dependence graph of lattice in Figure 1(a).

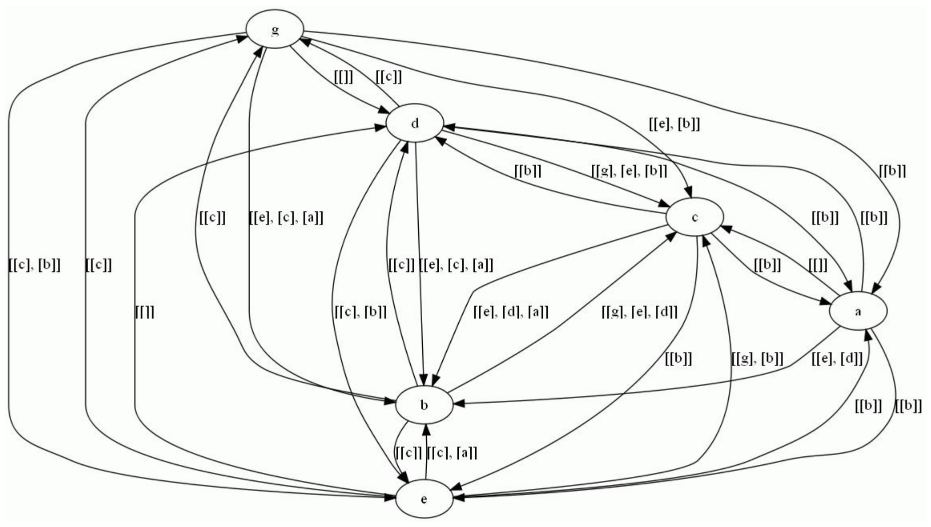


Fig. 2. Dependence graph of the lattice in Figure 1(a)

The subgraph $\delta_\emptyset$ of the dependence graph to the edges containing the empty set as label corresponds to the subgraph of the lattice induced by its join-irreducible, since $j < j'$ implies $j < j' \vee \perp$ and $\omega(j, j') = J_\perp = \{\emptyset\}$. As a direct consequence, a lattice is distributive when edge-labels of its dependence graph all are the empty set.

Canonical direct unit basis. A set of unit implication (or rules) Σ is a binary relation between $\mathcal{P}(S)$ and S where a rule (X, y), generaly denoted $X \rightarrow y$, means that X *"implies"* y, with X called the *premisse* and y the *conclusion.*

The dependence graph encodes a set of rules defined on the join-irreducibles J_L of a lattice, with a rule $B + j' \rightarrow j$ for each $(j, j') \in \delta_B$. This set of rules forms a particular basis of the lattice called the *canonical direct unit basis*, and denoted Σ_{cdb} [2].

Morover, an important result establishes that every lattice is isomorphic to the *closed set lattice* $(\mathcal{F}, \subseteq)$ of its canonical direct basis, where $\mathcal{F}$ is a family on J_L. This family contains all the *closures* of the basis where a closure is a subset X of J_L verifying all rules, i.e. for each rule $B \rightarrow y$, if $B \subseteq X$ then $y \in B$.

Therefore, the dependence graph of a lattice encodes the canonical direct basis from which the lattice reconstruction is possible as a closed set lattice.

4 Generation algorithm

Since the size of the dependence graph can be exponential in the size of the lattice, this generation problem belongs to the more general class of problems having an input of size n, and an output of size N bounded by 2^n. For this class of problems, a classical worst-case analysis makes them exponential, thus NP-hard, with an exponential space. However, a more precise information can be obtained by output-sensitive analysis techniques (see a survey in [9]). These analyzes are relevant since the recent improvements in storage and processing capacity increasingly often allow to handle some exponential data, what was not possible even some time ago. The idea is to consider the time complexity needed to generate only one element of the output (i.e. one rule or one minimal generator in our case).

The time complexity per Σ_{cdb}-rule has then to be considered. Although the most common algorithms have an exponential delay complexity, there exist some algorithms with a polynomial delay complexity.

The definition of minimal generators for an element x induces an exponential generation since any subset of J_x as to be tested. Another strategy is issued from the equivalence between minimal generators and minimal transversals of a closed set, problem known to be exponential. This strategy has been capitalized by Pfaltz's incremental algorithm [12], and by Jen's algorithm [5] used in data-mining to compute minimal generators. Jen's algorithm computes minimal generators from the *faces* of x defined by considering immediate predecessors of x in the lattice.

However, in logic area, the algorithm attributable to Ibaraki et al. ([8]) computes a Σ_{cdb}-rule - and thus the dependence graph - in polynomial time with a family $\mathcal{F}$ of closed set as input.

Algorithm 1 generates the dependence graph with the same polynomial complexity per rule or per minimal generator.

```
Name: dependenceGraph
Data   : A lattice $L = (S, \leq)$
Result : The dependence graph of the lattice
begin
    compute the set $J_L$ of join-irreducibles of the lattice;
    initialize a graph $G$ with join-irreducibles as nodes;
    compute a topological sort $T$ of the lattice;
    foreach $x \in T$ do
        foreach $(j, j') \in J \times J$ such that $x \vee j' \geq j$ do
            add the edge $(j, j')$ in $G$;
            compute the set $J_x$ of join-irreducible predecessors of $x$;
            initalize the empty family $GM_x$;
            let $G'$ subgraph of $G$ induced by $J_x$ on nodes and edges's labels;
            foreach edge $(k, k')$ of $G'$ do
                foreach valuation $B$ of the edge $(k, k')$ do
                    if $B \vee k' = x$ then add the set $B + k'$ to the family $GM_x$
                end
            end
            if $GM_x$ is empty then $GM_x = \{J_x\}$;
        end
    end
    label the edge $(j, j')$ with $GM_x$;
    return $G$;
end
```

Algorithm 1: Generation of the dependence graph of a lattice

Proposition 1. *Algorithm 1 generates the dependence graph, the minimal generators, or the canonical direct basis of a lattice $L = (S, \leq)$ in $O(|\Sigma_{cdb}||S||J_L|^3)$, i.e. in $O(|S||J_L|^3)$ per Σ_{cdb}-rule or per minimal generator.*

Proof. First, computation of the relation δ on the join-irreducibles can be done in $O(|S|^2|J_L|^2)$ by determining, for each $x \in S$, if x is a minimal element in the lattice such that $j\delta_x j'$.

Computation of edges's labels is more difficult. One can observe that minimal generators are recursively defined according to the relation $\leq$ in the lattice. Indeed, if we consider two join-irreducibles such that $j\delta_x j'$ - with x an element of the lattice - or equivalently $j\delta_B j'$ - with B minimal generator of x - then $B + j'$ is a minimal generator of $x \vee j'$, thus recursively defined from B. One can distinguish between two cases:

- When B is strictly included in J_x, then B can be deduced from a minimal generator of a predecessor of x.
- When $B = J_x$, then J_x is the only minimal generator of x.

Therefore, a travel of the lattice from the bottom to the top allows to recursively compute minimal generators in $O(|\Sigma_{cdb}||S||J_L|^3)$.

The dependence graph of a lattice, and thus its canonical direct basis and its minimal generators, can easily be generated with a binary table as input, after its concept lattice generation. In particular, Bordat's algorithm [4] generates the Hasse diagram of the concept lattice of a binary table from the botom to the top using a successor function. Therefore, another strategy would consists in computing the dependence graph along with the lattice generation.

5 Conclusion

In this paper, we introduced the dependence graph as a representation of a finite lattice encoding both its canonical direct basis and its minimal generators. We propose a new generation algorithm with an improvment complexity. This structure can be used in various domains of computer science, such as data-mining and knowledge representation. Indeed, the canonical direct basis of rules is a nice basis to represent dependencies between attributes, in a classification task for example. The use of minimal generators could gives raise to an attributs set reduction, usefull for data indexation for example.

References

1. M. Barbut and B. Monjardet. *Ordres et classifications : Algèbre et combinatoire.* Hachette, Paris, 1970. 2 tomes.
2. K. Bertet and B. Monjardet. The multiple facets of the canonical direct unit implicational basis. *Theoretical Computer Science*, 411(22-24):2155–2166, 2010.
3. G. Birkhoff. *Lattice theory.* American Mathematical Society, 1st edition, 1940.

4. J.P. Bordat. Calcul pratique du treillis de Galois d'une correspondance. *Math. Sci. Hum.*, 96:31–47, 1986.
5. A. Le Floch, C. Fisette, R. Missaoui, P. vatchev, and R. Godin. Jen: un algorithme efficace de construction de générateurs minimaux pour l'identification de règles d'association. *Nouvelles Technologies de l'Information (numéro spécial)*, 1(1):135–146, 2003.
6. R. Freeze, J. Jezek, and J.B. Nation. Free lattices. In Providence, editor, *Mathematical survey and monographs. Americal Mathematical Society*, volume 42, 1995.
7. B. Ganter and R. Wille. *Formal concept analysis, Mathematical foundations*. Springer Verlag, Berlin, 1999.
8. T. Ibaraki, A. Kogan, and K. Makino. Functional dependencies in Horn theories. *Artificial Intelligence*, 108:1–30, 1999.
9. E. Lawler, J.K. Lenstra, and A.H.G. Rinnoy kan. Generating all maximal independant sets: Np-hardness and polynomial time algorithms. *SIAM Journal on Computing*, 9:558–565, 1980.
10. D. Maier. *The Theory of Relational Databases*. Computer Sciences Press, 1983.
11. J. B. Nation. An approach to lattice varieties of finite height. *Algebra Universalis*, 27(4):521–543, 1990.
12. JL. Pfaltz and CM. Taylor. Scientific discovery through iterative transformations of concept lattices. In *Workhop on Discrete Applied Mathematics, in conjonction with the* 2^{nd} *SIAM International Conference on Data-Mining*, pages 65–74, 2002.
13. R. Taouil and Y. Bastide. Computing proper implications. In 9^{th} *International Conference on Conceptual Structures*, Stanford, USA, 2002.

Linking L-Chu correspondences and completely lattice L-ordered sets

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

[1] University of Pavol Jozef Šafárik, Košice, Slovakia*
[2] Dept. Matemática Aplicada, Univ. Málaga, Spain**

Abstract. Continuing our categorical study of L-fuzzy extensions of formal concept analysis, we provide a representation theorem for the category of L-Chu correspondences between L-formal contexts and prove that it is equivalent to the category of completely lattice L-ordered sets.

1 Introduction

This paper deals with an extremely general form of Formal Concept Analysis (FCA) based on categorical constructs and L-fuzzy sets. FCA has become an extremely useful theoretical and practical tool for formally describing structural and hierarchical properties of data with "object-attribute" character, and this applicability justifies the need of a deeper knowledge of its underlying mechanisms: and one important way to obtain this extra knowledge turns out to be via generalization and abstraction.

Several approaches have been presented for generalizing the framework and the scope of formal concept analysis and, nowadays, one can see works which extend the theory by using ideas from fuzzy set theory, rough set theory, or possibility theory [1, 10, 18, 20–22, 24].

Concerning applications of fuzzy formal concept analysis, one can see papers ranging from ontology merging [9], to applications to the Semantic Web by using the notion of concept similarity or rough sets [11, 12], and from noise control in document classification [19] to the development of recommender systems [7].

We are concerned in this work with the category L-ChuCors, built on top of several fuzzy extensions of the classical concept lattice, mainly introduced by Bělohlávek [3, 5, 6], who extended the underlying interpretation on classical logic to the more general framework of L-fuzzy logic [13].

The categorical treatment of morphisms as fundamental structural properties has been advocated by [17] as a means for the modelling of data translation, communication, and distributed computing, among other applications. Our approach broadly continues the research line which links the theory of Chu spaces with concept lattices [25] but, particularly, is based on the notion of Chu correspondences between formal contexts developed by Mori in [23]. Previous work

* Partially supported by grant VEGA 1/0832/12 and APVV-0035-10.
** Partially supported by Spanish Ministry of Science and FEDER funds through project TIN09-14562-C05-01 and Junta de AndalucÃa project P09-FQM-5233.

in this categorical approach has been developed by the authors in [14, 16]. The category L-ChuCors is formed by considering the class of L-contexts as objects and the L-fuzzy Chu correspondences as arrows between objects. Recently, the authors developed a further abstraction [15] aiming at formally describing structural properties of intercontextual relationships of L-contexts.

The main result in this work is a constructive proof of the equivalence between the categories of L-formal contexts and L-Chu correspondences and that of completely lattice L-ordered sets and their corresponding morphisms. In order to obtain a reasonably self-contained document, Section 2 introduces the basic definitions concerning the L-fuzzy extension of formal concept analysis, as well as those concerning L-Chu correspondences; then, the categories associated to L-formal contexts and L-CLLOS are defined in Section 3 and, finally, the proof of equivalence is in Section 4.

2 Preliminaries

2.1 Basics of *L*-fuzzy FCA

Definition 1. *An algebra $\langle L, \wedge, \vee, \otimes, \rightarrow, 0, 1\rangle$ is said to be a* ***complete residuated lattice*** *if*

- *$\langle L, \wedge, \vee, 0, 1\rangle$ is a complete lattice with the least element 0 and the greatest element 1,*
- *$\langle L, \otimes, 1\rangle$ is a commutative monoid,*
- *$\otimes$ and $\rightarrow$ are adjoint, i.e. $a \otimes b \leq c$ if and only if $a \leq b \rightarrow c$, for all $a, b, c \in L$, where $\leq$ is the ordering in the lattice generated from $\wedge$ and $\vee$.*

Definition 2. *Let L be a complete residuated lattice, an* ***L-fuzzy context*** *is a triple $\langle B, A, r\rangle$ consisting of a set of objects B, a set of attributes A and an L-fuzzy binary relation r, i.e. a mapping $r\colon B \times A \rightarrow L$, which can be alternatively understood as an L-fuzzy subset of $B \times A$.*

Definition 3. *Consider an L-fuzzy context $\langle B, A, r\rangle$. Mappings $\uparrow\colon L^B \rightarrow L^A$ and $\downarrow\colon L^A \rightarrow L^B$ can be defined for every $f \in L^B$ and $g \in L^A$ as follows:*

$$\uparrow(f)(a) = \bigwedge_{o \in B} \big(f(o) \rightarrow r(o, a)\big) \qquad\qquad \downarrow(g)(o) = \bigwedge_{a \in A} \big(g(a) \rightarrow r(o, a)\big)$$

Definition 4. *An* ***L-fuzzy concept*** *is a pair $\langle f, g\rangle$ such that $\uparrow(f) = g$ and $\downarrow(g) = f$. The first component f is said to be the* ***extent*** *of the concept, whereas the second component g is the* ***intent*** *of the concept.*

The set of all L-fuzzy concepts associated to a fuzzy context $\langle B, A, r\rangle$ will be denoted as L-FCL(B, A, r).

An ordering between L-fuzzy concepts is defined as follows: $\langle f_1, g_1\rangle \leq \langle f_2, g_2\rangle$ if and only if $f_1 \subseteq f_2$ ($f_1(o) \leq f_2(o)$ for all $o \in B$) if and only if $g_1 \supseteq g_2$ ($g_1(o) \geq g_2(o)$ for all $a \in A$).

Theorem 1 (See [5]). *The poset* $(L\text{-FCL}(B, A, r), \leq)$ *is a complete lattice where*

$$\bigwedge_{j\in J} \langle f_j, g_j\rangle = \Big\langle \bigwedge_{j\in J} f_j, \uparrow(\bigwedge_{j\in J} f_j)\Big\rangle$$
$$\bigvee_{j\in J} \langle f_j, g_j\rangle = \Big\langle \downarrow(\bigwedge_{j\in J} g_j), \bigwedge_{j\in J} g_j\Big\rangle$$

Moreover a complete lattice $\mathcal{V} = \langle V, \leq\rangle$ *is isomorphic to* $L\text{-FCL}(B, A, r)$ *iff there are mappings* $\gamma : B \times L \to V$ *and* $\mu : A \times L \to V$*, such that* $\gamma(B \times L)$ *is* $\bigvee$*-dense and* $\mu A \times L$ *is* $\bigwedge$*-dense in* $\mathcal{V}$*, and* $(k \otimes l) \leq r(o, a)$ *is equivalent to* $\gamma(o, k) \leq \mu(a, l)$ *for all* $o \in B$*,* $a \in A$ *and* $k, l \in L$*.*

Bělohlávek has extended the fundamental theorem of concept lattices by Dedekind-MacNeille completion in fuzzy settings by using the notions of L-equality and L-ordering. All the definitions and related constructions given until the end of the section are from [6].

Definition 5. *A binary* L*-relation* $\approx$ *on* X *is called an* ***L-equality*** *if it satisfies*

1. $(x \approx x) = 1$*, (reflexivity),*
2. $(x \approx y) = (y \approx x)$*, (symmetry),*
3. $(x \approx y) \otimes (y \approx z) \leq (x \approx z)$*, (transitivity),*
4. $(x \approx y) = 1$ *implies* $x = y$

L-equality is a natural generalization of the classical (bivalent) notion.

Definition 6. *An* ***L-ordering*** *(or fuzzy ordering) on a set* X *endowed with an* L*-equality relation* $\approx$ *is a binary* L*-relation* $\preceq$ *which is compatible w.r.t.* $\approx$ *(i.e.* $f(x) \otimes (x \approx y) \leq f(y)$*, for all* $x, y \in X$*) and satisfies*

1. $x \preceq x = 1$*, (reflexivity),*
2. $(x \preceq y) \wedge (y \preceq x) \leq (x \approx y)$*, (antisymmetry),*
3. $(x \preceq y) \otimes (y \preceq z) \leq (x \preceq z)$*, (transitivity).*

If $\preceq$ *is an* L*-order on a set* X *with an* L*-equality* $\approx$*, we call the pair* $\langle\langle X, \approx\rangle \preceq\rangle$ *an* L*-ordered set.*

Clearly, if $L = 2$, the notion of L-order coincides with the usual notion of (partial) order.

Definition 7. *An* L*-set* $f \in L^X$ *is said to be an* ***L-singleton*** *in* $\langle X, \approx\rangle$ *if it is compatible w.r.t.* $\approx$ *and the following holds:*

1. *there exists* $x_0 \in X$ *with* $f(x_0) = 1$
2. $f(x) \otimes f(y) \leq (x \approx y)$*, for all* $x, y \in X$*.*

Definition 8. *For an* L*-ordered set* $\langle\langle X, \approx\rangle \preceq\rangle$ *and* $f \in L^X$ *we define the* L*-sets* $\inf(f)$ *and* $\sup(f)$ *in* X *by*

1. $\inf(f)(x) = (\mathcal{L}(f))(x) \wedge (\mathcal{UL}(f))(x)$
2. $\sup(f)(x) = (\mathcal{U}(f))(x) \wedge (\mathcal{LU}(f))(x)$

where

- $\mathcal{L}(f)(x) = \bigwedge_{y \in X} \big(f(y) \to (x \preceq y)\big)$
- $\mathcal{U}(f)(x) = \bigwedge_{y \in X} \big(f(y) \to (y \preceq x)\big)$

$\inf(f)$ *and* $\sup(f)$ *are called* infimum *or* supremum*, respectively.*

Definition 9. *An L-ordered set $\langle\langle X, \approx\rangle \preceq\rangle$ is said to be* ***completely lattice L-ordered set*** *if for any $f \in L^X$ both $\sup(f)$ and $\inf(f)$ are $\approx$-singletons.*

By proving of all the following lemmas some of the properties of residuated lattices are used. All details could be found in [4]. Some of needed properties are listed below.

1. $(k \to (l \to m)) = ((k \otimes l) \to m) = ((l \otimes k) \to m) = (l \to (k \to m))$
2. $k \to \bigwedge_{i \in I} m_i = \bigwedge_{i \in I}(k \to m_i)$
3. $(\bigvee_{i \in I} m_i) \to k = \bigwedge_{i \in I}(m_i \to k)$

Lemma 1. *For any pair of L-concepts $\langle f_i, g_i\rangle \in L\text{-FCL}(B, A, r)$ $(i \in \{1, 2\})$ of any L-context $\langle B, A, r\rangle$ the following equality holds.*

$$\bigwedge_{o \in B} \big(f_1(o) \to f_2(o)\big) = \bigwedge_{a \in A} \big(g_2(a) \to g_1(a)\big)$$

Proof.

$$\begin{aligned}
\bigwedge_{o \in B} \big(f_1(o) \to f_2(o)\big) &= \bigwedge_{o \in B} \big(f_1(o) \to \downarrow(g_2)(o)\big) \\
&= \bigwedge_{o \in B} \Big(f_1(o) \to \bigwedge_{a \in A} \big(g_2(a) \to r(o, a)\big)\Big) \\
&= \bigwedge_{a \in A} \Big(g_2(a) \to \bigwedge_{o \in B} \big(f_1(o) \to r(o, a)\big)\Big) \\
&= \bigwedge_{a \in A} \big(g_2(a) \to \uparrow(f_1)(a)\big) \\
&= \bigwedge_{a \in A} \big(g_2(a) \to g_1(a)\big) \qquad \square
\end{aligned}$$

Definition 10. *We define an L-equality $\approx$ and L-ordering $\preceq$ on the set of formal concepts $L\text{-FCL}(C)$ of L-context C as follows:*

1. $\langle f_1, g_1\rangle \preceq \langle f_2, g_2\rangle = \bigwedge_{o \in B} \big(f_1(o) \to f_2(o)\big) = \bigwedge_{a \in A} \big(g_2(a) \to g_1(a)\big)$
2. $\langle f_1, g_1\rangle \approx \langle f_2, g_2\rangle = \bigwedge_{o \in B} \big(f_1(o) \leftrightarrow f_2(o)\big) = \bigwedge_{a \in A} \big(g_2(a) \leftrightarrow g_1(a)\big)$

where $k \leftrightarrow m$ is defined as $(k \to m) \wedge (m \to k)$ for any $k, m \in L$.

Definition 11. *Let $C = \langle B, A, r\rangle$ be an L-fuzzy formal context and γ be an L-set from $L^{L\text{-FCL}(C)}$. We define L-sets of objects and attributes $\bigcup_B \gamma$ and $\bigcup_A \gamma$, respectively, as follows:*

1. $(\bigcup_B \gamma)(o) = \displaystyle\bigvee_{\langle f,g\rangle \in L\text{-FCL}(C)} (\gamma(\langle f, g\rangle) \otimes f(o))$, *for $o \in B$*
2. $(\bigcup_A \gamma)(a) = \displaystyle\bigvee_{\langle f,g\rangle \in L\text{-FCL}(C)} (\gamma(\langle f, g\rangle) \otimes g(a))$, *for $a \in A$*

Theorem 2. *Let $C = \langle B, A, r\rangle$ be an L-context. $\langle\langle L\text{-FCL}(C), \approx\rangle, \preceq\rangle$ is a completely lattice L-ordered set in which infima and suprema can be described as follows: for an L-set $\gamma \in L^{L-\text{FCL}(C)}$ we have:*

$$^1\inf(\gamma) = \Big\{\langle \downarrow(\bigcup_A \gamma), \uparrow\downarrow(\bigcup_A \gamma)\rangle\Big\} \qquad ^1\sup(\gamma) = \Big\{\langle \downarrow\uparrow(\bigcup_B \gamma), \uparrow(\bigcup_B \gamma)\rangle\Big\}$$

Moreover a completely lattice L-ordered set $\mathcal{V} = \langle\langle V, \approx\rangle, \preceq\rangle$ is isomorphic to $\langle\langle L\text{-FCL}(\langle B, A, r\rangle), \approx_1\rangle, \preceq_1\rangle$ iff there are mappings $\gamma : B \times L \to V$ and $\mu : A \times L \to V$, such that $\gamma(B\times L)$ is $\{0, 1\}$-supremum dense and $\mu(A\times L)$ is $\{0, 1\}$-infimum dense in $\mathcal{V}$, and $((k \otimes l) \to r(o, a)) = (\gamma(o, k) \preceq \mu(a, l))$ for all $o \in B$, $a \in A$ and $k, l \in L$. In particular, $\mathcal{V}$ is isomorphic to $\langle\langle L\text{-FCL}(V, V, \preceq), \approx_1\rangle, \preceq_1\rangle$.

2.2 *L*-Chu correspondences

Definition 12. *Consider two L-fuzzy contexts $C_i = \langle B_i, A_i, r_i\rangle, (i = 1, 2)$, then the pair $\varphi = (\varphi_L, \varphi_R)$ is called a **correspondence** from C_1 to C_2 if φ_L and φ_R are L-multifunctions, respectively, from B_1 to B_2 and from A_2 to A_1 (that is, $\varphi_L : B_1 \to L^{B_2}$ and $\varphi_R : A_2 \to L^{A_1}$).*

*The L-correspondence φ is said to be a **weak L-Chu correspondence** if the equality*

$$\bigwedge_{a_1\in A_1} (\varphi_R(a_2)(a_1) \to r_1(o_1, a_1)) = \bigwedge_{o_2\in B_2} (\varphi_L(o_1)(o_2) \to r_2(o_2, a_2)) \qquad (1)$$

holds for all $o_1 \in B_1$ and $a_2 \in A_2$.

*A weak Chu correspondence φ is an **L-Chu correspondence** if $\varphi_L(o_1)$ is an L-set of objects closed in C_2 and $\varphi_R(a_2)$ is an L-set of attributes closed in C_1 for all $o_1 \in B_1$ and $a_2 \in A_2$. We will denote the set of all L-Chu correspondences from C_1 to C_2 by L-ChuCors(C_1, C_2).*

Definition 13. *Given a mapping $\varpi\colon X \to L^Y$, we define $\varpi_+\colon L^X \to L^Y$ for all $f \in L^X$ by $\varpi_+(f)(y) = \bigvee_{x\in X} \big(f(x) \otimes \varpi(x)(y)\big)$.*

3 Introducing the relevant categories

3.1 The category *L*-ChuCors

- **objects** L-fuzzy formal contexts

- **arrows** L-Chu correspondences
- **identity arrow** $\iota : C \to C$ of L-context $C = \langle B, A, r\rangle$
  - $\iota_L(o) = \downarrow\uparrow(\chi_o)$, for all $o \in B$
  - $\iota_R(a) = \uparrow\downarrow(\chi_a)$, for all $a \in A$
- **composition** $\varphi_2 \circ \varphi_1 : C_1 \to C_3$ **of arrows** $\varphi_1 : C_1 \to C_2$, $\varphi_2 : C_2 \to C_3$ ($C_i = \langle B_i, A_i, r_i\rangle$, $i \in \{1, 2\}$)
  - $(\varphi_2 \circ \varphi_1)_L : B_1 \to L^{B_3}$ defined as $(\varphi_2 \circ \varphi_1)_L(o_1) = \downarrow_3\uparrow_3 \big(\varphi_{2L+}(\varphi_{1L}(o_1))\big)$ where

$$\varphi_{2L+}(\varphi_{1L}(o_1))(o_3) = \bigvee_{o_2 \in B_2} \varphi_{1L}(o_1)(o_2) \otimes \varphi_{2L}(o_2)(o_3)$$

  - and $(\varphi_2 \circ \varphi_1)_R : A_3 \to L^{A_1}$ defined as $(\varphi_2 \circ \varphi_1)_R(a_3) = \uparrow_1\downarrow_1 \big(\varphi_{1R+}(\varphi_{2R}(a_3))\big)$ where

$$\varphi_{1R+}(\varphi_{2R}(a_3))(a_1) = \bigvee_{a_2 \in A_2} \varphi_{2R}(a_3)(a_2) \otimes \varphi_{1R}(a_2)(a_1)$$

All details about definition of the category L-ChuCors could be found in [15].

3.2 Category *L*-CLLOS

Here we define another category

Objects are completely lattice L-ordered sets (in short, L-CLLOS) i.e. our objects will be represented as $\mathcal{V} = \langle\langle V, \approx\rangle, \preceq\rangle$

Arrows are pairs of mappings between two L- CLLOSs i.e. $\langle s, z\rangle$ between $\mathcal{V}_1$ and $\mathcal{V}_2$, such that:
1. $s : V_1 \to V_2$,
2. $z : V_2 \to V_1$,
3. $(s(v_1) \preceq_2 v_2) = (v_1 \preceq_1 z(v_2))$, for all $(v_1, v_2) \in V_1 \times V_2$.

Identity arrow of $\langle\langle V, \approx\rangle, \preceq\rangle$ is a pair of identity morphisms $\langle id_V, id_V\rangle$

Composition of arrows is based on composition of mappings: consider two arrows $\langle s_i, z_i\rangle : \mathcal{V}_i \to \mathcal{V}_{i+1}$, where $i \in \{1, 2\}$. Composition is defined as follows:

$$\langle s_2, z_2\rangle \circ \langle s_1, z_1\rangle = \langle s_2 \circ s_1, z_1 \circ z_2\rangle.$$

Thus, given a pair of two arbitrary elements $(v_1, v_3) \in V_1 \times V_3$ then:

$$\begin{aligned} \big((s_2 \circ s_1)(v_1) \preceq_3 v_3\big) &= \big(s_2(s_1(v_1)) \preceq_3 v_3\big) \\ &= \big(s_1(v_1) \preceq_2 z_2(v_3)\big) \\ &= \big(v_1 \preceq_1 z_1(z_2(v_3))\big) \\ &= \big(v_1 \preceq_1 (z_1 \circ z_2)(v_3)\big) \end{aligned}$$

Associativity of composition follows trivially because of the associativity of composition of mappings between sets.

4 The categories L-ChuCors and L-CLLOS are equivalent

In this section we start to build the equivalence by introducing a functor Γ from L-ChuCors to L-CLLOS in the following way:

1. $\Gamma(C) = \langle\langle L\text{-FCL}(C), \approx\rangle, \preceq\rangle$ for any L-context C will be its L-concept L-CLLOS.
2. $\Gamma(\varphi) = \langle\varphi_\vee, \varphi_\wedge\rangle$. To any L-Chu correspondence $\varphi \in L\text{-ChuCors}(C_1, C_2)$, $\Gamma(\varphi)$ will be a pair of mappings $\langle\varphi_\vee, \varphi_\wedge\rangle$ defined as follows:
   - $\varphi_\vee\big(\langle f_1, g_1\rangle\big) = \big\langle \downarrow_2\uparrow_2 \big(\varphi_{L+}(f_1)\big), \uparrow_2 \big(\varphi_{L+}(f_1)\big)\big\rangle$
   - $\varphi_\wedge\big(\langle f_2, g_2\rangle\big) = \big\langle \downarrow_1 \big(\varphi_{R+}(g_2)\big), \uparrow_1\downarrow_1 \big(\varphi_{R+}(g_2)\big)\big\rangle$

   where $\langle f_i, g_i\rangle \in L\text{-FCL}(C_i)$ for $i \in \{1, 2\}$.

Lemma 2. *$\Gamma(\varphi) \in L\text{-CLLOS}(\Gamma(C_1), \Gamma(C_2))$ for any $\varphi \in L\text{-ChuCors}(C_1, C_2)$.*

Proof. Consider two arbitrary L-concepts $\langle f_i, g_i\rangle$ of $\langle\langle L\text{-FCL}(C_i), \approx_i\rangle, \preceq_i\rangle$ for $i \in \{1, 2\}$, such that $C_i = \langle B_i, A_i, r_i\rangle$.

$$\begin{aligned}
\varphi_\vee\big(\langle f_1, g_1\rangle\big) &\preceq_2 \langle f_2, g_2\rangle \\
&= \big\langle \downarrow_2\uparrow_2 \big(\varphi_{L+}(f_1)\big), \uparrow_2 \big(\varphi_{L+}(f_1)\big)\big\rangle \preceq_2 \langle f_2, g_2\rangle \\
&= \bigwedge_{a_2\in A_2} \big(g_2(a_2) \to \uparrow_2 \big(\varphi_{L+}(f_1)\big)(a_2)\big) \\
&= \bigwedge_{a_2\in A_2} \big(g_2(a_2) \to \bigwedge_{o_2\in B_2} \big(\bigvee_{o_1\in B_1} \big(\varphi_L(o_1)(o_2) \otimes f_1(o_1)\big) \to r_2(o_2, a_2)\big)\big) \\
&= \bigwedge_{a_2\in A_2} \bigwedge_{o_1\in B_1} \big(g_2(a_2) \to \big(f_1(o_1) \to \bigwedge_{o_2\in B_2} \big(\varphi_L(o_1)(o_2) \to r_2(o_2, a_2)\big)\big)\big) \\
&= \bigwedge_{a_2\in A_2} \bigwedge_{o_1\in B_1} \big(f_1(o_1) \to \big(g_2(a_2) \to \bigwedge_{a_1\in A_1} \big(\varphi_R(a_2)(a_1) \to r_1(o_1, a_1)\big)\big)\big) \\
&= \bigwedge_{o_1\in B_1} \big(f_1(o_1) \to \bigwedge_{a_1\in A_1} \big(\bigvee_{a_2\in A_2} \big(\varphi_R(a_2)(a_1) \otimes g_2(a_2)\big) \to r_1(o_1, a_1)\big)\big) \\
&= \bigwedge_{o_1\in B_1} \big(f_1(o_1) \to \downarrow_1 \big(\varphi_{R+}(g_2)\big)(o_1)\big) \\
&= \langle f_1, g_1\rangle \preceq_1 \big\langle \downarrow_1 \big(\varphi_{R+}(g_2)\big), \uparrow_1\downarrow_1 \big(\varphi_{R+}(g_2)\big)\big\rangle \\
&= \langle f_1, g_1\rangle \preceq_1 \varphi_\wedge\big(\langle f_2, g_2\rangle\big)
\end{aligned}$$

□

Lemma 3. *For the identity arrow $\iota \in L\text{-ChuCors}(C, C)$ of any L-context $C = \langle B, A, r\rangle$, $\Gamma(\iota)$ is the identity arrow from $L\text{-CLLOS}(\Gamma(C), \Gamma(C))$.*

Proof. Consider any L-concept $\langle f, g\rangle$ from $L\text{-FCL}(C)$.

$$\begin{aligned}
\uparrow \big(\iota_{L+}(f)\big)(a) &= \bigwedge_{o\in B} \big(\iota_{L+}(f)(o) \to r(o, a)\big) \\
&= \bigwedge_{o\in B} \big(\bigvee_{b\in B} \big(\iota_L(b)(o) \otimes f(b)\big) \to r(o, a)\big)
\end{aligned}$$

$$\begin{aligned}
&= \bigwedge_{o\in B}\bigwedge_{b\in B}\big(\big(\iota_L(b)(o)\otimes f(b)\big)\to r(o,a)\big)\\
&= \bigwedge_{o\in B}\big(f(b)\to \bigwedge_{b\in B}\big(\iota_L(b)(o)\to r(o,a)\big)\\
&= \bigwedge_{b\in B}\big(f(b)\to \uparrow\downarrow\uparrow\big(\chi_b\big)(a)\big)\\
&= \bigwedge_{b\in B}\big(f(b)\to r(b,a)\big) = \uparrow(f)(a)
\end{aligned}$$

Therefore, we have $\iota_\vee(\langle f,g\rangle)=\langle f,g\rangle$. The proof for $\iota_\wedge$ is similar. □

Lemma 4. *Consider arbitrary $\varphi_i \in L$-ChuCors(C_i, C_{i+1}) for $i\in\{1,2\}$ and any element $o_1\in B_1$ and $g_3\in L^{A_3}$. Then*

$$\downarrow_1\big(\varphi_{1R^+}\big(\varphi_{2R^+}(g_3)\big)\big)(o_1) = \downarrow_1\big(\varphi_{1R^+}\big(\uparrow_2\downarrow_2\big(\varphi_{2R^+}(g_3)\big)\big)\big)(o_1).$$

Proof.

$$\begin{aligned}
&\downarrow_1\big(\varphi_{1R^+}\big(\varphi_{2R^+}(g_3)\big)\big)(o_1)\\
&= \bigwedge_{a_1\in A_1}\big(\varphi_{1R^+}\big(\varphi_{2R^+}(g_3)\big)(a_1)\to r_1(o_1,a_1)\big)\\
&= \bigwedge_{a_1\in A_1}\big(\bigvee_{a_2\in A_2}\big(\varphi_{1R}(a_2)(a_1)\otimes\varphi_{2R^+}(g_3)(a_2)\big)\to r_1(o_1,a_1)\big)\\
&= \bigwedge_{a_2\in A_2}\big(\varphi_{2R^+}(g_3)(a_2)\to\bigwedge_{a_1\in A_1}\big(\varphi_{1R}(a_2)(a_1)\to r_1(o_1,a_1)\big)\big)\\
&= \bigwedge_{a_2\in A_2}\big(\varphi_{2R^+}(g_3)(a_2)\to\bigwedge_{o_2\in B_2}\big(\varphi_{1L}(o_1)(o_2)\to r_2(o_2,a_2)\big)\big)\\
&= \bigwedge_{o_2\in B_2}\big(\varphi_{1L}(o_1)(o_2)\to\bigwedge_{a_2\in A_2}\big(\varphi_{2R^+}(g_3)(a_2)\to r_2(o_2,a_2)\big)\big)\\
&= \bigwedge_{o_2\in B_2}\big(\varphi_{1L}(o_1)(o_2)\to\downarrow_2\uparrow_2\downarrow_2\big(\varphi_{2R^+}(g_3)\big)(o_2)\big)
\end{aligned}$$

by applying the same chain of modifications in opposite way we will obtain

$$= \downarrow_1\big(\varphi_{1R^+}\big(\uparrow_2\downarrow_2\big(\varphi_{2R^+}(g_3)\big)\big)\big)(o_1)$$

□

Lemma 5. *Mapping Γ is closed under arrow composition.*

Proof. Consider $\varphi_i\in L$-ChuCors(C_i,C_{i+1}) for $i\in\{1,2\}$. Let $\langle f_i,g_i\rangle\in L$-FCL$(C_i)$ be an arbitrary L-context for all $i\in\{1,3\}$. Recall that

1. $\Gamma(\varphi_2\circ\varphi_1)=\big\langle(\varphi_2\circ\varphi_1)_\vee,(\varphi_2\circ\varphi_1)_\wedge\big\rangle$
2. $\Gamma(\varphi_2)\circ\Gamma(\varphi_1)=\big\langle\varphi_{2\vee}\circ\varphi_{1\vee},\varphi_{1\wedge}\circ\varphi_{2\wedge}\big\rangle$

The proof will be based on equality of corresponding elements of the previous pairs: only one part will be proved, the other one is similar.

$$\begin{aligned}
&(\varphi_{1\wedge} \circ \varphi_{2\wedge})\big(\langle f_3, g_3\rangle\big) = \varphi_{1\wedge}\big(\varphi_{2\wedge}\big(\langle f_3, g_3\rangle\big)\big)\\
=&\Big\langle \downarrow_1 (\varphi_{1R^+}(\uparrow_2\downarrow_2 (\varphi_{2R^+}(g_3)))), \uparrow_1\downarrow_1 (\varphi_{1R^+}(\uparrow_2\downarrow_2 (\varphi_{2R^+}(g_3))))\Big\rangle\\
&\text{by lemma 4 we have}\\
=&\Big\langle \downarrow_1 (\varphi_{1R^+}(\varphi_{2R^+}(g_3))), \uparrow_1\downarrow_1 (\varphi_{1R^+}(\varphi_{2R^+}(g_3)))\Big\rangle = \star
\end{aligned}$$

$$\begin{aligned}
&\downarrow_1 (\varphi_{1R^+}(\varphi_{2R^+}(g_3)))(o_1) =\\
=& \bigwedge_{a_1\in A_1} \Big(\bigvee_{a_2\in A_2} (\varphi_{1R}(a_2)(a_1) \otimes \varphi_{2R^+}(g_3)(a_2)) \to r_1(o_1, a_1)\Big)\\
=& \bigwedge_{a_1\in A_1} \Big(\bigvee_{a_2\in A_2} (\varphi_{1R}(a_2)(a_1) \otimes \bigvee_{a_3\in A_3} (\varphi_{2R}(a_3)(a_2) \otimes g_3(a_3))) \to r_1(o_1, a_1)\Big)\\
=& \bigwedge_{a_1\in A_1} \Big(\bigvee_{a_3\in A_3} \big(\varphi_{1R^+}(\varphi_{2R}(a_3))(a_1) \otimes g_3(a_3)\big) \to r_1(o_1, a_1)\Big)\\
=& \bigwedge_{a_3\in A_3} \Big(g_3(a_3) \to \bigwedge_{a_1\in A_1} \big(\varphi_{1R^+}(\varphi_{2R}(a_3))(a_1) \to r_1(o_1, a_1)\big)\Big)\\
=& \bigwedge_{a_3\in A_3} \Big(g_3(a_3) \to\downarrow_1\uparrow_1\downarrow_1 \big(\varphi_{1R^+}(\varphi_{2R}(a_3))\big)(o_1)\Big)\\
=& \bigwedge_{a_3\in A_3} \Big(g_3(a_3) \to\downarrow_1 \big((\varphi_2 \circ \varphi_1)_R(a_3)\big)(o_1)\Big)\\
=& \bigwedge_{a_3\in A_3} \Big(g_3(a_3) \to \bigwedge_{a_1\in A_1} \big((\varphi_2 \circ \varphi_1)_R(a_3)(a_1) \to r_1(o_1, a_1)\big)\Big)\\
=& \bigwedge_{a_1\in A_1} \bigwedge_{a_3\in A_3} \Big(\big(g_3(a_3) \otimes (\varphi_2 \circ \varphi_1)_R(a_3)(a_1)\big) \to r_1(o_1, a_1)\Big)\\
=& \bigwedge_{a_1\in A_1} \Big(\bigvee_{a_3\in A_3} (g_3(a_3) \otimes (\varphi_2 \circ \varphi_1)_R(a_3)(a_1)) \to r_1(o_1, a_1)\Big)\\
=&\downarrow_1 \big((\varphi_2 \circ \varphi_1)_{R^+}(g_3)\big)(o_1)
\end{aligned}$$

$$\begin{aligned}
\star =&\Big\langle \downarrow_1 ((\varphi_2 \circ \varphi_1)_{R^+}(g_3)), \uparrow_1\downarrow_1 ((\varphi_2 \circ \varphi_1)_{R^+}(g_3))\Big\rangle\\
=&(\varphi_2 \circ \varphi_1)_\wedge(\langle f_3, g_3\rangle)
\end{aligned}$$

□

Proposition 1. *Γ is a functor from L-*ChuCorrs *to L-*CLLOS*.*

Proof. Straightforward from the previous lemmas. □

We continue by showing that the previous functor satisfies the conditions to define a categorical equivalence, characterized by the following result:

Theorem 3 (See [2]). *The following conditions on a functor $F : \mathcal{C} \to \mathcal{D}$ are equivalent:*

- *F is an equivalence of categories.*
- *F is full and faithful and "essentially surjective" on objects: for every $D \in \mathcal{D}$ there is some $C \in \mathcal{C}$ such that $F(C) \cong D$.*

Let us recall the definition of the notions required by the previous theorem:

Definition 14.

1. *A functor $F : \mathcal{C} \to \mathcal{D}$ is* faithful *if for all objects A, B of a category $\mathcal{C}$, the map $F_{A,B} : \mathrm{Hom}_{\mathcal{C}}(A, B) \to \mathrm{Hom}_{\mathcal{D}}(F(A), F(B))$ is injective.*
2. *Similarly, F is* full *if $F_{A,B}$ is always surjective.*

In our cases, for proving fullness and faithfulness of the functor Γ we need to prove surjectivity and injectivity of the mapping

$$\Gamma_{C_1,C_2} : L\text{-ChuCors}(C_1, C_2) \to L\text{-CLLOS}(\Gamma(C_1), \Gamma(C_2))$$

for any two L-contexts C_1 and C_2. This will be done in the forthcoming lemmas.

Lemma 6. *Γ is full.*

Proof. The point of the proof is to show that given any arrow $\langle s, z\rangle$ from the set L-CLLOS$(\Gamma(C_1), \Gamma(C_2))$ there exists an L-Chu correspondence $\varphi^{\langle s,z\rangle}$ from the set L-ChuCors(C_1, C_2), for any two L-contexts $C_i = \langle B_i, A_i, r_i\rangle$ for $i = \{1, 2\}$. Let us define the following mappings:

- $\varphi_L^{\langle s,z\rangle}(o_1) = Ext\big(s\big(\langle \downarrow_1\uparrow_1 (\chi_{o_1}), \uparrow_1 (\chi_{o_1})\rangle\big)\big)$
- $\varphi_R^{\langle s,z\rangle}(a_2) = Int\big(z\big(\langle \downarrow_2 (\chi_{a_2}), \uparrow_2\downarrow_2 (\chi_{a_2})\rangle\big)\big)$

$$\begin{aligned}
\uparrow_2 \big(\varphi_L^{\langle s,z\rangle}(o_1)\big)(a_2) &= \bigwedge_{o_2\in B_2} \big(\varphi_L^{\langle s,z\rangle}(o_1)(o_2) \to r_2(o_2, a_2)\big)\\
&= \bigwedge_{o_2\in B_2} \big(Ext(s(\langle \downarrow_1\uparrow_1 (\chi_{o_1}), \uparrow_1 (\chi_{o_1})\rangle))(o_2) \to \downarrow_2 (\chi_{a_2})(o_2)\big)\\
&= s\big(\langle \downarrow_1\uparrow_1 (\chi_{o_1}), \uparrow_1 (\chi_{o_1})\rangle\big) \preceq_2 \langle \downarrow_2 (\chi_{a_2}), \uparrow_2\downarrow_2 (\chi_{a_2})\rangle\\
&= \langle \downarrow_1\uparrow_1 (\chi_{o_1}), \uparrow_1 (\chi_{o_1})\rangle \preceq_1 z\big(\langle \downarrow_2 (\chi_{a_2}), \uparrow_2\downarrow_2 (\chi_{a_2})\rangle\big)\\
&= \bigwedge_{a_1\in A_1} \big(Int(z(\langle \downarrow_2 (\chi_{a_2}), \uparrow_2\downarrow_2 (\chi_{a_2})\rangle))(a_1) \to \uparrow_1 (\chi_{o_1})(a_1)\big)\\
&= \bigwedge_{a_1\in A_1} \big(\varphi_R^{\langle s,z\rangle}(a_2)(a_1) \to r_1(o_1, a_1)\big)\\
&= \downarrow_1 \big(\varphi_R^{\langle s,z\rangle}(a_2)\big)(o_1)
\end{aligned}$$

So $\varphi^{\langle s,z\rangle} \in L$-ChuCors$(C_1, C_2)$ and Γ_{C_1,C_2} is surjective, hence Γ is full. □

Lemma 7. *$\varGamma$ is faithfull*

Proof. Now the point is to prove the injectivity of $\varGamma_{C_1,C_2}$.

Consider two L-Chu correspondences φ_1, φ_2 from L-ChuCors(C_1, C_2) such that $\varphi_1 \neq \varphi_2$, and let us fix the pair $(o_1, a_2) \in B_1 \times A_2$, such that

$$\uparrow_2 \big(\varphi_{1L}(o_1)\big)(a_2) = \downarrow_1 \big(\varphi_{1R}(a_2)\big)(o_1) \neq \uparrow_2 \big(\varphi_{2L}(o_1)\big)(a_2) = \downarrow_1 \big(\varphi_{2R}(a_2)\big)(o_1)$$

Let us assume that either $\downarrow_1 \big(\varphi_{1R}(a_2)\big)(o_1) > \uparrow_2 \big(\varphi_{2L}(o_1)\big)(a_2)$ or that both values from L are incomparable, that is equivalent to the following:

$$\downarrow_1 \big(\varphi_{1R}(a_2)\big)(o_1) \to \uparrow_2 \big(\varphi_{2L}(o_1)\big)(a_2) < 1$$

Now consider the L-concept $\langle \downarrow_1 (\varphi_{1R}(a_2)), \varphi_{1R}(a_2)\rangle$ and let us compare its images under the mappings $\varphi_{1\vee}$ and $\varphi_{2\vee}$.

$$\begin{aligned}
&\uparrow_2 \big(\varphi_{2L^+}\big(\downarrow_1 (\varphi_{1R}(a_2))\big)\big)(a_2) \\
&= \bigwedge_{o_2 \in B_2} \big(\varphi_{2L^+}(\downarrow_1 (\varphi_{1R}(a_2)))(o_2) \to r_2(o_2, a_2)\big) \\
&= \bigwedge_{o_2 \in B_2} \big(\bigvee_{b_1 \in B_1} \big(\varphi_{2L}(b_1)(o_2) \otimes \downarrow_1 (\varphi_{1R}(a_2))(b_1)\big) \to r_2(o_2, a_2)\big) \\
&= \bigwedge_{b_1 \in B_1} \big(\downarrow_1 (\varphi_{1R}(a_2))(b_1) \to \bigwedge_{o_2 \in B_2} \big(\varphi_{2L}(b_1)(o_2) \to r_2(o_2, a_2)\big)\big) \\
&= \bigwedge_{b_1 \in B_1} \big(\downarrow_1 (\varphi_{1R}(a_2))(b_1) \to \uparrow_2 (\varphi_{2L}(b_1))(a_2)\big) \\
&\leq \downarrow_1 (\varphi_{1R}(a_2))(o_1) \to \uparrow_2 (\varphi_{2L}(o_1))(a_2) \\
&< 1 \text{ because of the restriction given above}
\end{aligned}$$

Similarly, we can obtain:

$$\begin{aligned}
&\uparrow_2 (\varphi_{1L^+}(\downarrow_1 (\varphi_{1R}(a_2))))(a_2) = \\
&= \bigwedge_{b_1 \in B_1} \big(\downarrow_1 (\varphi_{1R}(a_2))(b_1) \to \uparrow_2 (\varphi_{1L}(b_1))(a_2)\big) \\
&= \bigwedge_{b_1 \in B_1} \big(\downarrow_1 (\varphi_{1R}(a_2))(b_1) \to \downarrow_1 (\varphi_{1R}(a_2))(b_1)\big) = 1
\end{aligned}$$

It means that $\varphi_{1\vee}\big(\langle\downarrow_1 (\varphi_{1R}(a_2)), \varphi_{1R}(a_2)\rangle\big) \neq \varphi_{2\vee}\big(\langle\downarrow_1 (\varphi_{1R}(a_2)), \varphi_{1R}(a_2)\rangle\big)$

Hence $\varphi_{1\vee}(\langle\downarrow_1 (\varphi_{1R}(a_2)), \varphi_{1R}(a_2)\rangle) \neq \varphi_{2\vee}(\langle\downarrow_1 (\varphi_{1R}(a_2)), \varphi_{1R}(a_2)\rangle)$ and $\varphi_{1\vee} \neq \varphi_{2\vee}$. So $\varGamma_{C_1,C_2}$ is injective and $\varGamma$ is faithfull. $\square$

Proposition 2. *The functor $\varGamma$ is an equivalence functor between L-*ChuCors *and L-*CLLOS*.*

Proof. Fullness and faithfulness of $\varGamma$ is given by previous lemmas. Essential surjectivity on objects is ensured by the fact that given any object $\langle\langle V, \approx\rangle, \preceq\rangle$ of L-CLLOS there exists an L-context $\langle V, V, \preceq\rangle$, such that $\varGamma\big(\langle V, V, \preceq\rangle\big) \cong \langle\langle V, \approx\rangle, \preceq\rangle$. Hence, we can state that $\varGamma$ is the functor of equivalence between L-ChuCors and L-CLLOS. $\square$

References

1. C. Alcalde, A. Burusco, R. Fuentes-González, and I. Zubia. The use of linguistic variables and fuzzy propositions in the L-Fuzzy concept theory. *Computers & Mathematics with Applications* 62(8): 3111–3122, 2011.
2. S. Awodey. *Category Theory*, Oxford Logic Guides 49, 2006.
3. R. Bělohlávek. Fuzzy concepts and conceptual structures: induced similarities. In *Joint Conference on Information Sciences*, pages 179–182, 1998.
4. R. Bělohlávek. *Fuzzy Relational Systems: Foundation and Principles,* Kluwer, 2002.
5. R. Bělohlávek. Lattices of fixed points of fuzzy Galois connections. *Mathematical Logic Quartely*, 47(1):111–116, 2001.
6. R. Bělohlávek. Concept lattices and order in fuzzy logic. *Annals of Pure and Applied Logic*, 128:277–298, 2004.
7. P. du Boucher-Ryana and D. Bridge. Collaborative recommending using formal concept analysis. *Knowledge-Based Systems*, 19(5):309–315, 2006.
8. P. Burillo, R. Fuentes-González, L. González. On Completeness and Direction in Fuzzy Relational Systems. *Mathware and Soft Computing*, 5:243–251, 1998.
9. R.-C. Chen, C.-T. Bau, and C-J. Yeh Merging domain ontologies based on the WordNet system and Fuzzy Formal Concept Analysis techniques, *Applied Soft Computing* 11(2):1908–1923, 2011.
10. D. Dubois and H. Prade. Possibility theory and formal concept analysis: Characterizing independent sub-contexts, *Fuzzy Sets and Systems* 196: 4–16, 2012.
11. A. Formica. Concept similarity in formal concept analysis: An information content approach. *Knowledge-Based Systems*, 21(1):80–87, 2008.
12. A. Formica. Semantic Web search based on rough sets and Fuzzy Formal Concept Analysis. *Knowledge-Based Systems*, 26:40–47, 2012.
13. J. Goguen. L-fuzzy sets. *J. Math. Anal. Appl.*, 18:145–174, 1967.
14. S. Krajči. A categorical view at generalized concept lattices. *Kybernetika*, 43(2):255–264, 2007.
15. 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.
16. O. Krídlo and M. Ojeda Aciego. On L-fuzzy Chu correspondences. *Intl J of Computer Mathematics*, 88(9):1808–1818, 2011.
17. M. Krötzsch, P. Hitzler, and G.-Q. Zhang. Morphisms in context. *Lecture Notes in Computer Science*, 3596:223–237, 2005.
18. Y. Lei and M. Luo. Rough concept lattices and domains. *Annals of Pure and Applied Logic*, 159:333–340, 2009.
19. S.-T. Li and F.C. Tsai. Noise control in document classification based on fuzzy formal concept analysis. *Proc. of FUZZ-IEEE'11*:2583–2588, 2011.
20. J. Medina. Multi-adjoint property-oriented and object-oriented concept lattices. *Information Sciences*, 190:95–106, 2012.
21. J. Medina and M. Ojeda-Aciego. Multi-adjoint t-concept lattices. *Information Sciences*, 180(5):712–725, 2010.
22. J. Medina, M. Ojeda-Aciego, and J. Ruiz-Calviño. Formal concept analysis via multi-adjoint concept lattices. *Fuzzy Sets and Systems*, 160(2):130–144, 2009.
23. H. Mori. Chu correspondences. *Hokkaido Mathematical Journal*, 37:147–214, 2008.
24. Q. Wu and Z. Liu. Real formal concept analysis based on grey-rough set theory. *Knowledge-Based Systems*, 22(1):38–45, 2009.
25. G.-Q. Zhang. Chu spaces, concept lattices, and domains. *Electronic Notes in Theoretical Computer Science*, 83, 2004.

Using intensifying hedges to reduce size of multi-adjoint concept lattices with heterogeneous conjunctors

Jan Konecny[1], Jesús Medina[2], and Manuel Ojeda-Aciego[3]

[1] Dept. Computer Science, Palacky University, Olomouc, Czech Republic*
[2] Dept. Mathematics, University of Cádiz, Spain**
[3] Dept. Applied Mathematics, University of Málaga, Spain***

Abstract. In this work we focus on the use of intensifying hedges as a tool to reduce the size of the recently introduced multi-adjoint concept lattices with heterogeneous conjunctors.

1 Introduction and preliminaries

Formal concept analysis (FCA) is a very active topic for several research groups throughout the world [6, 1, 3, 5, 7, 8, 10, 11]. In this work, the authors aim to merge recent advances obtained in this area: on the one hand, the use of hedges as structures which allow to modulate the size of fuzzy concept lattices [4] and, on the other hand, the consideration of heterogeneous conjunctors in the general approach to fuzzy FCA so-called multi-adjoint framework [9].

One of the key features of the latter approach is that some quasi-closure operators arise which, although do not directly allow to prove the complete lattice structure of the resulting set of concepts as usual, i.e. in terms of a Galois connection, actually do provide means to manually build the operators for suprema and infima of a set of concepts. The core notion in [9] is that of P-connected pair of posets which, in some sense, turns out to be a more abstract notion than a truth-stressing hedge. As a consequence of this observation, due to Radim Belohlavek, we now focus on the use of the specific properties of hedges in order to import some results related to the size of fuzzy concept lattices to the more general framework of [9].

The structure of the paper is the following: in Section 2 the preliminary definitions are introduced, interested readers will obtain further comment on the intuitions underlying the definitions in the original papers [4, 9]; the main results are presented in Section 3.

* Partially supported by grant No. P202/10/0262 of the Czech Science Foundation.
** Partially supported by Spanish Ministry of Science and FEDER funds through project TIN09-14562-C05-03 and Junta de Andalucía project P09-FQM-5233.
*** Partially supported by Spanish Ministry of Science and FEDER funds through project TIN09-14562-C05-01 and Junta de Andalucía project P09-FQM-5233.

2 Preliminaries

In this section, we introduce the basic definitions and preliminary results which will be used later in the core of this work.

Definition 1. *Let $(L, \preceq, \top, \bot)$ be a complete lattice, a truth-stressing hedge in L is a mapping $*\colon L \to L$ satisfying, for each $x, y \in L$,*

$$*(\top) = \top, \tag{1}$$

$$*(x) \preceq x, \tag{2}$$

$$x \preceq y \text{ implies } *(x) \preceq *(y), \tag{3}$$

$$*(*(x)) = *(x) \tag{4}$$

$\operatorname{fix}(*)$ *denotes set of fixed points of $*$ in L, i.e.* $\operatorname{fix}(*) = \{a \in L \mid *(a) = a\}$.

In [3, 4] truth-stressing hedges were used to decrease size of a concept lattice (in fact, the truth-stressing hedges were defined on a residuated lattice).

Later in this work, we will need the following lemmas.

Lemma 1. *Let $(L, \preceq)$ be a complete lattice, for any mapping $*\colon L \to L$ satisfying (2), (3), and (4) we have, for each $x_i \in L$,*

$$\bigvee_{i\in I} *(x_i) = *(\bigvee_{i\in I} *(x_i)) \quad \textit{and} \quad *(\bigwedge_{i\in I} *(x_i)) = *(\bigwedge_{i\in I} x_i). \tag{5}$$

In addition, if we have $x_j = \bigvee_{i\in I} x_i$ for some $j \in I$ then

$$*(\bigvee_{i\in I} x_i) = \bigvee_{i\in I} *(x_i). \tag{6}$$

Similarly, if we have $x_j = \bigwedge_{i\in I} x_i$ for some $j \in I$ then

$$*(\bigwedge_{i\in I} x_i) = \bigwedge_{i\in I} *(x_i). \tag{7}$$

Lemma 2. **(a)** *Let $*\colon L \to L$ be a mapping satisfying (2), (3), and (4). Then $\operatorname{fix}(*)$ is a $\vee$-subsemilattice of L.*
(b) *Let K be a $\vee$-subsemilattice of L then the mapping $*_K\colon L \to L$ defined by*

$$*_K(x) = \bigvee\{y \in K \mid y \leq x\}$$

satisfies (2), (3), and (4).
(c) $*_{\operatorname{fix}(*)} = *$ *and* $\operatorname{fix}(*_K) = K$.

By Lemma 2 the set $\operatorname{fix}(*)$ of truth-stressing hedge $*$ is a $\vee$-subsemilattice. Now we will introduce the basic notions of multi-adjoint concept lattices with heterogeneous conjunctors, in order to show how both frameworks, hedges and heterogeneous conjunctors, can be merged.

Firstly, let us introduced a bit of terminology: in the rest of this work we will call a mapping $*\colon L \to L$ satisfying (2), (3), and (4) an **intensifying hedge**, following the terminology introduced in [2]. In terms of interior structures $(L, \preceq)$, a mapping satisfying (2)–(4) is an interior operator on the lattice of truth degrees.

The two main notions on which multi-adjoint concept lattices with heterogeneous conjunctors is defined are given below: the P-connection between posets, and the adjoint triples.

Definition 2. *Given the posets $(P_1, \leq_1)$, $(P_2, \leq_2)$ and $(P, \leq)$, we say that P_1 and P_2 are* P-connected *if there exist non-decreasing mappings $\psi_1\colon P_1 \to P$, $\phi_1\colon P \to P_1$, $\psi_2\colon P_2 \to P$ and $\phi_2\colon P \to P_2$ verifying that $\phi_1(\psi_1(x)) = x$, and $\phi_2(\psi_2(y)) = y$, for all $x \in P_1$, $y \in P_2$.*

Definition 3. *Let $(P_1, \leq_1)$, $(P_2, \leq_2)$, $(P_3, \leq_3)$ be posets, and consider mappings $\&\colon P_1 \times P_2 \to P_3$, $\swarrow\colon P_3 \times P_2 \to P_1$, $\nwarrow\colon P_3 \times P_1 \to P_2$, then $(\&, \swarrow, \nwarrow)$ is an* adjoint triple *with respect to P_1, P_2, P_3 if: $x \leq_1 z \swarrow y$ iff $x \,\&\, y \leq_3 z$ iff $y \leq_2 z \nwarrow x$, where $x \in P_1$, $y \in P_2$ and $z \in P_3$.*

From Lemma 2 we immediately obtain the following proposition:

Corollary 1. *Consider the posets $(P_1, \leq_1)$, $(P_2, \leq_2)$ and $(P, \leq)$, and assume that L_1 and L_2 are P-connected, then:*

(a) If $\psi_1 \circ \phi_1$ is contractive (i.e. satisfies (2)*) then P_1 is isomorphic to a $\vee$-subsemilattice of P.*
(b) If $*\colon P_1 \to P_1$ is an intensifying hedge (i.e. satisfies properties (2), (3), *and* (4)*) then the composition $\psi_1 \circ * \circ \phi_1\colon P \to P$ is an intensifying hedge in* $\mathrm{fix}(\psi_1 \circ \phi_1)$.

Lemma 3. *Let $(L, \preceq), (L_1, \preceq_1), (L_2, \preceq_2)$ be lattices and let $(\&, \swarrow, \nwarrow)$ be an adjoint triple. For $a, a_i \in L_1$, $b, b_i \in L_2$, we have*

$$\bigvee_{i\in I}(a_i \,\&\, b) = (\bigvee_{1 i\in I} a) \,\&\, b \quad \text{and} \quad \bigvee_{i\in I}(a \,\&\, b_i) = a \,\&(\bigvee_{2 i\in I} b_i) \tag{8}$$

Definition 4. *A multi-adjoint frame is a tuple*

$$(L_1, L_2, P, \&_1, \swarrow^1, \nwarrow_1, \dots, \&_n, \swarrow^n, \nwarrow_n)$$

where L_i are complete lattices and P i a poset, such that $(\&_i, \swarrow^i, \nwarrow_i)$ is an adjoint triple with respect to L_1, L_2, P for all $i = 1, \dots, n$.

Definition 5. *Let $(L_1, L_2, P, \&_1, \dots, \&_n)$ be a multi-adjoint frame, a* multi-adjoint context *is a tuple (A, B, R, σ) such that A and B are non-empty sets (usually interpreted as attributes and objects, respectively), R is a P-fuzzy relation $R\colon A \times B \to P$ and $\sigma\colon B \to \{1, \dots, n\}$ is a mapping which associates any element in B with some particular adjoint triple in the frame.*

Given a complete lattice $(L, \preceq)$ such that L_1 and L_2 are L-connected, a multi-adjoint frame $(L_1, L_2, P, \&_1, \dots, \&_n)$, and a context (A, B, R, σ), we can define the mappings ${}^{\uparrow_{c\sigma}}\colon L^B \to L^A$ and ${}^{\downarrow^{c\sigma}}\colon L^A \to L^B$ defined for all $g \in L^B$ and $f \in L^A$ as follows:

$$g^{\uparrow_{c\sigma}}(a) = \psi_1(\inf\{R(a,b) \swarrow^{\sigma(b)} \phi_2(g(b)) \mid b \in B\}) \tag{9}$$

$$f^{\downarrow^{c\sigma}}(b) = \psi_2(\inf\{R(a,b) \nwarrow_{\sigma(b)} \phi_1(f(a)) \mid a \in A\}) \tag{10}$$

The notion of concept is defined as usual. A *concept* is a pair $\langle g, f\rangle$ satisfying $g \in L^B$, $f \in L^A$ and that $g^{\uparrow_{c\sigma}} = f$ and $f^{\downarrow^{c\sigma}} = g$.

Definition 6. *Given the complete lattices $(L_1, \preceq_1)$, $(L_2, \preceq_2)$ and $(L, \preceq)$, where L_1 and L_2 are L-connected, the set of* multi-adjoint L-connected concepts *associated to a multi-adjoint frame $(L_1, L_2, P, \&_1, \dots, \&_n)$ and context (A, B, R, σ) is given by $\mathfrak{M}_L = \{\langle g, f\rangle \mid \langle g, f\rangle$ is a concept$\}$.*

The main theorem of concept lattices in [9], proves that $\mathfrak{M}_L$ has the structure of a complete lattice:

Theorem 1 ([9]). *Given complete lattices $(L_1, \preceq_1)$, $(L_2, \preceq_2)$ and $(L, \preceq)$, where L_1 and L_2 are L-connected, a context (A, B, R, σ), and a multi-adjoint frame $(L_1, L_2, L, \&_1, \dots, \&_n)$, the multi-adjoint L-connected concept lattice $\mathfrak{M}_L$ is actually a complete lattice with the meet and join operators $\curlywedge, \curlyvee\colon \mathfrak{M}_L \times \mathfrak{M}_L \to \mathfrak{M}_L$ defined below, for all $\langle g_1, f_1\rangle, \langle g_2, f_2\rangle \in \mathfrak{M}_L$,*

$$\langle g_1, f_1\rangle \curlywedge \langle g_2, f_2\rangle = \langle \psi_2 \circ \phi_2(g_1 \wedge g_2), (f_1 \vee f_2)^{\downarrow^c \uparrow_c}\rangle$$
$$\langle g_1, f_1\rangle \curlyvee \langle g_2, f_2\rangle = \langle (g_1 \vee g_2)^{\uparrow_c \downarrow^c}, \psi_1 \circ \phi_1(f_1 \wedge f_2)\rangle$$

The order $\preceq$ which corresponds to $\curlywedge$ and $\curlyvee$ is defined as

$$\langle g_1, f_1\rangle \preceq \langle g_2, f_2\rangle \quad \text{iff} \quad \phi_2(g_1) \le \phi_2(g_2) \quad (\text{iff } \phi_1(f_2) \le \phi_1(f_1))$$

In what follows $\mathfrak{M}$ denotes multi-adjoint L-connected concept lattice of given context (A, B, R, σ). We will also omit subscript $\sigma(b)$ and write just $\swarrow$ instead of $\swarrow^{\sigma(b)}$.

3 Reducing the size of multi-adjoint concept lattices

The size of the concept lattice $\mathfrak{M}$ can be reduced either by a suitable selection of a $\vee$-subsemilattice of L_1 (and/or L_2) and the use of a restriction of $\&$. The following proposition says that the selection of $\vee$-subsemilattices of L_1 (resp. L_2) yields a reduction of size of concept lattice and, moreover, preserves intents (or extents) of the original concept lattice, meaning that each intent of the reduced concept lattice is an intent of the original concept lattice.

Proposition 1. *Let* $\mathbf{A} = (L_1, L_2, P, \&_1, \dots, \&_n), \mathbf{A}' = (K_1, L_2, P, \&'_1, \dots, \&'_n)$ *be multi-adjoint frames, s.t.* K_1 *is a* $\vee$*-subsemilattice of* L_1*, and* $\&'_1, \dots, \&'_n$ *are restrictions of* $\&_1, \dots, \&_n$ *to* $K_1 \times L_2$ *and* $\psi'_1 = \psi_1$*,* $\psi'_2 = \psi_2$*,* $\phi'_2 = \phi_2$*,* $\phi'_1 = *_{K_1} \circ \phi_1$*, where* $*_{K_1}$ *is the hedge associated to* K_1 *as introduced in Lemma 2. Then,* $\mathrm{Int}(\mathfrak{M}_{\mathbf{A}'}) \subseteq \mathrm{Int}(\mathfrak{M}_{\mathbf{A}})$ *where* $\mathrm{Int}(\mathfrak{M})$ *denotes the set of intents in* $\mathfrak{M}$*.*

Proof (sketch). We have $z \nwarrow' x = z \nwarrow x$, for each $x \in K_1, z \in P$, whence $f^{\downarrow'} = f^{\downarrow}$, for each $f \colon A \to \psi_1(K_1)$ where $\psi_1(K_1) \in L$ is image of ψ_1 (note that $\swarrow'$ is well-defined since K_1 is $\vee$-subsemilattice) and thus by Proposition 16 in [9] $\mathrm{Ext}(\mathfrak{M}_{\mathbf{A}'}) \subseteq \mathrm{Ext}(\mathfrak{M}_{\mathbf{A}})$. □

Remark 1. One can state a dual proposition to Proposition 1 for intents. Let $\mathbf{A} = (L_1, L_2, P, \&_1, \dots, \&_n), \mathbf{A}' = (L_1, K_2, P, \&'_1, \dots, \&'_n)$ be multi-adjoint frames, s.t. K_2 is a $\vee$-subsemilattice of L_2, and $\&'_1, \dots, \&'_n$ are restrictions of $\&_1, \dots, \&_n$ to $L_1 \times K_2$ and $\phi'_2 = *_{K_2} \circ \phi_2$.

The following proposition says that by selection of $\vee$-subsemilattices of both L_1 and L_2 we obtain a reduction of the size as well. However, the preservation of intents (or extents) is lost.

Proposition 2. *Let* $\mathbf{A} = (L_1, L_2, L, \&_1, \dots, \&_n), \mathbf{A}' = (K_1, K_2, L, \&'_1, \dots, \&'_n)$ *be multi-adjoint frames, s.t.* K_1 *is a* $\vee$*-subsemilattice of* L_1*,* K_2 *is a* $\vee$*-subsemilattice of* L_2*, and* $\&'_1, \dots, \&'_n$ *are restrictions of* $\&_1, \dots, \&_n$ *to* $K_1 \times K_2$*, and* $\phi'_1 = *_{K_1} \circ \phi_1, \phi'_2 = *_{K_2} \circ \phi_2$*. Then we have* $|\mathfrak{M}_{\mathbf{A}'}| \leq |\mathfrak{M}_{\mathbf{A}}|$*.*

Proof (sketch). By applying Proposition 1 and Remark 1 we obtain the result. □

In the next result we show how to generate new adjoint triples using hedges.

Lemma 4. *Assume* $(\&, \swarrow, \nwarrow)$ *is an adjoint triple with respect to* L_1*,* L_2*,* P*, and* $*_1 \colon L_1 \to L_1$*,* $*_2 \colon L_2 \to L_2$ *are hedges, then* $x \&^* y = *_1(x) \& *_2(y)$ *has two residuated implications* $\swarrow^*, \nwarrow_*$ *which form a new adjoint triple with respect to* L_1*,* L_2*,* P*, if and only if the following equalities hold:*

$$*_1(z \swarrow *_2(y)) = *_1(\bigvee\{x \mid x \&^* y \leq z\}) \tag{11}$$

$$*_2(z \nwarrow *_1(x)) = *_2(\bigvee\{y \mid x \&^* y \leq z\}) \tag{12}$$

Proof. "$\Rightarrow$": Let $(\&^*, \swarrow^*, \nwarrow_*)$ be an adjoint triple. We have

$$x \&^* y \leq z \quad \text{iff} \quad y \preceq_2 z \nwarrow_* x$$

by definition. In particular, we obtain

$$*_1(x) \&^* *_2(y) \leq z \quad \text{iff} \quad *_2(y) \preceq_2 z \nwarrow_* *_1(x)$$

and $*_1(x) \&^* *_2(y) = *_1(*_1(x)) \& *_2(*_2(y)) = *_1(x) \& *_2(y) = x \&^* y$. Hence, we have

$$x \&^* y \leq z \quad \text{iff} \quad *_2(y) \preceq_2 z \nwarrow_* *_1(x)$$

From (3) and (4) we obtain that

$$*_2(y) \preceq_2 z \nwarrow_* *_1(x) \quad \text{implies} \quad *_2\,(y) \preceq_2 *_2(z \nwarrow_* *_1(x))$$

and due to (2) we have

$$*_2(y) \preceq_2 *_2(z \nwarrow_* *_1(x)) \quad \text{implies} \quad *_2\,(y) \preceq_2 z \nwarrow_* *_1(x)$$

Therefore, we have

$$x \,\&^*\, y \leq z \quad \text{iff} \quad *_2\,(y) \preceq_2 *_2(z \nwarrow_* *_1(x)). \tag{13}$$

Analogously, we obtain

$$*_1(x) \,\&\, *_2(y) \leq z \quad \text{iff} \quad *_2\,(y) \preceq_2 *_2(z \nwarrow *_1(x)) \tag{14}$$

By setting $y = (z \nwarrow *_1(x))$, in Equation (13), and $y = (z \nwarrow_* *_1(x))$, in Equation (15), we obtain equivalent inequalities $*_2(z \nwarrow *_1(x)) \preceq *_2(z \nwarrow_* *_1(x))$, $*_2(z \nwarrow *_1(x)) \succeq *_2(z \nwarrow_* *_1(x))$ respectively. Thus we have

$$*_2(z \nwarrow *_1(x)) = *_2(z \nwarrow_* *_1(x)).$$

Which is equal to (12). The first equation (11) can be obtained dually.

"$\Leftarrow$": Assume (12) holds true. By properties of adjointness, to show that $\&^*$ generates an adjoint triple we need to show that

$$R = \{y \mid *_1(x) \,\&\, *_2(y) \leq z\}$$

has a greatest element.

In the previous part, we proven that

$$*_1(x) \,\&\, *_2(y) \leq z \quad \text{iff} \quad *_2\,(y) \preceq_2 *_2(z \nwarrow *_1(x)) \tag{15}$$

hence $R = \{y \mid *_2(y) \preceq *_2(z \nwarrow *_1(x))\}$. Now, if R has no greatest element, i.e. $\bigvee R \notin R$, then we have $*_2(\bigvee R) \not\preceq *_2(z \nwarrow *_1(x))$ which is a contradiction with the assumption. By the contradiction we proved that R has a greatest element. □

Proposition 3. *Let $\mathbf{A} = (L_1, L_2, P, \&_1, \ldots, \&_n)$ be a multi-adjoint frame $*_1, *_2$ be hedges on L_1 and L_2, respectively. Let $\mathbf{A}' = (\mathrm{fix}(*_1), \mathrm{fix}(*_2), P, \&_1', \ldots, \&_n')$ s.t. $\&_1', \ldots, \&_n'$ are restrictions of $\&_1, \ldots, \&_n$ to $\mathrm{fix}(*_1) \times \mathrm{fix}(*_2)$, and $\phi_1' = *_1 \circ \phi_1, \phi_2' = *_2 \circ \phi_2$. Let $\mathbf{A}^* = (L_1, L_2, P, \&_1^*, \ldots, \&_n^*)$ be a multi-adjoint frame where $\&_i^*$ is defined by $a \,\&_i^*\, b = *_1(a) \,\&_i\, *_2(b)$, for all $i \in \{1, \ldots, n\}$, and the conditions in Lemma 4 are satisfied. Then $(\mathfrak{M}_{A'}, \preceq')$ and $(\mathfrak{M}_{A^*}, \preceq^*)$ are isomorphic.*

Proof. Let $\mathbb{K} = (A, B, R, \sigma)$ be a formal context, denote by ${}^{\uparrow}, {}^{\downarrow}$ concept-forming operators induced by $\mathbb{K}$ and $\mathbf{A}'$ and denote by ${}^{\Uparrow}, {}^{\Downarrow}$ concept-forming operators induced by $\mathbb{K}$ and $\mathbf{A}^*$. Furthermore, denote compositions $\psi_1 \circ *_1 \circ \phi_1$ and $\psi_2 \circ *_2 \circ \phi_2$ by $\bullet_1$ and $\bullet_2$ respectively.

For each mapping $g : B \to L$ we have

$$\begin{aligned}
\bullet_1(g^{\Uparrow}(a)) &= \bullet_1(\psi_1 \bigwedge\nolimits_1 (R(a,b) \swarrow^{*} \phi_2(g(b))))\\
&= \psi_1 *_1 (\phi_1\psi_1 \bigwedge\nolimits_1 (R(a,b) \swarrow^{*} (\phi_2(g(b)))))\\
&= \psi_1 \bigwedge\nolimits_1 *_1 (\bigvee\nolimits_1 \{x \mid *_1(x) \mathbin{\&} *_2(\phi_2(g(b))) \leq R(a,b))\})\\
&\overset{(\Delta)}{=} \psi_1 \bigwedge\nolimits_1 (\bigvee\nolimits_1 \{*_1(x) \mid *_1(x) \mathbin{\&} *_2(\phi_2(g(b))) \leq R(a,b))\})\\
&= \psi_1 \bigwedge\nolimits_1 (\bigvee\nolimits_1 \{x \in \mathrm{fix}(*_1) \mid x \mathbin{\&} *_2(\phi_2(g(b))) \leq R(a,b))\})\\
&= \psi_1 \bigwedge\nolimits_1 (R(a,b) \swarrow' *_2(\phi_2(g(b))))\\
&= \psi_1 \bigwedge\nolimits_1 (R(a,b) \swarrow' \phi_2\psi_2 *_2 (\phi_2 g(b)))\\
&= \psi_1 \bigwedge\nolimits_1 (R(a,b) \swarrow' \phi_2 \bullet_2 (g(b)))\\
&= (\bullet_2 \circ g)^{\uparrow}(a)
\end{aligned}$$

where (Δ) is due to Lemma 1 (6) and the fact that & generates adjoint triple and thus $\bigvee_1\{x \mid *_1(x) \mathbin{\&} *_2(\phi_2(g(b))) \leq R(a,b))\})$ has a greatest elements. Dually, we have $\bullet_2 \circ (f^{\Downarrow}) = (\bullet_1 \circ f)^{\downarrow}$ for each mapping $f : A \to L$. From that we have

$$g^{\uparrow} = \bullet_1 \circ (g^{\Uparrow}) \quad \text{and} \quad f^{\downarrow} = \bullet_2 \circ (f^{\Downarrow})$$

for each $g : B \to \mathrm{fix}(\bullet_2)$, $f : A \to \mathrm{fix}(\bullet_1)$. As a result of the previous equalities, we have that $\bullet_2$ is a surjective mapping $\mathrm{Ext}(\mathfrak{M}_{A^*}) \to \mathrm{Ext}(\mathfrak{M}_{A'})$ and $\bullet_1$ is a surjective mapping $\mathrm{Int}(\mathfrak{M}_{A^*}) \to \mathrm{Int}(\mathfrak{M}_{A'})$. In addition, for $g \in \mathrm{Ext}(\mathfrak{M}_{A^*})$ we have

$$\begin{aligned}
\bullet_2(g)^{\Uparrow}(a) &= \psi_1 \bigwedge\nolimits_1 R(a,b) \swarrow^{*} \phi_2\psi_2 *_2 \phi_2(g(b))\\
&= \psi_1 \bigwedge\nolimits_1 \bigvee\nolimits_2 \{x \mid *_1(x) \mathbin{\&} *_2 *_2 (\phi_2(g(b))) \leq R(a,b)\}\\
&= \psi_1 \bigwedge\nolimits_1 \bigvee\nolimits_2 \{x \mid *_1(x) \mathbin{\&} *_2(\phi_2(g(b))) \leq R(a,b)\}\\
&= \psi_1 \bigwedge\nolimits_1 R(a,b) \swarrow^{*} \phi_2(g(b)))\\
&= g^{\Uparrow}(a)
\end{aligned}$$

and dually $\bullet_1(f)^{\Downarrow} = f^{\Downarrow}$. Putting it together, we have $g = g^{\Uparrow\Downarrow} = \bullet_1(g^{\Uparrow})^{\Downarrow} = \bullet_2(g)^{\uparrow\Downarrow}$ showing that $\uparrow\Downarrow$ is injective; whence $\bullet_1, \bullet_2$ are bijections.

To show that $\bullet_1, \bullet_2$ are order-preserving let $\langle g_1, f_1\rangle, \langle g_2, f_2\rangle \in \mathfrak{M}_{A^*}$. An extent of $\langle g_1, f_1\rangle \wedge \langle g_2, f_2\rangle$ is equal to $\psi_2\phi_2(g_1 \wedge g_2)$ by the main Theorem in [9].

For $g_1, g_2 \in \mathrm{Ext}(\mathfrak{M}_{A^*})$ we have

$$\begin{aligned}
\bullet_2\psi_2\phi_2(g_1 \wedge g_2) &= \psi_2 *_2 \phi_2\psi_2\phi_2(g_1 \wedge g_2)\\
&= \psi_2 *_2 \phi_2(g_1 \wedge g_2)\\
&= \psi_2 *_2 \phi_2 \bullet_2 (g_1 \wedge g_2)\\
&\overset{(\Delta)}{=} \psi_2\phi_2'(\bullet_2(g_1) \wedge \bullet_2(g_2))
\end{aligned}$$

Equality $(\varDelta)$ is due to Corollary 1(b) since note that g_1, g_2 are fixpoints of $\psi_2 \circ \phi_2$.

Now, let $g_1, g_2 \in \mathrm{Ext}(\mathfrak{M}_{A'})$. We have

$$\begin{aligned}
(\psi_2 \phi_2'(g_1 \wedge g_2))^{\uparrow\Downarrow} &= (\psi_2 *_2 \phi_2(g_1 \wedge g_2))^{\uparrow\Downarrow} \\
&= (\bullet_1(g_1 \wedge g_2))^{\uparrow\Downarrow} \\
&= (\bullet_1(g_1^{\uparrow\downarrow} \wedge g_2^{\uparrow\downarrow}))^{\uparrow\Downarrow} \\
&= (\bullet_1(\bullet_1(g_1^{\uparrow\Downarrow}) \wedge \bullet_1(g_2^{\uparrow\Downarrow}))^{\uparrow\Downarrow} \\
&\overset{(\varDelta)}{=} (\bullet_1 \bullet_1 (g_1^{\uparrow\Downarrow} \wedge g_2^{\uparrow\Downarrow}))^{\uparrow\Downarrow} \\
&= (\bullet_1(g_1^{\uparrow\Downarrow} \wedge g_2^{\uparrow\Downarrow}))^{\uparrow\Downarrow} \\
&= (g_1^{\uparrow\Downarrow} \wedge g_2^{\uparrow\Downarrow})^{\Uparrow\Downarrow} \\
&\overset{(\nabla)}{=} \psi_2\phi_2(g_1^{\uparrow\Downarrow} \wedge g_2^{\uparrow\Downarrow})
\end{aligned}$$

Equality $(\varDelta)$ is due to Corollary 1(b) since g_1, g_2 are fixpoints of $\psi_2 \circ \phi_2$; equality (∇) is due to [9, Lemma 21].

This proves that $\bullet_1, \bullet_2, \uparrow\Downarrow$, and $\downarrow\Uparrow$ are order-preserving. $\square$

Example 1. Consider the multi-adjoint frame depicted in Fig. 1 (structures are the same as in [9, Example 3 (Fig. 2)] (where all $\&_i$'s coincide). Figure 2 depicts a formal context with two objects and two attributes, together with their associated multi-adjoint concept lattice.

Concept lattices with truth-stressing hedges

In this part, we follow the way in which the hedges are used in [4], i.e. we generalize concept-forming operators using intensifying hedges. Then we show how this is related to the theory described above.

We define the concept-forming operators as follows

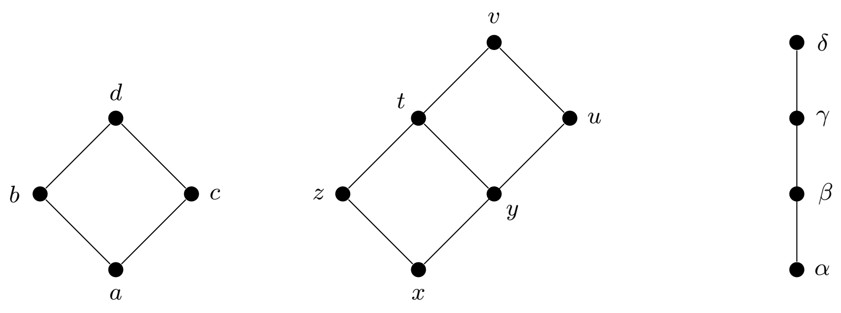
$$\begin{aligned}
g^{\vartriangle}(a) &= \psi_1 \bigwedge\nolimits_{1 b \in B} R(a, b) \swarrow *_2(\phi_2(g(b))), \\
f^{\triangledown}(b) &= \psi_2 \bigwedge\nolimits_{2 a \in A} R(a, b) \nwarrow *_1(\phi_1(f(a))).
\end{aligned}$$

Note that this is not strictly the same approach as used in Proposition 3 since $\swarrow$ and $\nwarrow$ are residua of the original adjoint operators $\&$, not the altered operators $\&^*$. In fact, generally there is no base operation $\&$ such that $(\cdot) \swarrow *_2(\cdot)$ and $(\cdot) \nwarrow *_1(\cdot)$ are its residua, since we do not generally have

$$x \le z \swarrow *_2(y) \text{ iff } y \le z \nwarrow *_1(x)$$

for each $x \in L_1, y \in L_2, z \in L$.

Lemma 5. *Assume* $(\&, \swarrow, \nwarrow)$ *is an adjoint triple,* $*_1, *_2$ *are intensifying hedges, and* $\swarrow^{\diamond}, \nwarrow_{\diamond}$ *being defined as* $z \swarrow^{\diamond} y = z \swarrow *_2 y$, *and* $z \nwarrow_{\diamond} x = z \nwarrow *_1 x$; *then*



	a	b	c	d
ψ_1	x	t	u	v

	α	β	γ	δ
ψ_2	x	y	t	v

	x	y	z	t	u	v
ϕ_1	a	b	a	b	c	d

	x	y	z	t	u	v
ϕ_2	α	β	γ	γ	δ	δ

&	α	β	γ	δ
a	x	x	x	x
b	x	y	v	v
c	x	y	y	t
d	x	y	v	u

$\swarrow$	α	β	γ	δ
t	d	d	c	c
u	d	d	d	d
v	d	d	d	b
x	d	a	a	a
y	d	d	c	a
z	d	a	a	a

$\nwarrow$	a	b	c	d
t	δ	β	δ	β
u	δ	δ	δ	δ
v	δ	δ	γ	γ
x	δ	α	α	α
y	δ	β	γ	β
z	δ	α	α	α

Fig. 1. L_1 (top left), L (top middle), L_2 (top right), connection operators $\phi_1, \phi_2, \psi_1, \psi_2$ (middle), adjoint triple ($\langle \&, \nwarrow, \swarrow \rangle$) (bottom).

	1	2
1	u	v
2	v	y

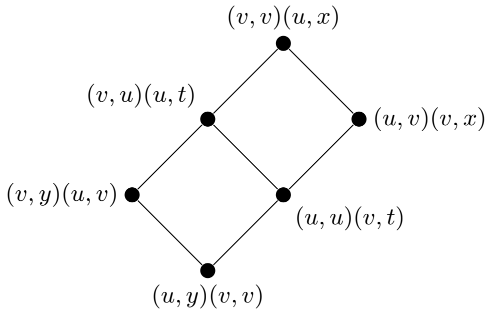


Fig. 2. Multi-adjoint formal context with two objects and two attributes (left) and the multi-adjoint concept lattice associated to the context (right).

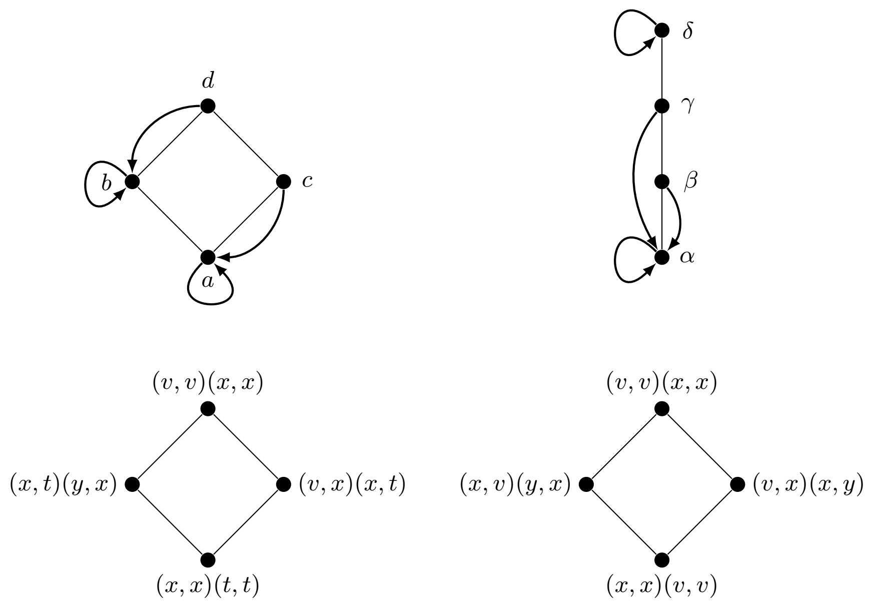


Fig. 3. Intensifying hedge on L_1 (top left) and L_2 (top right); concept lattices $\mathfrak{M}_{A'}$ (bottom left), $\mathfrak{M}_{A^*}$ (bottom right) of the formal context in Fig. 2; labels of nodes of $\mathfrak{M}_{A'}$ and $\mathfrak{M}_{A^*}$ represent characteristic vectors of corresponding extents and intents.

$\swarrow^{\diamond}, \nwarrow_{\diamond}$ are part of an adjoint triple with conjunctor $\&^{\diamond}$ if and only if for all x, y the following equality holds

$$x \,\&\, *_2(y) = *_1(x) \,\&\, y$$

and, in this case the previous value is the definition of $\&^{\diamond}$.

Proof. For all x, y, z, on the one hand, we have

$$*_1(x) \,\&\, y \leq z \quad \text{iff} \quad y \preceq_2 z \nwarrow *_1(x) \quad \text{iff} \quad y \preceq_2 z \nwarrow_{\diamond} x.$$

On the other hand, we have

$$x \,\&\, *_2(y) \leq z \quad \text{iff} \quad x \preceq_1 z \swarrow *_2(y) \quad \text{iff} \quad x \preceq_1 z \swarrow^{\diamond} y.$$

Thus we have

$$y \preceq_2 z \nwarrow_{\diamond} x \quad \text{iff} \quad x \preceq_1 z \swarrow^{\diamond} y$$

is equivalent to

$$x \,\&\, *_2(y) \leq z \quad \text{iff} \quad *_1\,(x) \,\&\, y \leq z,$$

which is equivalent to $x \,\&\, *_2(y) = *_1(x) \,\&\, y$. $\square$

However, the concept-forming operators $\vartriangle, \triangledown$ are in one-to-one correspondence with concept-forming operators $\uparrow, \downarrow$ with restrictions of L_1 and L_2 to subsemilattices $\text{fix}(*_1)$ and $\text{fix}(*_2)$:

$$\begin{aligned}
\bullet_1(g^{\vartriangle}(a)) &= \bullet_1(\psi_1 \textstyle\bigwedge_{b \in B} R(a,b) \swarrow *_2\phi_2(g(b))) \\
&= \psi_1 *_1 (\textstyle\bigwedge_{1\, b \in B} R(a,b) \swarrow *_2(\phi_2(g(b)))) \\
&= \psi_1 \textstyle\bigwedge_{1\, b \in B} *_1\,(R(a,b) \swarrow *_2(\phi_2(g(b)))) \\
&= \psi_1 \textstyle\bigwedge_{1\, b \in B} *_1 \bigvee_1 \{x \mid x \,\&\, *_2(\phi_2(g(b))) \leq R(a,b)\} \\
&= \psi_1 \textstyle\bigwedge_{1\, b \in B} \bigvee_1 \{*_1(x) \mid x \,\&\, *_2(\phi_2(g(b))) \leq R(a,b)\} \\
&\overset{(\Delta)}{=} \psi_1 \textstyle\bigwedge_{1\, b \in B} \bigvee_1 \{*_1(x) \mid *_1(x) \,\&\, *_2(\phi_2(g(b))) \leq R(a,b)\} \\
&= \psi_1 \textstyle\bigwedge_{1\, b \in B} \bigvee_1 \{x \in \text{fix}(*_1) \mid x \,\&\, *_2(\phi_2(g(b))) \leq R(a,b)\} \\
&= \psi_1 \textstyle\bigwedge_{1\, b \in B} \bigvee_1 \{x \in \text{fix}(*_1) \mid x \,\&\, \phi_2'(g(b)) \leq R(a,b)\} \\
&= \psi_1 \textstyle\bigwedge_{1\, b \in B} R(a,b) \swarrow \phi_2'(g(b)) \\
&= g^{\uparrow}(a)
\end{aligned}$$

where equality (Δ) holds because each x satisfying $x \,\&\, y \leq z$ satisfies $*_2(x) \,\&\, y \leq z$ as well; and because each $*_2(x)$ such that $*_2(x) \,\&\, y \leq z$ there is x' (explicitly, $*_2(x)$) with $*_2(x') = *_2(x)$ such that $x' \,\&\, y \leq z$. Dually, one can show $\bullet_2(f^{\triangledown}) = f^{\downarrow}$.

References

1. C. Alcalde, A. Burusco, R. Fuentes-González, and I. Zubia. The use of linguistic variables and fuzzy propositions in the L-fuzzy concept theory. *Computers & Mathematics with Applications*, 62(8):3111 – 3122, 2011.
2. E. Bartl, R. Belohlavek, and V. Vychodil. Bivalent and other solutions of fuzzy relational equations via linguistic hedges. *Fuzzy Sets and Systems*, 187(1):103 – 112, 2012.
3. R. Belohlavek and V. Vychodil. Fuzzy concept lattices constrained by hedges. *Journal of Advanced Computational Intelligence and Intelligent Informatics*, 11:536–545, 2007.
4. R. Belohlavek and V. Vychodil. Formal concept analysis and linguistic hedges. *Int. J. General Systems*, 41(5):503–532, 2012.
5. D. Dubois, F. D. de Saint-Cyr, and H. Prade. A possibility-theoretic view of formal concept analysis. *Fundamenta Informaticae*, 75(1-4):195–213, 2007.
6. G. Georgescu and A. Popescu. Concept lattices and similarity in non-commutative fuzzy logic. *Fundamenta Informaticae*, 53(1):23–54, 2002.
7. S. Krajči. A generalized concept lattice. *Logic Journal of IGPL*, 13(5):543–550, 2005.
8. Y. Lei and M. Luo. Rough concept lattices and domains. *Annals of Pure and Applied Logic*, 159(3):333–340, 2009.
9. J. Medina and M. Ojeda-Aciego. On multi-adjoint concept lattices based on heterogeneous conjunctors. *Fuzzy Sets and Systems*, 2012.
10. J. Medina, M. Ojeda-Aciego, and J. Ruiz-Calviño. Formal concept analysis via multi-adjoint concept lattices. *Fuzzy Sets and Systems*, 160(2):130–144, 2009.
11. Y. Y. Yao. Concept lattices in rough set theory. In *Proceedings of Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS'04)*, pages 796–801, 2004.

A contribution to semantic indexing and retrieval based on FCA - An application to song datasets

Víctor Codocedo[1]*, Ioanna Lykourentzou[1,2]**, and Amedeo Napoli[1]

[1] LORIA - CNRS - INRIA - Univesité de Lorraine, BP 239, 54506 Vandœuvre-les-Nancy.
victor.codocedo@loria.fr, amedeo.napoli@loria.fr,
[2] Centre de Recherche Public Henri Tudor - 29, avenue John F. Kennedy L-1855 Luxembourg-Kirchberg, Luxembourg
ioanna.lykourentzou@tudor.lu

Abstract. Semantic indexing and retrieval is an important research area, as the available amount of information on the Web is growing more and more. In this paper, we introduce an original approach to semantic indexing and retrieval based on Formal Concept Analysis. The concept lattice is used as a semantic index and we propose an original algorithm for traversing the lattice and answering user queries. This framework has been used and evaluated on a song dataset.

1 Introduction

Semantic indexing and retrieval refer to organizing a set of information items inside an index according to the semantic relations and concepts that they share, and then searching within this index to identify the items, the context of which matches a given user query [7, 15]. Semantic retrieval is based on flexible and partial matching techniques contrasting exact matching techniques. The organization and retrieval of information based on *context*, if effectively carried out, can significantly improve the understanding of information, as well as to provide users with richer and more meaningful search results, understanding *context* as a set of "external elements" which helps to understand or to manipulate information. For these reasons, context-based methods of classification and retrieval are applied on multiple types of information, ranging from text-based documents to multimedia content.

In the past years, Formal Concept Analysis (FCA [6]) has been applied to document indexation (an Information Retrieval task [10]) since it proposes a robust and formal framework to exploit the relations that documents (objects) have through the terms they share (attributes). For example, the work of Priss [13] uses concept lattices to improve the representation of a document collection by merging it with information from thesauri and thus creating a multi-faceted extended context. In a similar approach, the work of Carpineto et al. [4] presents CREDO, a system that queries Google to construct a faceted browser from a concept lattice to help the user on its search experience.

* Part of the Quaero Programme, funded by OSEO, French State agency for innovation.
** Supported by the National Research Fund, Luxembourg, and cofunded under the Marie Curie Actions of the European Commission (FP7-COFUND).

Different from browsing support and automatic facet construction, some approaches use the concept lattice directly as a document index and propose different strategies to explore it in order to find those documents relevant for a given user query. In general, given a document-term concept lattice and a conjunctive user query, these approaches work by identifying a formal concept in the lattice that best represents the query. This formal concept is then denominated as the "query concept". Two strategies can be distinguished to explore the concept lattice in order to retrieve documents: the work in [2] proposes a neighbourhood expansion strategy where concepts are ranked according to the minimal distance they have from the "query concept". In [11], the strategy used is the exploration of the super-hierarchy (super-concepts) of the "query concept". Other approaches, like the system FooCA presented in [8], while based on the FCA framework, do not rely on the lattice structure for an automated document retrieval. As described in [3], there are few works in the area of concept lattice-based information retrieval, and even though the first approaches were presented more than 40 years ago [18], the state-of-the-art remains little explored.

In this paper we propose a semantic indexation and retrieval technique supported over the FCA framework. It is based on the basic general idea of constructing a document-term concept lattice and identifying a "query concept", where we propose a novel exploration strategy based on the notion of "cousin concepts". We illustrate this technique using a song dataset, where songs are indexed using the terms appearing in their lyrics (songs are documents). The specific goal of this approach is to retrieve relevant songs to a user based on the terms provided in his query using a concept lattice as a semantic index. In order to enrich the song descriptions we use Wordnet (an external knowledge source which describes semantic relations among terms). As an additional characteristic we address the problem of semantic indexing and retrieval as a 3-step knowledge discovery on databases (KDD) process, comprising three main steps: data preparation, discovery and filtering of the results.

The main contributions of this work are the following:

- The notion of cousin concepts.
- Highlighting the capabilities of using FCA and a concept lattice as a semantic index and proposing a novel algorithm that traverses the lattice to retrieve information based on the content of the concepts.
- Proposing the incorporation of semantic indexing to current song retrieval systems and providing initial results, as a proof-of-concept of the potential that such an application can have.

We have selected songs indexing as an application domain since few work have been done in content-based indexation using lyrics. To the authors knowledge, the work in this area mostly focus on using low-level features of songs [19, 5, 9] (such as bit-rate, authors name, length, etc.) while high-level features (such as the semantics of the lyrics) have been used mainly for sentiment analysis classification [12, 17].

The rest of this paper is organized as follows. Firstly, Section 2 introduces the use of FCA for semantic indexing and querying and present our approach according to the main steps of KDD. Next, Section 3 presents the evaluation results obtained for our approach. Finally, Section 4 presents a discussion of the extension of our work and the conclusions of our research.

Table 1: Synsets retrieved for the word "soldier"from Wordnet

Synset	Synonyms	Definition
soldier.n.01	soldier	an enlisted man who serves in an army
soldier.n.02	soldier	a wingless sterile ant or termite having...
soldier.v.01	soldier	serve as a soldier in the military

2 Semantic indexing based on FCA

The problem addressed in this paper can be defined as finding songs the lyrics of which are *related* to a set of user-provided keywords, through a sufficient "closeness of meaning". Subsequently, our goal is to construct a semantic index, from a given set of songs and their lyrics and a relation which will successfully support the context-based song retrieval.

To address the above we first constructed a dataset of songs, where each song is related to a number of semantic meanings as follows. Two sources of data, namely musiXmatch and WordNet were used. MusiXmatch is a lyrics database, recently released as part of the MillionSong Dataset [1]. It was used to provide the lyrics for each song, in the form of a bag-of-words, already preprocessed to eliminate morphological word duplications. WordNet[3] is a well-known semantic dictionary and it was used to associate every word in the lyrics of a song with a set of synonym terms, called synsets, where each synset corresponds to one specific meaning of the word. Synsets also have semantic distances to one another, based on their position within WordNet's semantic hierarchy. The created dataset includes 357 songs, where each song is represented by its title, lyrics (in the form of a bag-of-words) and each word of the lyrics is connected to a set of synsets.

In the following, a song s_i is defined as a pair $\{t_i, L_i\}$ where t_i denotes the title and L_i the lyrics of the song in the form of a bag-of-words, as provided by musiXmatch.

2.1 Task 1: Lyrics Annotation

Given a song $s_i = \{t_i, L_i\}$, let $synsets(L_i)$ be the list of WordNet synsets that are associated with each word w_j in the lyrics L_i. Each retrieved synset has a definition and a set of synonyms. Word w_j is part of the synonyms. As an example, Table 1 illustrates the synsets retrieved from WordNet for the word "Soldier".

From the above example it can be understood that not every synset retrieved through WordNet is valuable in the context of a song. For instance, in the context of a war, a reference to a "soldier" would be clearly related to the definition of an *enlisted man who serves in an army* and not to the definition of a *soldier ant*. The third definition can also be disregarded since it is associated with a verb.

Therefore, to accurately annotate each song, we need to keep only those synsets that correspond to the actual context of the song.

[3] `http://wordnet.princeton.edu`

To address this, a filtering process took place as follows: a well-known similarity metric, namely the Wu-Palmer Similarity Measure [20], was used to measure the semantic similarity between every pair of synsets in the $synsets(L_i)$ set. The Wu-Palmer similarity measure $wp(ss_1, ss_2) = [0,1], ss_1 \in synsets(L_i) \wedge ss_2 \in synsets(L_i)$ is provided by the WordNet API and measures semantic similarity using path distance and the difference of levels in the synset tree. Then for each synset ss_j we calculate the average distance with every other synset $ss_k \in synsets(L_i)$ as defined in equation 1.

$$avg\_sim(ss_j) = \frac{\sum\limits_{j \neq k} wp(ss_j, ss_k)}{|synsets(L_i)|} \tag{1}$$

The synset with the lowest avg_sim is deleted from $synsets(L_i)$. The filtering is repeated until we reach to a threshold of 20 most similar elements in $synsets(L_i)$, which will be considered to constitute the so-called *semantic core* of the song s_i, denoted as $core(s_i)$. The threshold of 20 synsets was selected heuristically, since it was found to represent the semantic core of the songs, in the specific dataset, in a concise and non redundant way. As an example, Table 2 shows some of the synsets selected to describe the song titled "The Green Beret Balad". As it may be observed, the synsets that do not belong to the song's context (which is mostly about the notions of war, battle, man, etc.) will be less related to the rest of the song's synsets, and therefore they will be more likely to be omitted. In this example, the synset "soldier.n.01: an enlisted man or woman who serves in an army" has been selected, instead of the other WordNet's alternative "soldier ant: soldier.n.02: a wingless sterile ant or termite having a large head and powerful jaws adapted for defending the colony".

Table 2: Synsets describing the "Green Beret Balad" song

Synset	Mean Similarity	Definition
serviceman.n.01	0.665785412147	someone who serves in the armed forces
young.n.01	0.652204776648	any immature animal
man.n.03	0.652204776648	the generic use of the word to refer to any human being
soldier.n.01	0.629664596273	an enlisted man or woman who serves in an army
brave.n.01	0.622338466487	a North American Indian warrior
green_beret.n.01	0.606135516657	a soldier who is a member of the United States...
back.n.04	0.596281910309	(football) a person who plays in the backfield
wing.n.06	0.596281910309	a hockey player stationed in a forward position on either side
son.n.01	0.594639167088	a male human offspring
wife.n.01	0.594639167088	a married woman; a man's partner in marriage
valet.n.01	0.569009183037	a manservant who acts as a personal attendant to his employer

The outcome of the filtering process is the set $core(s_i)$, which is considered as the final set of semantic annotations of the song s_i since it refers to a well-defined semantic schema, i.e. WordNet, where each annotation contains a definition and relations with other annotations.

2.2 Task 2: Semantic Index Creation

We built the semantic song index as a concept lattice using Formal Concept Analysis (FCA) following the lines of [11, 4, 13]. The basics of FCA are introduced in [6].

In the formal context of songs considered in this paper $\mathcal{K} = (G, M, I)$, the set of objects G contains the songs of the dataset while the set of attributes M contains all the WordNet synsets included in the semantic cores of the songs in G. The set I contains the relations gIm which stand for "song g has synset m in its semantic core". Table 3 shows an example of a formal context created from 11 songs and 6 synsets. The concept lattice obtained from this example context is illustrated in Figure 1. The concept lattice is presented in its *reduced notation* where objects (songs) and attributes (synsets) are shown only next to their object/attribute-concept, i.e. the most general concept introducing the attribute (which is inherited from higher to lower levels), the most specific concept having the object in its extent (the object being shared from lower to higher levels).

	bolshevik.n.01	son.n.01	man.n.03	white.n.01	serviceman.n.01	buddy.n.01
song1		x	x		x	
song6						x
song10		x	x		x	
song14		x	x		x	
song16	x	x	x	x	x	
song18		x	x		x	
song24			x		x	
song27	x	x				x
song32		x	x		x	x
song33			x	x		
song39	x		x	x	x	x

Table 3: Formal context example.

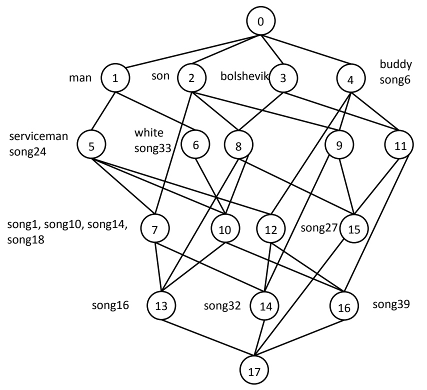


Fig. 1: The semantic index as a concept lattice obtained through FCA. Each concept is labelled with a unique identifier.

2.3 Task 3: Semantic Index querying

A simple query to the constructed semantic index (i.e. the concept lattice) is a pair $q = (A_q, B_q)$ where A_q denotes an empty extent to be filled and $B_q = \{ss\}$ is a synset to be searched for. Actually, the retrieval is based on two steps. The first one corresponds to "exact matching" (as in [11]) and the second corresponds to "partial matching" based on the cousin relation introduced hereafter. The first step consists of searching within the concept lattice for the attribute-concept (A_{ss}, B_{ss}) of attribute ss, i.e. finding the most general concept where ss appears in an intent (also denoted by $\mu(ss)$ in [6]). The extent of A_{ss} contains the list of all songs which are directly associated with the synset ss. This *direct answer* constitutes only a part of the answer. The second step is related to partial matching based on the cousin relation defined in the following.

Definition of cousin concepts. Two concepts (A_1, B_1) and (A_2, B_2) which are not comparable for $\leq_{\mathcal{K}}$ are said to be *cousins* iff there exists $(A_3, B_3) \neq \perp$ such that $(A_3, B_3) \leq_{\mathcal{K}} (A_1, B_1)$ and $(A_3, B_3) \leq_{\mathcal{K}} (A_2, B_2)$ and $d_{\mathcal{K}}((A_2, B_2), (A_3, B_3)) = 1$ (or $d_{\mathcal{K}}((A_1, B_1), (A_3, B_3)) = 1$), where $\perp$ is the bottom concept and $d_{\mathcal{K}}$ measures the minimal distance between two formal concepts in the lattice $\mathcal{K}$. Intuitively, this means that (A_1, B_1) and (A_2, B_2) do not subsume each other and that (A_3, B_3) can be either the lower bound or be subsumed by the lower bound $(A_1, B_1) \sqcap (A_2, B_2)$ (where $(A_1, B_1) \sqcap (A_2, B_2)$ denotes the lower bound of (A_1, B_1) and (A_2, B_2). Actually, (A_3, B_3) represents songs related to both (A_1, B_1) and (A_2, B_2): two songs are related if their semantic cores share some elements, which is the case here, as $A_3 \subseteq A_1 \cap A_2$ and $B_3 \subseteq B_1 \cup B_2$. For example, in Figure 2, concept 3 is a cousin of 2 because of concept 8, concept 11 is a cousin of concept 12 because of concept 16 and so on.

For a given attribute concept (A_{ss}, B_{ss}), the querying algorithm traverses the lattice to extract all *cousin concepts* (A_i, B_i) of the synset ss, and then it moves down the concept lattice, repeating the same extraction level by level. It should be noticed that the original synset query ss is not present in any of the intents of the *cousin concepts* B_i, this is why we can speak of "partial matching". Every cousin concept (A_i, B_i) is ranked according to the intersection that its extent has with the extent of the original attribute concept using the following metric:

$$rank(A_i, A_{ss}) = \frac{|A_i \cap A_{ss}|}{|A_i|} \tag{2}$$

This metric is two-fold since it allows the detection of concepts (A_i, B_i) which are far from the original concept and share no common objects with the extent of (A_{ss}, B_{ss}) ($A_i \cap A_{ss} = \emptyset$ and rank = 0) and those that are too abstract and describe too many objects ($|A_i| \gg |A_{ss}|$ and rank $\sim$ 0).

Hereafter, we give details on the steps of the querying algorithm through the use of an example, graphically illustrated in Figure 2. Let us consider a user query for songs related to the synset "bolshevik.n.01" (concept 3 on Figure 2).

1. Find the attribute-concept (A_{ss}, B_{ss}) for the synset $\{ss\}$: concept 3.
2. Find the sub-hierarchy of (A_{ss}, B_{ss}) in the concept lattice, i.e. all concepts subsumed by (A_{ss}, B_{ss}) and order them by levels: concepts 8, 11, 10, 15, 13, 16, 17 (solid arrows in Figure 2).
3. For each concept in this sub-hierarchy, find the super-concepts which are cousin concepts of (A_{ss}, B_{ss}) (and then for the descendants of (A_{ss}, B_{ss})): concept 2 is a cousin of 3 because of 8, 4 is a cousin of 3 because of 11, 6 is a cousin of 8 because of 10, etc. The final list is ordered by levels: concepts 2, 4, 5, 6, 9, 7, 12, 14 (dashed arrows in Figure 2).
4. Calculate the rank value of each cousin concept (according to Eq. 2) and sort these cousin concepts in descending order: concepts 6, 12, 9, 4, 2, 5, 7, 14.

5. The result is composed of the songs in the extent of the attribute-concept (A_{ss}, B_{ss}) and the extents of the cousin concepts.

The final result, in terms of the retrieved cousin concepts, their rankings and the songs in their extents, is shown in Table 4.

Concept	$rank$	Songs
3 (AC)		16,26,39
6	66%	33
12	50%	32
9	50%	
4	50%	
2	29%	1,10,14,18
5	25%	24
7	17%	
14	0%	

Table 4: Ranked list of retrieved songs for query "bolshevik.n.01"

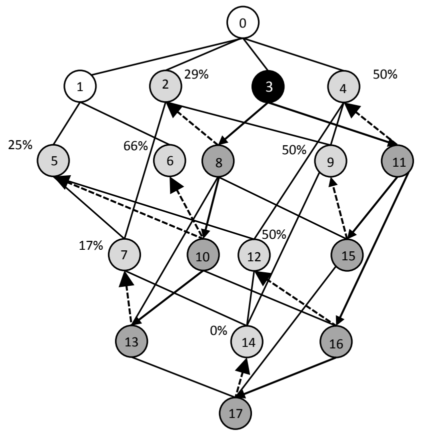


Fig. 2: Querying the semantic index. Starting from attribute concept 3, bold arrows show subhierarchy, dashed arrows show cousin concepts depicted next to their ranking value.

It can be noticed that the basic target of the algorithm proposed above is semantic retrieval. The order in which the algorithm presents the retrieved groups of songs to the user is a "recommendation decision", which could depend on user preferences and imply the use of a threshold to filter the final list of songs. Semantic retrieval does not necessarily imply the use of a threshold, which is why, for the scope of this paper, we present all the songs retrieved.

3 An application to song retrieval

As described in the previous section, the lattice is queried using a synset ss (e.g. *bolshevik.n.01*). This synset is directly related to a set of songs ($direct\_answer_{ss}$), i.e. those which are in the extent of the attribute concept (A_{ss}, B_{ss}) of that synset (in the case of *bolshevik.n.01*, songs 16, 27 and 39). These songs will be retrieved along with the set of songs found in the extents of the *cousin concepts* ($indirect\_answer_{ss}$) of (A_{ss}, B_{ss}) (songs 1, 10, 14, 18, 24, 32 and 33).

Regarding the songs in $direct\_answer_{ss}$, we are interested in examining whether our approach can find them if we apply it on a modified formal context where their relations with the synset ss have been eliminated. Of course, in this case, these songs cannot be retrieved as directly related songs, but only as songs found in the extents of *cousin concepts*. For example, if we eliminate the relation between song 16 and the synset *bolshevik.n.01* we want to know if this song can be retrieved by querying the new lattice using the synset *bolshevik.n.01*.

Regarding the set $indirect\_answer_{ss}$, we are interested in examining how it changes after the application of our approach on the modified formal context since the elimination of a *(song,synset)* relation will affect the structure of the concept lattice and hence the output of the proposed retrieval algorithm. Small variations in the content of this set will indicate robustness.

3.1 Leave-one-out cross validation, precision and recall

To evaluate the above and subsequently the definition of cousin concepts presented in section 2.3 we used and adapted the leave-one-out cross validation (LOOCV) methodology, which is a special type of cross validation [14]. Our adaptation consists of intentionally removing a single *(song,synset)* relation from a formal context (hereafter referred to as the *primary formal context*) and constructing its associated concept lattice (i.e. removing a cross from the primary formal context). The modified formal context is called a *scenario*. Therefore, each scenario is identified by a pair synset (ss_{scn}) and song (s_{scn}), the relation of which was eliminated for the scenario's construction.

For a given scenario, if song s_{scn} can be retrieved by querying for synset ss_{scn} we mark the scenario as *successful* (e.g. querying for *bolshevik.n.01* and retrieving song 16 for scenario with $ss_{scn} =$ *bolshevik.n.01* and $s_{scn} =$ song 16). The number of *successful* scenarios for a synset ss is given by $successful\_scenarios(ss)$. The *total number of scenarios* for a synset ss (given by $total\_scenarios(ss)$) is determined by the number of songs where the synset appears in (i.e. the number of crosses on the synset column in the primary formal context), since we only eliminate at each time a single *(song,synset)* relation from the primary formal context (e.g. for the synset *bolshevik.n.01* we construct 3 scenarios for songs 16, 27 and 39). We can then define $success\_rate(ss)$ as illustrated in equation 3.

$$success\_rate(ss) = \frac{successful\_scenarios(ss)}{total\_scenarios(ss)} \tag{3}$$

Because the relation between a song and a synset is eliminated in a given scenario, the song cannot be retrieved by the "exact matching" of the synset ss (the song will not be contained in $direct\_answer_{ss}$). Instead, the method proposed using cousin concepts has to be used and the song should be found through "partial matching". The test consist on observing if the song can be found in $indirect\_answer_{ss}$. Hence, the measure of *success_rate* represents the tendency of how related are those songs found through "partial matching" with the query and the usefulness of the proposed method.

To evaluate the changes in the set $indirect\_answer_{ss}$, we compare the full set of retrieved songs from each scenario with the respective set of songs retrieved from the primary concept lattice (i.e. the concept lattice constructed from the primary formal context). We calculate precision defined in Eq. 4 as the proportion of true positives over the retrieved list of songs in a scenario. Accordingly, we calculate recall, in Eq. 5, as the proportion of true positives over the retrieved list of songs in the primary concept lattice, i.e. the concept lattice without any removal. The expression $Ret(scn, ss)$ denotes the total set of songs retrieved from scenario scn ($direct\_answer_{ss} \cup indirect\_answer_{ss}$) querying for synset ss while $primary$ denotes the primary concept lattice.

$$precision(scn, ss) = \frac{Ret(scn, ss) \cap Ret(primary, ss)}{Ret(scn, ss)} \tag{4}$$

$$recall(scn, ss) = \frac{Ret(scn, ss) \cap Ret(primary, ss)}{Ret(primary, ss)} \tag{5}$$

3.2 Results

For each synset we calculate the mean precision and recall from all their scenarios. From our test set of 357 songs and 1848 synsets we selected 192 synsets and simulated 1027 scenarios (working with approximately 1000 scenarios allows a lower volume of computation and more different trials). Table 5 shows the values of these measures for 10 synsets. For example, it can be seen that synset anteroom.n.01 has relations with 4 songs. A *success_rate* of 1 means that all simulations were successful. Recall of 0.9 and Precision of 0.96 mean that for an elimination of 25% of the relations for the synset (1 over 4 songs), still 90% of the information was retrieved and 96% of the information was correct. There is a positive relation between the number of songs in which the synset appears and the *success_rate* measure. This is not strange since synsets appearing in a few songs will be in fewer concepts in the lattice and hence the simulation affects them in the worst manner. For example, for the synset bar.n.03, the elimination of one relation with a song leads to the elimination of 50% of its relations (1/2), while for the synset battle.n.01 the elimination of one relation with a song leads to the elimination of only 5% of its relations (1/20).

Figure 3 shows the distribution of *success_rate*, recall and precision (in the interval of $[0, 1]$ in axis y) over the number of songs where synsets appear in (in axis x). The *success_rate* maintains a growing tendency showing that better results are obtained with synsets which appear in a greater number of songs. In a wider sense, precision and recall maintain their values over 70% over all the samples. This is especially important in values of songs per synset below 5 since losing a single connection could disconnect songs more significantly. In the case of the first point (2.5 songs per synset) losing one connection means losing 40% of the connections of the attribute concept, however over 70% of the original set of songs is retrieved.

It should be noticed that a certain degree of bias, caused by the inclusion of the directly related songs in the measures of precision/recall, is to be expected. That is, given that for each scenario we are eliminating only one *(song, synset)* relation, the remaining directly related songs will be present in both sets retrieved when querying the scenario and the primary concept lattice. Therefore, the precision/recall measures are meant to be used, in the context of this paper, as a means of examining how the set of cousin concepts is affected for each synset, and they should not be considered as a medium of comparison with other information retrieval approaches.

Finally, even if more experiments have to be completed, we can conclude that the *definition of cousin concepts* is valuable and allows the use of a concept lattice as a semantic index to retrieve objects not directly related to a query.

synset	songs	success_rate	recall	precision
anteroom.n.01	4	1.0	0.9	0.964
bustle.n.01	3	0.333	0.564	0.611
ambition.n.01	9	0.888	0.888	0.938
child.n.03	13	0.923	0.945	0.982
arrest.n.02	4	0.25	0.75	0.807
battle.n.01	20	0.9	0.956	0.989
champion.n.02	2	0.0	0.083	1.0
better.n.03	3	0.0	0.641	0.694
attack.n.01	2	1.0	0.730	0.791
bar.n.03	2	0.0	0.083	1.0

Table 5: Simulations results.

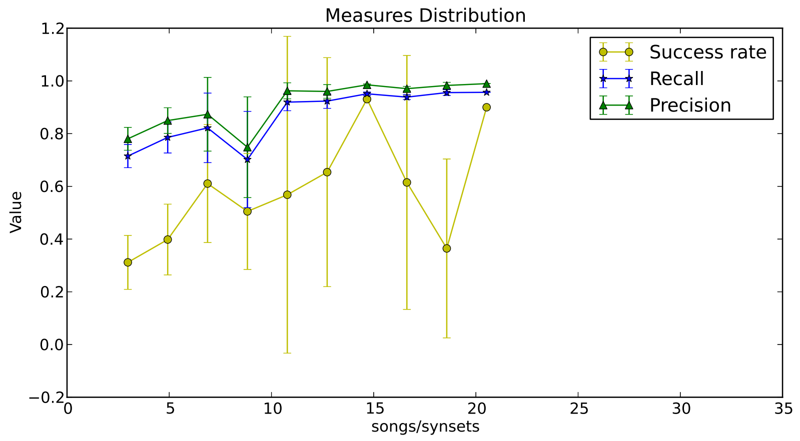


Fig. 3: Distribution of measures over songs per synset.

4 Discussion and Conclusion

In this paper we propose an approach for semantic indexing and retrieval of songs based on Formal Concept Analysis and exploiting the representation of a song's lyrics as a collection of relevant WordNet synsets.

A number of limitations may be found to the present work. First the evaluation focuses, at this stage, mainly on examining the robustness of the cousin concepts. This evaluation choice was made on the basis of two reasons: i) the definition of cousin concepts is the core of the proposed approach and therefore its efficiency is the first parameter that needs to be evaluated and ii) the literature reports an absence of other approaches that perform indexing and retrieval based on the semantics of song lyrics, and which could be used as a comparison benchmark. Given the above, and in view of the positive results that the current evaluation has yielded, the next stage is to proceed with a user evaluation study, which will allow us to examine the relativeness of the retrieved results according to the intended meaning of specific users' queries.

A second limitation refers to the requirements of the proposed approach in case it is intended for a large-scale application. Specifically, at this stage a relatively small dataset of 357 songs was used, however additional experiments would be necessary to examine how the size of the dataset affects the performance of the approach. Another necessity refers to defining in more detail the way that a user query can be matched to a set of synsets, which can then be used to query the lattice-based song index.

Future work includes two directions: i) enriching the semantic song representation with other information resources such as DBpedia and ii) expanding the proposed approach to different data collections.

On the first direction, additional information about the meaning of the songs can be found in external user-contributed sources, such as Wikipedia and its semantic equivalent DBPedia. Therefore, as a first future direction we plan to enrich the constructed semantic song representations with categorical knowledge, i.e. regarding the broader "topic" that each song is about, from DBPedia. To take advantage of this categorical knowledge we plan to extend the proposed FCA-based approach with *Relational Concept Analysis* [16], in order to create a semantic index where songs are related not only through their lyrics but also through their categories.

A second potential extension we consider applying the proposed approach to other types of information content, such as scientific papers or news information, to examine its generalization capability and to facilitate comparison with other benchmark techniques of semantic search and retrieval.

As a conclusion, in this paper we propose a novel contribution to the field of semantic indexing and retrieval, which is based on Formal Concept Analysis. We use the concept lattice as a semantic index and propose a novel algorithm to traverse the lattice in order to match user queries with semantically relevant information items. The approach was tested on a song dataset and the obtained results show good capabilities.

References

1. Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere, 'The million song dataset', in *Proceedings of the 12th International Conference on Music Information Retrieval*, (2011).
2. Claudio Carpineto and Giovanni Romano, 'Order-theoretical ranking', *Journal of the American Society for Information Sciencies*, **51**(7), 587–601, (May 2000).
3. Claudio Carpineto and Giovanni Romano, *Concept Data Analysis: Theory and Applications*, July 2004.
4. Claudio Carpineto, Giovanni Romano, and Fondazione Ugo Bordoni, 'Exploiting the potential of concept lattices for information retrieval with credo', *Journal of Universal Computer Science*, **10**, 985–1013, (2004).

5. Zhouyu Fu, Guojun Lu, Kai Ming Ting, and Dengsheng Zhang, ‘A survey of audio-based music classification and annotation’, *Multimedia, IEEE Transactions on*, **13**(2), 303 –319, (april 2011).
6. Benrhard Ganter and Rudoph Wille, *Formal Concept Analysis*, Springer, Berlin, 1999.
7. Atanas Kiryakov, Borislav Popov, Ivan Terziev, Dimitar Manov, and Damyan Ognyanoff, ‘Semantic annotation, indexing, and retrieval’, *Web Semantics: Science, Services and Agents on the World Wide Web*, **2**(1), 49 – 79, (2004).
8. Bjoern Koester, ‘Conceptual knowledge retrieval with fooca: Improving web search engine results with contexts and concept hierarchies’, in *Industrial Conference on Data Mining*, ed., Petra Perner, volume 4065 of *Lecture Notes in Computer Science*, pp. 176–190. Springer, (2006).
9. Mika Kuuskankare and Mikael Laurson, ‘Mir in enp - rule-based music information retrieval from symbolic music notation’, in *Proceedings of the 10th International Society for Music Information Retrieval Conference, ISMIR 2009*. International Society for Music Information Retrieval, (2009).
10. Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze, *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, USA, 2008.
11. Nizar Messai, Marie-Dominique Devignes, Amedeo Napoli, and Malika Smal-Tabbone, ‘Querying a bioinformatic data sources registry with concept lattices’, in *Proceedings of ICCS 2005*, pp. 323–336. LNCS 3596, Springer, (2005).
12. Michael Kai Petersen, Lars Kai Hansen, and Andrius Butkus, ‘Computer music modeling and retrieval. genesis of meaning in sound and music’, chapter Semantic Contours in Tracks Based on Emotional Tags, 45–66, Springer, (2009).
13. Uta Priss, ‘Lattice-based information retrieval’, *Knowledge Organization*, **27**, 132–142, (2000).
14. Payam Refaeilzadeh, Lei Tang, and Huan Liu, ‘Cross-validation’, in *Encyclopedia of Database Systems*, eds., Ling Liu and M. Tamer Özsu, 532–538, Springer US, (2009).
15. Fuji Ren and David B. Bracewell, ‘Advanced information retrieval’, *Electronic Notes in Theoretical Computer Science*, **225**(0), 303 – 317, (2009).
16. Mohamed Rouane-Hacene, Marianne Huchard, Amedeo Napoli, and Petko Valtchev, ‘A proposal for combining formal concept analysis and description logics for mining relational data’, in *Proceedings of ICFCA 2007*, pp. 51–65. LNAI 4390, Springer, (2007).
17. Govind Sharma and M. Narasimha Murty, ‘Mining sentiments from songs using latent dirichlet allocation’, in *Proceedings of the 10th international conference on Advances in intelligent data analysis X*, IDA’11, pp. 328–339. Springer, (2011).
18. Dagobert Soergel, ‘Mathematical analysis of documentation systems : An attempt to a theory of classification and search request formulation’, *Information Storage and Retrieval*, 129–173, (1967).
19. Rainer Typke, Frans Wiering, and Remco C. Veltkamp, ‘A survey of music information retrieval systems’, in *Proceedings of the 6th International Conference on Music Information Retrieval*, (2005).
20. Zhibiao Wu and Martha Palmer, ‘Verbs semantics and lexical selection’, in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, ACL ’94, pp. 133–138, Stroudsburg, PA, USA, (1994). Association for Computational Linguistics.

Efficient Vertical Mining of Minimal Rare Itemsets

Laszlo Szathmary[1], Petko Valtchev[2], Amedeo Napoli[3], and Robert Godin[2]

[1] University of Debrecen, Faculty of Informatics, Department of IT,
H-4010 Debrecen, Pf. 12, Hungary
`szathmary.laszlo@inf.unideb.hu`

[2] Dépt. d'Informatique UQAM, C.P. 8888,
Succ. Centre-Ville, Montréal H3C 3P8, Canada
{`valtchev.petko, godin.robert`}`@uqam.ca`

[3] LORIA (CNRS - Inria NGE - Université de Lorraine) BP 239, 54506
Vandœuvre-lès-Nancy Cedex, France
`napoli@loria.fr`

Abstract. Rare itemsets are important sort of patterns that have a wide range of practical applications, in particular, in analysis of biomedical data. Although mining rare patterns poses specific algorithmic problems, it is yet insufficiently studied. In a previous work, we proposed a levelwise approach for rare itemset mining that traverses the search space bottom-up and proceeds in two steps: (1) moving across the frequent zone until the minimal rare itemsets are reached and (2) listing all rare itemsets. As the efficiency of the frequent zone traversal is crucial for the overall performance of the rare miner, we are looking for ways to speed it up. Here, we examine the benefits of depth-first methods for that task as such methods are known to outperform the levelwise ones in many practical cases. The new method relies on a set of structural results that helps save a certain amount of computation and eventually ensures it outperforms the current levelwise procedure.

1 Introduction

Pattern mining is a basic data mining task whose aim is to uncover the hidden regularities in a set of data records, called *transactions* [1]. These regularities typically manifest themselves as repeating co-occurrences of properties, or *items*, in the transactions, i.e., item patterns. As there is a potentially huge number of patterns, quality measures are applied to filter only promising patterns, i.e., patterns of potential interest to the analyst.

Designing a faithful interestingness metric in a domain independent fashion is not realistic [2]. Indeed, without an access to the semantics of the items or another source of domain knowledge, it is impossible for the mining tool to assess the real value behind a pattern. As a simplifying hypothesis, the overwhelming majority of pattern miners chose to look exclusively on item combinations that are sufficiently frequent, i.e., observed in a large enough proportion of the transactions. This roughly translates the intuition that significant regularities should occur often within a dataset.

Yet such a hypothesis fails to reflect the entire variety of situations in data mining practice [3]. More precisely, it ignores some of the key factors for the success of the mining task, namely, the expectations of the analyst and further to that, her/his knowledge of the dataset and of the domain it stems from. Indeed, while an analyst with little or no knowledge of the dataset will most probably be happy with the most frequent patterns thereof, a better informed one may find them of little surprise and hence barely useful. More generally speaking, in some specific situations, frequency may be the exact opposite of pattern interestingness. The reason behind is that in these cases, the most typical item combinations from the data correspond to widely-known and well-understood phenomena, hence there is no point in presenting them to the analyst. In contrast, less frequently occurring pattern may point to unknown or poorly studied aspects of the underlying domain [3].

The above schematic description fits to a wide range of mining situations where biomedical data are involved. For instance, in pharmacovigilance, one is interested in associating the drugs taken by patients to the adverse effects the latter may present as a result (*safety signals* or *drug interactions* in the technical language of the field). To do that, a now popular way is to mine the databases of pharmacovigilance reports, where each individual case is thoroughly described, for such associations. However, as the data is accumulated throughout the years, the most frequent associations tend to translate well-known signals and interactions. The new and potentially interesting associations are less frequent and hence "hidden" behind these most often occurring combinations.

The problem of unraveling them is a non-trivial one: In [4], a method based on frequent pattern mining has been shown to only be able of dealing with a small proportion of the existing pharmacovigilance datasets. The main difficulty is that the data cannot be advantageously segmented as the new signals/interactions may appear in any record. Alternatively, the problem cannot be approached as outlier detection as a potential manifestation of a new signal need not have any exceptional characteristics. Moreover, in order for an association to be validated, it must occur in at least a given minimal number of patient records (typically, five). Yet mining all patterns with only this weak constraint results in an enormously-sized output whereby the overwhelming majority brings no new insights.

The conclusion we drew out of that study was that the not-as-frequent, or rare, patterns need to be addressed by specially designed algorithms rather than by standard frequent miners fed with lower enough support. Similar observations have been made in the pattern mining literature more than half a decade ago [3]. Since that time, a variety of methods that target non-frequent datasets have been published, most of them adapting the classical levelwise mining schema exemplified by the *Apriori* algorithm [1] to various relaxations of the frequent itemset and frequent association notions [5,6,7] (see [8] for a recent survey thereof).

In our own approach, we focus on limiting the computational effort dedicated to the traversal of irrelevant areas of the search space. In fact, as indicated above, the rare itemsets represent a band of the underlying Boolean lattice of all itemsets that is located "above" the frequent part thereof and "below" the exceptional part (itemsets that occur in a tiny number, possibly none, of transactions). Thus, in a previous paper [9], we proposed a bottom-up, levelwise approach that traverses the frequent zone of the search space either exhaustively or in a more parsimonious manner by listing uniquely frequent generator itemsets. We also provided a levelwise method for generating all rare itemsets up to the minimal frequency required by the task (could be one in the worst case).

In this paper we are looking for a more efficient manner for traversing the frequent part of the Boolean lattice. In fact, the rapidity of pinpointing the minimal rare itemsets turned out to be a dominant factor for the overall performance of the rare pattern miner. It is therefore natural to investigate manners to speed it up. Further to that idea, and breaking with the dominant levelwise algorithmic schema, we study a depth-first method. Indeed, depth-first frequent pattern miners have been shown to outperform breadth-first ones on a number of datasets. We therefore decided to check the potential benefits of the approach in the rare pattern case. To that end, we have shown a set of structural results that allows for a sound substitution within the overall rare pattern mining architecture. Our experimental results show that the new method is most of the time much faster than the previous one.

The main contribution of this paper is a new algorithm called *Walky-G* for mining minimal rare itemsets. The algorithm limits the traversal of the frequent zone to frequent generators only. This traversal is achieved through a depth-first strategy.

The remainder of the paper is organized as follows. We first recall the basic concepts of frequent/rare pattern mining and then summarize the key aspects of our own approach. Next, we present a set of structural results about the search space and the supporting structure of the depth-first traversal of the pattern space. Then, the depth-first frequent zone-traversal algorithm is described and its *modus operandi* illustrated. A comparative study of its performance to those of the current breadth-first methods is also provided. Finally, lessons learned and future work are discussed.

2 Basic Concepts

Consider the following 6×5 sample dataset: $\boldsymbol{\mathcal{D}} = \{(1,\ ABCDE), (2,\ BCD), (3,\ ABC), (4,\ ABE), (5,\ AE), (6,\ DE)\}$. Throughout the paper, we will refer to this example as **"dataset $\boldsymbol{\mathcal{D}}$"**.

Consider a set of *objects* or *transactions* $\mathcal{O} = \{o_1, o_2, \ldots, o_m\}$, a set of *attributes* or *items* $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$, and a relation $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$. A set of items is called an *itemset*. Each transaction has a unique identifier (*tid*), and a set of transactions is called a *tidset*. The tidset of all transactions sharing a given itemset X is its *image*, denoted by $t(X)$. The *length* of an itemset X is $|X|$, whereas an itemset of length i is called an i-itemset. The (absolute) *support* of an itemset X, denoted by $supp(X)$, is the size of its image, i.e. $supp(X) = |t(X)|$.

A lattice can be separated into two segments or zones through a user-provided "minimum support" threshold, denoted by *min_supp*. Thus, given an itemset X, if $supp(X) \geq min\_supp$, then it is called *frequent*, otherwise it is called *rare* (or *infrequent*). In the lattice in Figure 1, the two zones corresponding to a support threshold of 2 are separated by a solid line. The rare itemset family and the corresponding lattice zone is the target structure of our study.

Definition 1. X subsumes Z, *iff* $X \supset Z$ *and* $supp(X) = supp(Z)$ [10].

Definition 2. *An itemset* Z *is a* generator *if it has no proper subset with the same support.*

Generators are also known as free-sets [11] and have been targeted by dedicated miners [12].

Property 1. Given $X \subseteq \mathcal{A}$, if X is a generator, then $\forall Y \subseteq X$, Y is a generator, whereas if X is not a generator, $\forall Z \supseteq X$, Z is not a generator [13].

Proposition 1. *An itemset* X *is a generator iff* $supp(X) \neq min_{i \in X}(supp(X \setminus \{i\}))$ [14].

Each of the frequent and rare zones is delimited by two subsets, the maximal elements and the minimal ones, respectively. The above intuitive ideas are formalized in the notion of a border introduced by Mannila and Toivonen in [15]. According to their definition, the maximal frequent itemsets constitute the *positive border* of the frequent zone[1] whereas the minimal rare itemsets form the *negative border* of the same zone.

Definition 3. *An itemset is a* maximal frequent itemset *(MFI) if it is frequent but all its proper supersets are rare.*

Definition 4. *An itemset is a* minimal rare itemset *(mRI) if it is rare but all its proper subsets are frequent.*

The levelwise search yields as a by-product all mRIs [15]. Hence we prefer a different optimization strategy that still yields mRIs while traversing only a subset of the frequent zone of the Boolean lattice. It exploits the minimal generator status of the mRIs. By Property 1, frequent generators (FGs) can be traversed in a levelwise manner while yielding their negative border as a by-product. It is enough to observe that mRIs are in fact generators:

Proposition 2. *All minimal rare itemsets are generators* [9].

[1] The frequent zone contains the set of frequent itemsets.

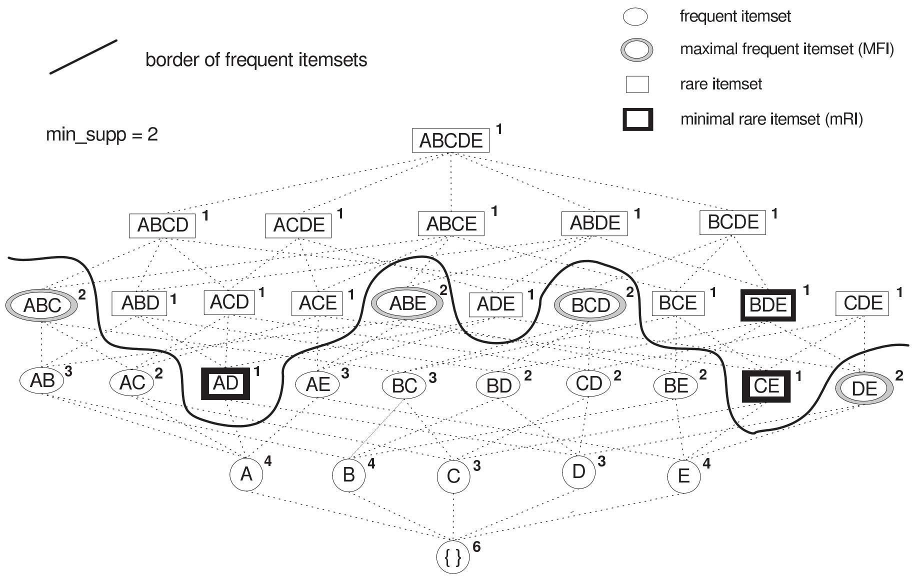


Fig. 1. The powerset lattice of dataset $\mathcal{D}$.

Finding Minimal Rare Itemsets in a Levelwise Manner

As pointed out by Mannila and Toivonen [15], the easiest way to reach the negative border of the frequent itemset zone, i.e., the mRIs, is to use a levelwise algorithm such as *Apriori*. Indeed, albeit a frequent itemset miner, *Apriori* yields the mRIs as a by-product.

Apriori-Rare [9] is a slightly modified version of *Apriori* that retains the mRIs. Thus, whenever an i-long candidate survives the frequent $i-1$ subset test, but proves to be rare, it is kept as an mRI.

MRG-Exp [9] produces the same output as *Apriori-Rare* but in a more efficient way. Following Proposition 2, *MRG-Exp* avoids exploring all frequent itemsets: instead, it looks after frequent generators only. In this case mRIs, which are rare generators as well, can be filtered among the negative border of the frequent generators. The output of *MRG-Exp* is identical to the output of *Apriori-Rare*, i.e. both algorithms find the set of mRIs.

3 Finding Minimal Rare Itemsets in a Depth-First Manner

Eclat [16] was the first FI-miner to combine the vertical encoding with a depth-first traversal of a tree structure, called IT-tree, whose nodes are $X \times t(X)$ pairs. *Eclat* traverses the IT-tree in a depth-first manner in a pre-order way, from left-to-right [16,17] (see Figure 2).

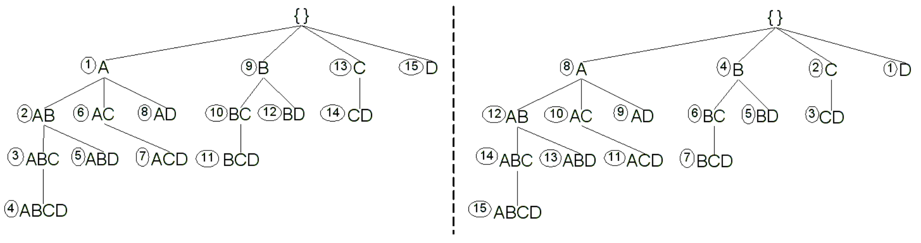


Fig. 2. Left: pre-order traversal with *Eclat*; **Right:** reverse pre-order traversal with *Eclat*. The direction of traversal is indicated in circles

3.1 Talky-G

Talky-G [18] is a vertical FG-miner following a depth-first traversal of the IT-tree and a right-to-left order on sibling nodes. *Talky-G* applies an inclusion-compatible traversal: it goes down the IT-tree while listing sibling nodes from right-to-left and not the other way round as in *Eclat*.

The authors of [19] explored that order for mining calling it *reverse pre-order*. They observed that for any itemset X its subsets appear in the IT-tree in nodes that lay either higher on the same branch as $(X, t(X))$ or on branches to the right of it. Hence, depth-first processing of the branches from right-to-left would perfectly match set inclusion, i.e., all subsets of X are met before X itself. While the algorithm in [19] extracts the so-called non-derivable itemsets, *Talky-G* uses this traversal to find the set of frequent generators. See Figure 2 for a comparison of *Eclat* and its “reversed” version.

3.2 Walky-G

In this subsection we present the algorithm *Walky-G*, which is the main contribution of this paper. Since *Walky-G* is an extension of *Talky-G*, we also present the latter algorithm at the same time. *Walky-G*, in addition to *Talky-G*, retains rare itemsets and checks them for minimality.

Hash structure. In *Walky-G* a hash structure is used for storing the already found frequent generators. This hash, called $fgMap$, is a simple dictionary with key/value pairs, where the key is an itemset (a frequent generator) and the value is the itemset's support.[2] The usefulness of this hash is twofold. First, it allows a quick look-up of the proper subsets of an itemset with the same support, thus the generator status of a frequent itemset can be tested easily (see Proposition 1). Second, this hash is also used to look-up the proper subsets of a minimal rare candidate. This way rare but non-minimal itemsets can be detected efficiently.

Pseudo code. Algorithm 1 provides the main block of *Walky-G*. First, the

[2] In our implementation we used the `java.util.HashMap` class for $fgMap$.

Algorithm 1 (main block of Walky-G):

```
 1) // for quick look-up of (1) proper subsets with the same support
 2) // and (2) one-size smaller subsets:
 3) $fgMap \leftarrow \emptyset$   // key: itemset (frequent generator); value: support
 4)
 5) root.itemset $\leftarrow \emptyset$   // root is an IT-node whose itemset is empty
 6) // the empty set is included in every transaction:
 7) root.tidset $\leftarrow$ {all transaction IDs}
 8) $fgMap$.put($\emptyset, |\mathcal{O}|$)   // the empty set is an FG with support 100%
 9) loop over the vertical representation of the dataset ($attr$) {
10)       if ($min\_supp \leq attr$.supp $< |\mathcal{O}|$) {
11)             // $|\mathcal{O}|$ is the total number of objects in the dataset
12)             root.addChild($attr$)   // attr is frequent and generator
13)       }
14)       if ($0 < attr$.supp $< min\_supp$) {
15)             saveMri($attr$)   // attr is a minimal rare itemset
16)       }
17) }
18) loop over the children of root from right-to-left ($child$) {
19)       saveFg($child$)   // the direct children of root are FGs
20)       extend($child$)   // discover the subtree below child
21) }
```

IT-tree is initialized, which involves the creation of the root node, representing the empty set (of 100% support, by construction). *Walky-G* then transforms the layout of the dataset in vertical format, and inserts below the root node all 1-long frequent itemsets. Such a set is an FG whenever its support is less than 100%. Rare attributes (whose support is less than min_supp) are minimal rare itemsets since all their subsets (in this case, the empty set) are frequent. Rare attributes with support 0 are not considered.

The `saveMri` procedure processes the given minimal rare itemset by storing it in a database, by printing it to the standard output, etc. At this point, the dataset is no more needed since larger itemsets can be obtained as unions of smaller ones while for the images intersection must be used.

The `addChild` procedure inserts an IT-node under a node. The `saveFg` procedure stores a given FG with its support value in the hash structure $fgMap$.

In the core processing, the `extend` procedure (see Algorithm 2) is called recursively for each child of the root in a right-to-left order. At the end, the IT-tree contains all FGs. Rare itemsets are verified during the construction of the IT-tree and minimal rare itemsets are retained. The `extend` procedure discovers all FGs in the subtree of a node. First, new FGs are tentatively generated from the right siblings of the current node. Then, certified FGs are added below the current node and later on extended recursively in a right-to-left order.

The `getNextGenerator` function (see Algorithm 3) takes two nodes and returns a new FG, or "null" if no FG can be produced from the input nodes. In addition, this method tests rare itemsets and retains the minimal ones. First, a candidate node is created by taking the union of both itemsets and the in-

Algorithm 2 ("extend" procedure):

```
Method: extend an IT-node recursively (discover FGs in its subtree)
Input:  an IT-node ($curr$)

1) loop over the right siblings of $curr$ from left-to-right ($other$) {
2)     $generator \leftarrow$ getNextGenerator($curr, other$)
3)     if ($generator \neq$ null) then $curr$.addChild($generator$)
4) }
5) loop over the children of $curr$ from right-to-left ($child$) {
6)     saveFg($child$)   // child is a frequent generator
7)     extend($child$)   // discover the subtree below child
8) }
```

tersection of their respective images. The input nodes are thus the candidate's *parents*. Then, the candidate undergoes a frequency test (test 1). If the test fails then the candidate is rare. In this case, the minimality of the rare itemset *cand* is tested. If all its one-size smaller subsets are present in $fgMap$ then *cand* is a minimal rare generator since all its subsets are FGs (see Property 1). From Proposition 2 it follows that an mRG is an mRI too, thus *cand* is processed by the `saveMri` procedure. If the frequency test was successful, the candidate is compared to its parents (test 2): if its tidset is equivalent to a parent tidset, then the candidate cannot be a generator. Even with both outcomes positive, an itemset may still not be a generator as a subsumed subset may lay elsewhere in the IT-tree. Due to the traversal strategy in *Walky-G*, all generator subsets of the current candidate are already detected and the algorithm has stored them in $fgMap$ (see the `saveFg` procedure). Thus, the ultimate test (test 3) checks whether the candidate has a proper subset with the same support in $fgMap$. A positive outcome disqualifies the candidate.

This last test (test 3) is done in Algorithm 4. First, one-size smaller subsets of *cand* are collected in a list. The two parents of *cand* can be excluded since *cand* was already compared to them in test 2 in Algorithm 3. If the support value of one of these subsets is equal to the support of *cand*, then *cand* cannot be a generator. Note that when the one-size smaller subsets are looked up in $fgMap$, it can be possible that a subset is missing from the hash. It means that the missing subset was tested before and turned out to subsume an FG, thus the subset was not added to $fgMap$. In this case *cand* has a non-FG subset, thus *cand* cannot be a generator either (by Property 1).

Candidates surviving the final test in Algorithm 3 are declared FG and added to the IT-tree. An unsuccessful candidate X is discarded which ultimately prevents any itemset Y having X as a prefix to be generated as candidate and hence substantially reduces the overall search space. When the algorithm stops, all frequent generators (and *only* frequent generators) are inserted in the IT-tree *and* in the $fgMap$ structure. Furthermore, upon the termination of the algorithm, all minimal rare itemsets have been found. For a running example, see Figure 3.

Algorithm 3 ("getNextGenerator" function):

Method: create a new frequent generator *and* filter minimal rare itemsets
Input: two IT-nodes (*curr* and *other*)
Output: a frequent generator or null

```
 1) $cand$.itemset $\leftarrow$ $curr$.itemset $\cup$ $other$.itemset
 2) $cand$.tidset $\leftarrow$ $curr$.tidset $\cap$ $other$.tidset
 3) if (cardinality($cand$.tidset) $<$ $min\_supp$)   // test 1: frequent?
 4) {    // now cand is an mRI candidate; let us test its minimality:
 5)       if (all one-size smaller subsets of $cand$ are in $fgMap$) {
 6)             saveMri($cand$)   // cand is an mRI, save it
 7)       }
 8)       return null   // not frequent
 9) }
10) // else, if it is frequent; test 2:
11) if (($cand$.tidset = $curr$.tidset) or ($cand$.tidset = $other$.tidset)) {
12)       return null   // not a generator
13) }
14) // else, if it is a potential frequent generator; test 3:
15) if (candSubsumesAnFg($cand$)) {
16)       return null   // not a generator
17) }
18) // if cand passed all the tests then cand is a frequent generator
19) return $cand$
```

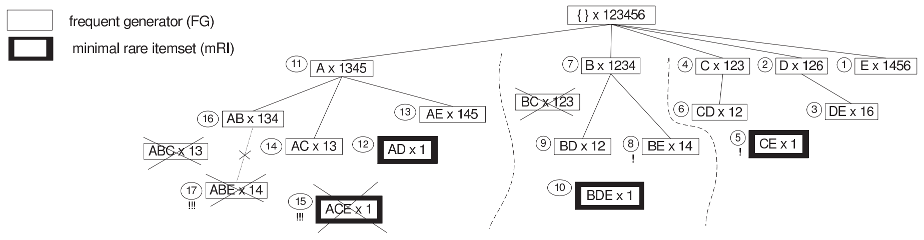


Fig. 3. The IT-tree built during the execution of *Walky-G* on dataset $\mathcal{D}$ with $min\_supp = 2$ (33%). Notice the two special cases: ACE is not an mRI because of CE; ABE is not an FG because of BE.

4 Experimental Results

In our experiments, we compared *Walky-G* against *Apriori-Rare* [9] and *MRG-Exp* [9]. The algorithms were implemented in Java in the Coron platform [20].[3] The experiments were carried out on a bi-processor Intel Quad Core Xeon 2.33 GHz machine running under Ubuntu GNU/Linux with 4 GB of RAM. All times reported are real, wall clock times as obtained from the Unix *time* command

[3] http://coron.loria.fr

Algorithm 4 ("candSubsumesAnFg" function):

Method: verify if *cand* subsumes an already found FG
Input: an IT-node (*cand*)

```
1) $subsets \leftarrow$ {one-size smaller subsets of $cand$ minus the two parents}
2) loop over the elements of $subsets$ ($ss$) {
3)     if ($ss$ is stored in $fgMap$) {
4)         $stored\_support \leftarrow fgMap$.get($ss$)   // get the support of ss
5)         if ($stored\_support = cand$.support) {
6)             return true   // case 1: cand subsumes an FG
7)         }
8)     }
9)     else   // if ss is not present in fgMap
10)    {   // case 2: cand has a non-FG subset ⇒ cand is not an FG either
11)        return true
12)    }
13) }
14) return false   // if we get here then cand is an FG
```

between input and output. For the experiments we have used the following datasets: T20I6D100K, C20D10K, C73D10K, and MUSHROOMS. The T20[4] is a sparse dataset, constructed according to the properties of market basket data that are typical weakly correlated data. The C20 and C73 are census datasets from the PUMS sample file, while the MUSHROOMS[5] describes mushrooms characteristics. The last three are highly correlated datasets.

The execution times of the three algorithms are illustrated in Table 1. The table also shows the number of frequent itemsets, the number of frequent generators, the proportion of the number of FGs to the number of FIs, and the number of minimal rare itemsets. The last column shows the number of mRIs whose support values exceed 0.

The T20 synthetic dataset mimics market basket data that are typical sparse, weakly correlated data. In this dataset, the number of FIs is small and nearly all FIs are generators. Thus, *MRG-Exp* works exactly like *Apriori-Rare*, i.e. it has to explore almost the same search space. Though *Walky-G* needs to explore a search space similar to *Apriori-Rare*'s, it can perform much better due to its depth-first traversal.

In datasets C20, C73, and MUSHROOMS, the number of FGs is much less than the total number of FIs. Hence, *MRG-Exp* and *Walky-G* can take advantage of exploring a much smaller search space than *Apriori-Rare*. Thus, *MRG-Exp* and *Walky-G* perform much better on dense, highly correlated data. For example, on MUSHROOMS at $min\_supp = 10\%$, *Apriori-Rare* needs to extract 600,817 FIs, while *MRG-Exp* and *Walky-G* extract 7,585 FGs only. This means that *MRG-Exp* and *Walky-G* reduce the search space of *Apriori-Rare* to 1.26%! The

[4] http://www.almaden.ibm.com/software/quest/Resources/
[5] http://kdd.ics.uci.edu/

Table 1. Response times of Apriori-Rare, MRG-Exp, and Walky-G.

min_supp	execution time (sec.)			# FIs	# FGs	$\frac{\#FGs}{\#FIs}$	# mRIs
	Apriori-Rare	MRG-Exp	**Walky-G**				(support > 0)
T20I6D100K							
10%	3.25	3.24	**1.61**	7	7	100.00%	907
0.75%	30.22	30.92	**11.80**	4,710	4,710	100.00%	211,561
0.5%	49.30	48.82	**15.89**	26,836	26,305	98.02%	268,589
0.25%	115.35	117.11	**33.47**	155,163	149,447	96.32%	534,088
C20D10K							
30%	21.92	5.49	**0.57**	5,319	967	18.18%	226
20%	56.43	9.70	**0.62**	20,239	2,671	13.20%	376
10%	157.09	18.27	**0.77**	89,883	9,331	10.38%	837
5%	366.34	28.35	**0.93**	352,611	23,051	6.54%	1,867
2%	878.93	40.77	**1.47**	1,741,883	57,659	3.31%	7,065
C73D10K							
95%	35.97	6.97	**0.84**	1,007	121	12.02%	1,622
90%	453.93	48.65	**0.90**	13,463	1,368	10.16%	1,701
85%	1,668.19	117.62	**0.95**	46,575	3,513	7.54%	1,652
Mushrooms							
40%	3.24	1.77	**0.50**	505	153	30.30%	251
30%	9.39	3.09	**0.51**	2,587	544	21.03%	402
15%	160.88	8.32	**0.66**	99,079	3,084	3.11%	1,741
10%	676.53	13.22	**0.76**	600,817	7,585	1.26%	2,916

advantages of the depth-first approach of *Walky-G* is more spectacular on dense datasets: the execution times, with the exception of one case in Table 1, are always below 1 second.

5 Conclusion and Future Work

We presented an approach for rare itemset mining from a dataset that splits the problem into two tasks. Our new algorithm, *Walky-G*, limits the traversal of the frequent zone to frequent generators *only*. This traversal is achieved through a depth-first strategy. Experimental results prove the interest of our method not only on dense, highly correlated datasets, but on sparse ones too. Our approach breaks with the dominant levelwise algorithmic schema and shows that it outperforms its current levelwise competitors.

References

1. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast discovery of association rules. In: Advances in knowledge discovery and data mining. American Association for Artificial Intelligence (1996) 307–328
2. Hand, D., Mannila, H., Smyth, P.: Principles of Data Mining. The MIT Press, Cambridge (MA) (2001)
3. Weiss, G.: Mining with rarity: a unifying framework. SIGKDD Explor. Newsl. **6**(1) (2004) 7–19

4. Hacene, M.R., Toussaint, Y., Valtchev, P.: Mining safety signals in spontaneous reports database using concept analysis. In: Proc. 12th Conf. on AI in Medicine, AIME 2009. Volume 5651 of Lecture Notes in Computer Science. (2009) 285–294
5. Liu, B., Hsu, W., Ma, Y.: Mining Association Rules with Multiple Minimum Supports. In: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '99), New York, NY, USA, ACM Press (1999) 337–341
6. Yun, H., Ha, D., Hwang, B., Ryu, K.: Mining association rules on significant rare data using relative support. Journal of Systems and Software **67**(3) (2003) 181–191
7. Koh, Y., Rountree, N.: Finding Sporadic Rules Using Apriori-Inverse. In: Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD '05), Hanoi, Vietnam. Volume 3518 of Lecture Notes in Computer Science., Springer (May 2005) 97–106
8. Koh, Y.S., Rountree, N.: Rare Association Rule Mining and Knowledge Discovery: Technologies for Infrequent and Critical Event Detection. Information Science Reference - Imprint of: IGI Publishing, Hershey, PA (2009)
9. Szathmary, L., Napoli, A., Valtchev, P.: Towards Rare Itemset Mining. In: Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI '07). Volume 1., Patras, Greece (Oct 2007) 305–312
10. Zaki, M.J., Hsiao, C.J.: CHARM: An Efficient Algorithm for Closed Itemset Mining. In: SIAM International Conference on Data Mining (SDM' 02). (Apr 2002) 33–43
11. Calders, T., Rigotti, C., Boulicaut, J.F.: A Survey on Condensed Representations for Frequent Sets. In Boulicaut, J.F., Raedt, L.D., Mannila, H., eds.: Constraint-Based Mining and Inductive Databases. Volume 3848 of Lecture Notes in Computer Science., Springer (2004) 64–80
12. Li, J., Li, H., Wong, L., Pei, J., Dong, G.: Minimum Description Length Principle: Generators Are Preferable to Closed Patterns. In: AAAI, AAAI Press (2006) 409–414
13. Kryszkiewicz, M.: Concise Representations of Association Rules. In: Proceedings of the ESF Exploratory Workshop on Pattern Detection and Discovery. (2002) 92–109
14. Bastide, Y., Taouil, R., Pasquier, N., Stumme, G., Lakhal, L.: Mining Frequent Patterns with Counting Inference. SIGKDD Explor. Newsl. **2**(2) (2000) 66–75
15. Mannila, H., Toivonen, H.: Levelwise Search and Borders of Theories in Knowledge Discovery. Data Mining and Knowledge Discovery **1**(3) (1997) 241–258
16. Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: New Algorithms for Fast Discovery of Association Rules. In: Proceedings of the 3rd International Conference on Knowledge Discovery in Databases. (August 1997) 283–286
17. Zaki, M.J.: Scalable Algorithms for Association Mining. IEEE Transactions on Knowledge and Data Engineering **12**(3) (2000) 372–390
18. Szathmary, L., Valtchev, P., Napoli, A., Godin, R.: Efficient Vertical Mining of Frequent Closures and Generators. In: Proc. of the 8th Intl. Symposium on Intelligent Data Analysis (IDA '09). Volume 5772 of LNCS., Lyon, France, Springer (2009) 393–404
19. Calders, T., Goethals, B.: Depth-first non-derivable itemset mining. In: Proceedings of the SIAM International Conference on Data Mining (SDM '05), Newport Beach, USA. (Apr 2005)
20. Szathmary, L.: Symbolic Data Mining Methods with the Coron Platform. PhD Thesis in Computer Science, Univ. Henri Poincaré – Nancy 1, France (Nov 2006)

Multiple Keyword Pattern Matching using Position Encoded Pattern Lattices

Fritz J. Venter[1], Bruce W. Watson[2], and Derrick G. Kourie[1]

[1] University of Pretoria, {fritz, dkourie}@fastar.org
[2] Stellenbosch University, bruce@fastar.org

Abstract. Formal concept analysis is used as the basis for two new multiple keyword string pattern matching algorithms. The algorithms addressed are built upon a so-called *position encoded pattern lattice* (PEPL). The algorithms presented are in conceptual form only; no experimental results are given. The first algorithm to be presented is easily understood and relies directly on the PEPL for matching. Its worst case complexity depends on both the length of the longest keyword, and the length of the search text. Subsequently a finite-automaton-like structure, called a *PEPL automaton*, is defined which is derived from the PEPL, and which forms the basis for a second more efficient algorithm. In this case, worst case behaviour depends only on the length of the input stream. The second algorithm's worst case performance is the same as the *matching phase* of the well-known (advanced) Aho-Corasick multiple-keyword pattern matching algorithm—widely regarded as the multiple keyword pattern matching algorithm of choice in contexts such as network intrusion detection. The first algorithm's performance is comparable to that of the *matching phase* of the lesser-known *failure-function* version of Aho-Corasick.

1 Overview

The *(multiple) keyword pattern matching problem* (in which the patterns are finite strings, or 'keywords') consists of finding *all occurrences* (including overlapping ones) of the keywords within an *input* string. Typically, the input string is much larger than the set of keywords and the set of keywords are fixed, meaning they can be preprocessed to produce data-structures for later use while processing the input string. We also make these assumptions in this paper, as this problem variant corresponds to many real-life applications in security, computational biology, etc [8]. Several decades of keyword pattern matching research have yielded many well-known algorithms, such as Knuth-Morris-Pratt, Boyer-Moore, Aho-Corasick, and Commentz-Walter. Overview articles are typically more accessible than the original literature—see [3, 4, 7] for comprehensive overviews and [10, 2] for taxonomies and correctness proofs of such algorithms.

In this text, the idea of "position encoding" of a set of patterns is introduced. This strategy serves as an alternative to traditional algorithms used to match a *set* of patterns. These algorithms typically rely on common pattern prefixes, suffixes and/or factors in general. The Aho-Corasick [1] algorithm (AC) is probably the best known and most widely used of these algorithms. Its best- and worst-case performance are both being linear in the size of the input stream and independent of the number of patterns to be matched.

Formal concept analysis (FCA) is used to leverage the potential benefits of position encoding. A formal context in which O is the set of *objects*, A is the set of *attributes*, and I is an *incidence relation* between the objects and the attributes will be denoted by $\mathbb{K} = \langle O, A, I\rangle$. The concept lattice, $\mathfrak{B}$, derived from the context $\langle O, A, I\rangle$ will be denoted by $\mathfrak{B}(\langle O, A, I\rangle)$. The extent, intent and set of own objects of a concept c in a concept lattice will be denoted by $\mathbf{extent}(c), \mathbf{intent}(c)$, and $\mathbf{ownobj}(c)$ respectively. The infimum of a set of concepts C will be denoted by $\mathbf{inf}(C)$. Finally, $\top(b)$ denotes the *attribute top* of attribute b—i.e. $\top(b)$ is the largest concept whose intent contains b.

In Section 2, it is shown how FCA can be used to construct a concept lattice from a position encoded set of patterns. Such a lattice is called a position encoded pattern lattice (PEPL). A first algorithm, called *PMatch*, is developed in Section 3, which takes such a PEPL together with text stream to be searched as input and produces the desired match occurrences as output, albeit in a rather inefficient way. As an alternative, a so-called *PEPL automaton* is defined in Section 4, based on the information in a PEPL. A second algorithm given in Section 5 uses this automaton and the text stream to be searched as input and also produces the desired match occurrences as output. However, in this instance the theoretical performance of the algorithm corresponds to that of Aho-Corasick. In a final section, we reflect on the implications of these results.

2 Position Encoded Pattern Lattices (PEPLs)

The length of string p will be denoted by $|p|$ and its $(i+1)^{st}$ element by p_i for $i \in [0, |p|)$. A *match occurrence* of a single pattern p in target s is a pair $\langle p, t\rangle$, such that $\forall\ k \in [0, |p|)$, $p_k = s_{t+k}$. The problem of matching a *set* of patterns P on target s can be defined as the requirement to construct the set of *all* match occurrences, denoted by MO in our algorithms.

Definition 1 (Position encoding of a set of patterns). *The* position encoding *of string w is the set of position-symbol pairs denoted by $\overset{\rightarrow\bullet}{w}$ and is given by $\overset{\rightarrow\bullet}{w} = (\bigcup\ k : k \in [1, |w|] : \{\langle k, w_{k-1}\rangle\})$.*

The position encoding *of a set of strings P is denoted $\overset{\rightarrow\bullet}{P}$ and is given by $\overset{\rightarrow\bullet}{P} = (\bigcup\ w : w \in P : \overset{\rightarrow\bullet}{w})$*

For example, the position encoding of “pack” is $\overset{\rightarrow\bullet}{pack} = \{\langle 1, p\rangle, \langle 2, a\rangle, \langle 3, c\rangle, \langle 4, k\rangle\}$, and of “packet” it is $\overset{\rightarrow\bullet}{packet} = \{\langle 1, p\rangle, \langle 2, a\rangle, \langle 3, c\rangle, \langle 4, k\rangle, \langle 5, e\rangle, \langle 6, t\rangle\}$. In this case, the position encoding of the set of patterns $P = \{\text{pack}, \text{packet}\}$ and of "packet" happens to be the same, i.e. $\overset{\to\bullet}{P} = \overset{\to\bullet}{packet}$.

Given any set of patterns, we can now constitute a formal context $K^{\#}$ along the following lines. Regard the words in the set of patterns as a set of objects. Let the position-symbol pairs of the position encoding of the set of patterns serve as attributes of these objects: a given word has as its attributes all the position-symbol pairs that make up its position-encoding.

As an example, consider the set of patterns $P = \{abc, aabc, abcc\}$. Table 1 shows the cross table that represents the position encoded formal context derived from P. This context can be denoted by $\langle P, \overset{\to\bullet}{P}, I^{\overset{\to\bullet}{P}} \rangle$, where $I^{\overset{\to\bullet}{P}}$ is the incidence relation between objects and attributes depicted in the cross table. The formal concept lattice to be derived from such a context will be called a *Position Encoded Pattern Lattice* (PEPL), denoted by $\overset{\to\bullet}{\mathfrak{P}}(\langle P, \overset{\to\bullet}{P}, I^{\overset{\to\bullet}{P}} \rangle)$ or, more concisely, by $\overset{\to\bullet}{\mathfrak{P}}$. The cover graph of the underlying PEPL is shown in Figure 2.

$\langle P, \overset{\to\bullet}{P}, I^{\overset{\to\bullet}{P}} \rangle$	$\langle 1, a\rangle$	$\langle 2, a\rangle$	$\langle 2, b\rangle$	$\langle 3, b\rangle$	$\langle 3, c\rangle$	$\langle 4, c\rangle$
abc	×		×		×	
aabc	×	×		×		
abcc	×		×		×	×

Fig. 1: Position encoded context for $P = \{abc, aabc, abcc\}$.

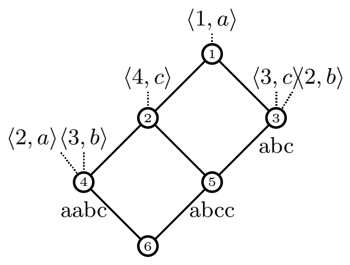


Fig. 2: Cover graph of PEPL for $P = \{abc, aabc, abcc\}$.

If a search of text s is currently at position $s[t]$, and it is found that $s[t+n] = a$, then attribute $\langle n, a\rangle$ is said to *positively check against* s at t.

It is evident that the intent of a PEPL concept has the following property: If all the attributes in the intent have been positively checked against a search text s at position t, then the (one or more) words that are own objects of the concept match the text, starting at position t. This is clearly the case for concepts 3, 4 and 5 with own objects "abc", "aabc" and "abcc" respectively.

3 PEPL-based Matching Using *PMatch*

Algorithm 1 described below is based on the insights of the previous section. Its top level procedure is called *PMatch*, which takes as input a PEPL $\overset{\to\bullet}{\mathfrak{P}}(\langle P, \overset{\to\bullet}{P}, I^{\overset{\to\bullet}{P}} \rangle)$ (or simply $\overset{\to\bullet}{\mathfrak{P}}$) and a text, s. It then finds in s all match occurrences of words in P, recording them in *MO*. The algorithm is articulated in Dijkstra's guarded command language (GCL), widely used for its conciseness and precision [5]. The definition of *PMatch* assumes constant *minlength(P)* as the length of the shortest keyword in P. To avoid notational clutter, $\overset{\rightarrow\bullet}{\mathfrak{P}}$, s and *MO* are assumed to be globally accessible to all procedures. A special symbol, *nil*, is used to designate a non-existent concept, specifically the parent of the top concept.

PMatch calls *matchIntent* for each character in s where a match could possibly start (i.e. the tail is ignored). The condition in the associated for-loop is intended to signify that these probes are from left to right. In each call the intent of the top of the lattice and its non-existent parent, *nil*, are used as parameters.

matchIntent takes a string position t, and two concepts, c and p, as parameters. It is assumed that c is a child of p and the set difference, Δ, between their intents is computed. The special case of $\top$, which has no parent, is catered for. A loop checks whether all the attributes in Δ indicate positional matches in the text s as offset by the current search position, t—i.e. the loop removes from Δ all attributes of the form $\langle i, \alpha\rangle$ such that $s[t+i] = \alpha$. If this reduces Δ to the empty set, then a match occurrence is considered to have been found for each own object at c. Moreover, $match(c, t)$ can be called to investigate whether further match occurrences at t can be inferred by considering c's children. $match(p, t)$, in turn, simply sweeps through the children of p, recursively invoking *matchIntent* in each case.

Algorithm 1 *PEPL Based Matching*

```
proc $PMatch(\overset{\rightarrow\bullet}{\mathfrak{P}}, s)$
    $MO, j := \varnothing, minlength(P)$;
    { Traverse target string s from left to right }
    for $(t \in [0, |s| - j + 1)) \rightarrow$
        $matchIntent(t, \top, nil)$
    rof
corp{ post : MO is the set of match occurrences of P in s }
proc $matchIntent(t, c, p)$
    if $(p = nil) \rightarrow \Delta := \mathbf{intent}(c)$
    [] $(p \neq nil) \rightarrow \Delta := \mathbf{intent}(c) \setminus \mathbf{intent}(p)$
    fi;
    do $(\exists\langle i, \alpha\rangle : \langle i, \alpha\rangle \in \Delta : (s[t + i - 1] = \alpha)) \rightarrow$
        $\Delta := \Delta \setminus \{\langle i, \alpha\rangle\}$
    od;
    if $(\Delta = \varnothing) \rightarrow MO := MO \cup \mathbf{ownobj}(c) \times \{t\}$;
                    for all $c' \in children(c) \rightarrow matchIntent(t, c', c)$ rof
    [] $(\Delta \neq \varnothing) \rightarrow$ skip
    fi
corp
```

To illustrate how Algorithm 1 works, consider the keywords to match $P = \{abc, aabc, abcc\}$ and the target $s = aaabcdabccd$. The formal context $\langle P, \overset{\rightarrow\bullet}{P}, I^{\overset{\rightarrow\bullet}{P}} \rangle$ is given in Fig. 1 and the cover graph for the corresponding PEPL, $\overset{\rightarrow\bullet}{\mathfrak{P}}$, is in Fig. 2. For convenience, the intents and own object sets of each concept are made explicit in Table 1. Table 2 provides a trace summary of calls to *matchIntent*. The first

Id of c	**intent**(c)	**ownobj**(c)
$1 = \top$	$\{\langle 1, a\rangle\}$	
2	$\{\langle 1, a\rangle, \langle 4, c\rangle\}$	
3	$\{\langle 1, a\rangle, \langle 3, c\rangle, \langle 2, b\rangle\}$	abc
4	$\{\langle 1, a\rangle, \langle 4, c\rangle, \langle 3, b\rangle, \langle 2, a\rangle\}$	aabc
5	$\{\langle 1, a\rangle, \langle 4, c\rangle, \langle 3, c\rangle, \langle 2, b\rangle\}$	abcc
$6 = \bot$	$\{\langle 1, a\rangle, \langle 4, c\rangle, \langle 3, b\rangle, \langle 3, c\rangle, \langle 2, b\rangle, \langle 2, a\rangle\}$	

Table 1: Details of concepts of the PEPL in Fig. 2

column shows t, the offset into s from which matching positions are calculated. The second and third columns show the lattice concept visited and its concept participating in the call to *matchIntent*. The fourth column marked Δ gives the set difference between the intent of the child and parent concept. A column per symbol in the string *aaabcdabccd* then follows. The last column gives the own object set to be to be used to update *MO* when a match has been found. Note that since $minlength(P) = 3$ and $|s| = 11$, the trace ranges over $t \in [0, 9)$. Each

t	c	p	Δ	a	a	a	b	c	d	a	b	c	c	d	**ownobj**(c)
0	1	nil	$\{\langle 1,a\rangle\}$	T											$\varnothing$
0	2	1	$\{\langle 4,c\rangle\}$				F								
0	3	1	$\{\langle 2,b\rangle, \langle 3,c\rangle\}$		F										
1	1	nil	$\{\langle 1,a\rangle\}$		T										$\varnothing$
1	2	1	$\{\langle 4,c\rangle\}$					T							$\varnothing$
1	4	2	$\{\langle 2,a\rangle, \langle 3,b\rangle\}$			T	T								$\{aabc\}$
1	3	1	$\{\langle 2,b\rangle, \langle 3,c\rangle\}$			F									
2	1	nil	$\{\langle 1,a\rangle\}$			T									
2	2	1	$\{\langle 4,c\rangle\}$						F						
2	3	1	$\{\langle 2,b\rangle, \langle 3,c\rangle\}$				T	T							$\{abc\}$
2	5	3	$\{\langle 4,c\rangle\}$						F						
3	1	nil	$\{\langle 1,a\rangle\}$				F								
4	1	nil	$\{\langle 1,a\rangle\}$					F							
5	1	nil	$\{\langle 1,a\rangle\}$						F						
6	1	nil	$\{\langle 1,a\rangle\}$							T					$\varnothing$
6	2	1	$\{\langle 4,c\rangle\}$										T		$\varnothing$
6	4	2	$\{\langle 2,a\rangle, \langle 3,b\rangle\}$								F				
6	5	2	$\{\langle 2,b\rangle, \langle 3,c\rangle\}$								T	T			$\{abcc\}$
6	3	1	$\{\langle 2,b\rangle, \langle 3,c\rangle\}$								T	T			$\{abc\}$
7	1	nil	$\{\langle 1,a\rangle\}$								F				
8	1	nil	$\{\langle 1,a\rangle\}$									F			

Table 2: Algorithm 1 trace: matching $\{abc, aabc, abcc\}$ in *aaabcdabccd*

row is a matching step of the algorithm—i.e. every row represents a call of the function *matchIntent* . As an example, the first row indicates that the matching position $t = 0$ and the attribute set to match is $\Delta = \{\langle 1,a\rangle\}$. The first (and only) element of the set is $\langle i,\alpha\rangle = \langle 1,a\rangle$. This means that position $t+i=1$ is checked for the symbol $\alpha = a$, which is indeed the case as indicated by the "T" (for the boolean value *true*) shown in the first column for the target string. All "T" entries in the table indicate that attributes in Δ have been successfully matched in the do-loop of *matchIntent*. Once Δ has been reduced to $\varnothing$, *MO* has to be updated. Of course, if the concept has no own object—as is the case for the top concept marked 1—then nothing is added to *MO* (i.e. $\mathbf{ownobj}(c) \times \{t\} = \varnothing$). Subsequent calls to *match* without updating t, recursively deal with children concepts of the one currently under test. The second row of the table therefore logs the results the call to *matchIntent* made via *match* in respect of concept 2, the leftmost child of concept 1. In this case, the intent difference set is $\Delta = \{\langle 4,c\rangle\}$, and since $\nexists\langle i,\alpha\rangle : \{\langle 4,c\rangle\} : (s[t+i-1] = \alpha)$, (or, more explicitly, $s[0+4-1] = b$ and not c) *matchIntent* cannot reduce Δ to $\varnothing$. This is indicated by "F" (for false) as an entry in the relevant column of the table. Control now returns to *match*, where the next child of concept 1, namely concept 3, is considered. Further rows of the table illustrate the execution steps of Algorithm 1 for the rest of the target string.

PMatch eliminates sets of words from P that do not match in s without ever backing up in s, i.e. t is monotonically increasing. In this sense *PMatch* is an *online* algorithm, similar to the AC algorithm. However, *PMatch* sometimes *revisits* symbols in s. Such revisits are reflected by the multiple entries in various columns representing symbols in *aaabcdabccd* in Table 2.

The execution complexity of the matching process *per position checked in s* is bounded by the size of the PEPL. Table 2 shows how all concepts are visited when $t = 6$. An (rather conservative) upper bound of the complexity of Algorithm 1 is therefore $(|\overset{\rightarrow\bullet}{\mathfrak{P}}| \times |s|)$. The advanced AC algorithm is of course more efficient than this. Not only does it check every symbol in s exactly once; it also avoids the application of the expensive set difference operator that is applied in *matchIntent* of Algorithm 1. Instead, the advanced AC simply makes an automaton transition and considers whether an accepting state has been entered. In the upcoming sections, we refine our algorithm to arrive at a PEPL-based algorithm with similar performance characteristics to advanced AC.

4 PEPL Automata

For PEPL based matching to achieve the same order-of-magnitude performance as the advanced AC algorithm, this section defines a structure called a PEPL *Automaton*.

Firstly, the position encoded formal context for the set of keywords P is augmented. This augmented context has additional entries to reflect information about each keyword $p \in P$ whose first symbol matches the symbol at the m^{th} index of some other keyword, where $m > 0$.

Definition 2 (Augmentation operator). *For two strings p and y we define the operator denoted $\#$ as*

$$p\#y = \begin{cases} \{(p)y\} & \textit{if } y \neq \varepsilon \wedge p \neq \varepsilon \\ \varnothing & \textit{otherwise} \end{cases}$$

Definition 3 (Augmentation of a string). *We define the augmentation of string y with respect to string x as*

$$\langle x, y\rangle^{\#} = (\bigcup\ p, s, r, t : ((x = p \cdot s \wedge y = s \cdot r \wedge s \neq \varepsilon) \vee (x = p \cdot y \cdot t)) : p\#y\ )$$

Thus, for each proper suffix[3] s of x that is also prefix of y, we compute the singleton set $p\#y$ (but possibly the empty set) and add all such singleton sets into one big set. Note that there may be several such sets. For example, if $x = aa$ and $y = aaaa$ then the following decompositions of x are relevant :
$x = \varepsilon \cdot aa$; $x = a \cdot a$; so that

$$\begin{aligned} \langle x, y\rangle^{\#} &= \varepsilon\#aaaa \cup a\#aaaa \\ &= \varnothing \cup \{(a)aaaa\}\} \\ &= \{(a)aaaa\} \end{aligned}$$

Definition 4 (String-augmentation of a language). *We define the string-augmentation of language V with respect to string w as follows.*

$$\langle V, w\rangle^{\#} = (\bigcup\ v : v \in V : \langle v, w\rangle^{\#})$$

Then

$$\begin{aligned} \langle\{x, y\}, y\rangle^{\#} &= \{(a)aaaa, (aa)aaaa, (aaa)aaaa\} \\ \langle\{x, y\}, x\rangle^{\#} &= \{(a)aa, (aa)aa, (aaa)aa\} \\ \langle\{x\}, y\rangle^{\#} &= \{(a)aaaa\} \\ \langle\{y\}, x\rangle^{\#} &= \{(a)aa, (aa)aa, (aaa)aa\} \end{aligned}$$

Definition 5. *For a set of patterns P we define the augmented patterns as*

$$P^{\#} = P\ \cup\ (\bigcup\ p : p \in P : \langle P \setminus \{p\}, p\rangle^{\#})$$

Thus,

$$\begin{aligned} \{x, y\}^{\#} &= \{x, y\} \cup \langle\{x\}, y\rangle^{\#} \cup \langle\{y\}, x\rangle^{\#} \\ &= \{aa, aaaa\} \cup \{(a)aaaa\}\{(a)aa, (aa)aa, (aaa)aa\} \\ &= \{aa, aaaa, (a)aaaa, (a)aa, (aa)aa, (aaa)aa\} \end{aligned}$$

[3] By proper suffix, we mean that the empty string is not taken as a suffix.

$K^{\#}$	$\langle 1, a\rangle$	$\langle 2, a\rangle$	$\langle 2, b\rangle$	$\langle 3, c\rangle$	$\langle 4, c\rangle$	$\langle 3, b\rangle$	$\langle 5, c\rangle$
abc	×		×	×			
aabc	×	×			×	×	
abcc	×		×	×	×		
(a)abc	×	×			×	×	
(a)abcc	×	×			×	×	×

Table 3: Context derived by augmenting $P = \{abc, aabc, abcc\}$

Given set of patterns P, we can constitute a formal context denoted $K^{\#}$ for a PEPL using objects from the augmented set of patterns $P^{\#}$ and attributes from $\overset{\rightarrow\bullet}{P}{}^{\#}$.

Example 1. As an example, consider again the set of patterns $P = \{abc, aabc, abcc\}$ augmented with the set of patterns $P^{\#}$ derived as follows:

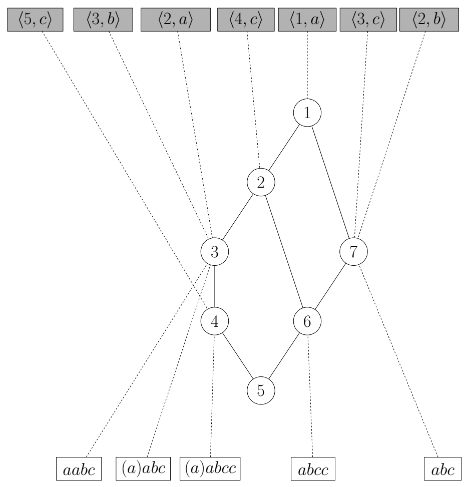
$$\begin{aligned} P^{\#} &= P \ \cup \ \langle\{aabc, abcc\}, abc\rangle^{\#} \ \cup \ \langle\{abc, abcc\}, aabc\rangle^{\#} \ \cup \ \langle\{abc, aabc\}, abcc\rangle^{\#} \\ &= P \ \cup \ \{(a)abc\} \ \cup \ \varnothing \ \cup \ \{(a)abcc\} \\ &= P \ \cup \ \{(a)abc, (a)abcc\} \\ &= \{abc, aabc, abcc, (a)abc, (a)abcc\} \end{aligned}$$

Table 3 depicts the context derived from the set of objects $P^{\#}$ and their corresponding position encoding as attributes.

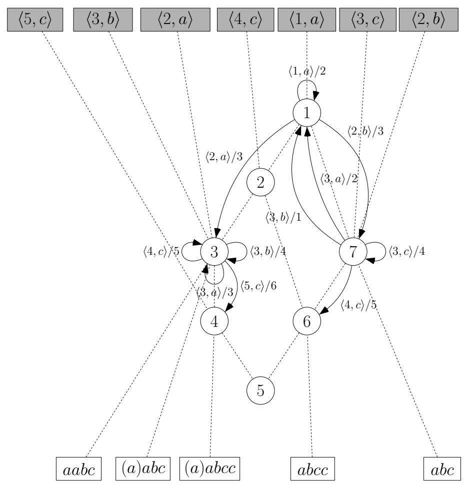
Algorithm 2 discussed in Section 5 uses $\overset{\bullet\rightarrow}{\mathfrak{P}}$ to match a set of keywords P against a target string. The algorithm relies on a second pre-processing step after $\overset{\bullet\rightarrow}{\mathfrak{P}}$ has been generated from its context. This step traverses $\overset{\bullet\rightarrow}{\mathfrak{P}}$ to create a special automaton, called a "Position Encoded Pattern Lattice Automaton" or simply a PEPL Automaton. It is defined as follows.

Definition 6 (PEPL Automaton). *Given $\overset{\bullet\rightarrow}{\mathfrak{P}}$, the PEPL that has been derived from set of patterns $P^{\#}$, the associated* PEPL automaton *is a five-tuple $\langle Q^{\overset{\bullet\rightarrow}{P}}, V^{\overset{\bullet\rightarrow}{P}}, \delta^{\overset{\bullet\rightarrow}{P}}, {q_0}^{\overset{\bullet\rightarrow}{P}}, F^{\overset{\bullet\rightarrow}{P}}\rangle$ such that:*

- $Q^{\overset{\bullet\rightarrow}{P}} \subseteq \{0\} \cup (\bigcup\ q, c : q = id(c) \wedge c \in \overset{\bullet\rightarrow}{\mathfrak{P}} : \{q\})$ *is regarded as the automaton's set of states, where $id(c)$ simply gives a unique numerical identifier for concept c.*
- $V^{\overset{\bullet\rightarrow}{P}} = \overset{\bullet\rightarrow}{P}$ *is regarded as the automaton's alphabet, indicating position-symbol pairs.*
- $\delta^{\overset{\bullet\rightarrow}{P}} : Q^{\overset{\bullet\rightarrow}{P}} \times V^{\overset{\bullet\rightarrow}{P}} \nrightarrow Q^{\overset{\bullet\rightarrow}{P}} \times |P^{\#}{}_{Max}|$ *is the automaton's transition function. The mapping is generally determined by the recursive relationship:*



(a) Cover Graph for the PEPL for the context of $P^{\#}$



(b) PEPL Automaton superimposed on the Cover Graph

Fig. 3: PEPL derived from $P^{\#}$

$\delta^{\overset{\bullet}{\overrightarrow{P}}}(id(c), \langle i, \alpha\rangle) = id(c')/i'$ *where variables* c' *and* i' *are defined as follows:*

$$\langle c', i'\rangle = \begin{cases} \langle c, i\rangle & \textit{if } A(c,i,\alpha) \ \wedge B(c,i,\alpha) \\ \langle \top(\langle 1, \alpha\rangle), 2\rangle & \textit{if } A(c,i,\alpha) \ \wedge \neg B(c,i,\alpha) \wedge \top(\langle 1,\alpha\rangle) \neq \bot \\ \langle \top, 1\rangle & \textit{if } A(c,i,\alpha) \ \wedge \neg B(c,i,\alpha) \wedge \top(\langle 1,\alpha\rangle) = \bot \\ \langle \mathbf{inf}(\{c, \top(\langle i,\alpha\rangle)\}), i+1\rangle & \textit{otherwise} \end{cases}$$

where $A(c,i,\alpha) \equiv \mathbf{inf}(\{c, \top(\langle i,\alpha\rangle)\}) = \bot$ *and* $B(c,i,\alpha) \equiv \langle i-1, \alpha\rangle \in$ $\mathbf{intent}(c)$

The the first level recursion starts off with $c = \top$*. The notation* q/x *means that when a transition to state* q *is made, the automaton produces the additional value* x*.*

- $q_0^{\overset{\bullet}{\overrightarrow{P}}} = \top$ *is the automaton's start state, which is also the top concept of the PEPL.*
- $F^{\overset{\bullet}{\overrightarrow{P}}} = (\bigcup\ q, c\ :\ q = id(c) \wedge c \in Q^{\overset{\bullet}{\overrightarrow{P}}} \wedge |\mathbf{ownobj}(c)| > 0\ :\ \{q\})$ *is the automaton's set of final states.*

The above definition embeds sufficient information to derive algorithmically a DFA whose transition diagram can be superimposed on the cover graph of the PEPL.

It is assumed below that an function, *getFA*, is available which delivers a PEPL automaton $\mathcal{M}^{\overset{\bullet\leftrightarrow}{P}}$ when provided with a PEPL. As an example, *getFA*($\overset{\bullet\leftrightarrow}{\mathfrak{P}}$) will return $\mathcal{M}^{\overset{\bullet\leftrightarrow}{P}}$, the *PEPL-Automaton* (partially) shown in Fig. 3b, superimposed over the cover graph for $\overset{\bullet\leftrightarrow}{\mathfrak{P}}$. Note that in order to avoid clutter, a number of arcs have not been shown in Fig. 3b. For example, the many of transitions to $\top$ have been left out.

5 Matching Using a *PEPL-Automaton*

Viewed as a DFA, a PEPL automaton could be used to test whether a given sequence of its alphabet are in the regular set of patterns that it describes. For example, it can easily be seen in Figure 3 that, starting from $\top$, successive transitions on elements of the string $\langle\langle 1, a\rangle, \langle 2, b\rangle, \langle 3, c\rangle\rangle$ lead to the final state 6, affirming that this sequence is indeed part of the set of patterns described by the automaton, and and since *abc* is an own object of 6, affirming that *abc* is in the original set of patterns, P.

However, the PEPL is not primarily intended to be used in this way. Instead, the PEPL is used in Algorithm 2 to find all match opportunities in P in a text s. The algorithm's **do** processes symbols of s, updating variables c and i to keep track of partial matches in that part of s already processed. This is expressed as loop *invariant* $Inv(c, i, t) \equiv$

- MO contains all matches in $s_{[0,t-i+1)}$. (These are the matches *already processed.*)
- And $s_{[t-i+1,t)}$ matches the first $i-1$ characters of all patterns in **extent**(c). (These are the partial matches *in progress.*)

To illustrate matching as executed by Algorithm 2, consider the steps logged in Table 4 when matching the set of patterns *abc,aabc,abcc* against the target string *aaabcdabccd*. The first five entries of each row in this table shows the values of variables $t, s[t], c, i$ as they have been updated as a result of the statements in the body of the main loop in Algorithm 2. The next set of entries in the respective row are all empty except for the position where $s[t]$ has been matched. At such a position a "T" or "F" is shown depending on the result of the match. The next entry in the row shows the size of the intent of c. The last entry in this row gives the patterns matched at the step represented by the row.

Algorithm 2 *PEPL Automaton Based Matching*

```
proc PAutMatch($\overset{\bullet\leftrightarrow}{\mathfrak{P}}$, $s$)
    $MO := \varnothing$;
    $\mathcal{M}^{\overset{\bullet\leftrightarrow}{P}} := getFA(\overset{\bullet\leftrightarrow}{\mathfrak{P}})$;
    $\langle c, i\rangle, t := \langle q_0^{\overset{\bullet\leftrightarrow}{P}}, 1\rangle, 0$;   { Recall that $q_0^{\overset{\bullet\leftrightarrow}{P}} = \top$ }
```

```
        { invariant: $Inv(c, i, t)$ }
        do  $(t < |s|) \rightarrow \langle c, i\rangle, t := \delta^{\overset{\bullet\leftrightarrow}{P}}(c, \langle i, s[t]\rangle), t + 1;$
                if  $(i = |\mathbf{intent}(c)|) \rightarrow MO := MO \cup (\{t\} \times \mathbf{ownobj}(c))$
                []  $(i \neq |\mathbf{intent}(c)|) \rightarrow$ skip
                fi
        od
corp { post : MO is the set of match occurrences of $P$ in $s$ }
```

As an example we present an explanation of the steps up to the first matched patterns being recorded. Consider the first row of Table 4. This row represents the first step of the matching process. After this step, a transition is made from the start state ($c = \top$) to the same state ($c' = \top$). The transition is due to the transition function $\delta^{\overset{\bullet\leftrightarrow}{P}}(c, \langle i, s[t]\rangle)$ returning the value $\top/2$ for the offset variable $i = 1$ and symbol $s[t = 0] = a$. The last entry in the row is empty as the top node does not contain any own objects. The next row shows the transition $\delta^{\overset{\bullet\leftrightarrow}{P}}(\top, \langle 2, a\rangle) = 3/3$ due to value of the variable t being incremented from its value in the previous step. This process continues until the patterns $aabc$, $(a)abc$ are recorded when the variables i and t are both (coincidentally) equal to 4 and state 4 is reached in the 5^{th} row of the table. Recall that a match is recorded for a state that is represented by a concept such that the size of such concept's intent (as shown in the second last entry) is the same as the offset variable i.

c_o	i_o	t	$s[t]$	c	i	a	a	a	b	c	d	a	b	c	c	d	$\|\mathbf{intent}(c')\|$	{Patterns Matched}
$\top$	1	0	a	$\top$	2	T											1	∅
$\top$	2	1	a	3	3		T										2	∅
3	3	2	a	3	3			T									4	∅
3	3	3	b	3	4				T								4	∅
3	4	4	c	4	5					T							4	{aabc,(a)abc}
4	5	5	d	$\top$	1						F						5	∅
$\top$	1	6	a	$\top$	2							T					5	∅
$\top$	2	7	b	7	3								T				3	∅
7	3	8	c	7	4									T			3	{abc}
7	4	9	c	6	5										T		4	{abcc}
6	5	10	d	$\top$	1											F	5	∅

Table 4: Positions visited in *aaabcdabccd* when matching *abc,aabc,abcc*

We have arrived at a particularly efficient algorithm, thanks to two observations about *PAutMatch* . Firstly, transitions in the PEPL-Automaton ($\delta^{\overset{\bullet\leftrightarrow}{L}}$) can be done in constant time using a lookup table. Secondly, the **if** statement can be made in constant time, and consists of simple integer arithmetic to advance through the lattice and target s, and an update of MO (only if a match has been found). The latter can be done using a precomputed lookup table, as is done in the advanced AC algorithm. These two characteristics are also found in the advanced AC algorithm, and is unavoidable in pattern matching algorithms, giving us the same exact (worst- and best-case) running time of $|s|$.

The example in Table 4 illustrates how, in contrast to the example in Table 2, each symbol in the string *aaabcdabccd* is visited exactly once to match the keywords $\{abc, aabc, abcc\}$.

6 Conclusion

The application of FCA in pattern matching was first introduced in [9]. There, two-dimensional pattern information was encoded into a concept lattice which was subsequently used as the basis for traversing a two-dimensional space in search of a specific pattern. Here, by contrast, common information about multiple keywords is encoded into a PEPL, to form the basis for discovering positional information about matching instances of those keywords in a linearly streamed text. The two new pattern matching algorithms are shown to have theoretical running-time comparable to the Aho-Corasick family of algorithms.

Ongoing work involves benchmarking the new algorithms against the Aho-Corasick and other multiple keyword pattern matching algorithms. We are also finding ways in which FCA can be effectively used in other stringology contexts [6].

References

1. Alfred V. Aho and Margaret J. Corasick. Efficient string matching: an aid to bibliographic search. *Communications of the ACM*, 18(6):333–340, June 1975.
2. Loek Cleophas, Bruce W. Watson, and Gerard Zwaan. A new taxonomy of sublinear right-to-left scanning keyword pattern matching algorithms. *Science of Computer Programming*, 75:1095–1112, 2010.
3. Maxime A. Crochemore and Wojciech Rytter. *Text Algorithms*. Oxford University Press, 1994.
4. Maxime A. Crochemore and Wojciech Rytter. *Jewels of Stringology*. World Scientific Publishing Company, 2003.
5. Derrick G. Kourie and Bruce W. Watson. *The Correctness-by-Construction Approach to Programming*. Springer Verlag, 2012.
6. Derrick G. Kourie, Bruce W. Watson, Loek Cleophas, and Fritz Venter. Failure deterministic finite automata. In Jan Holub, editor, *Proceedings of the Prague Stringology Conference (PSC)*. Czech Technical University, August 2012.
7. William F. Smyth. *Computing Patterns in Strings*. Addison-Wesley, 2003.
8. George Varghese. *Network Algorithmics: An Interdisciplinary Approach to Designing Fast Networked Devices*. Morgan Kaufmann, 2004.
9. Fritz Venter, Derrick G. Kourie, and Bruce W. Watson. FCA-based two dimensional pattern matching. In *Proceedings of the 7th International Conference on Formal Concept Analysis*, 2009.
10. Bruce W. Watson. *Taxonomies and Toolkits of Regular Language Algorithms*. PhD thesis, Eindhoven University of Technology, September 1995.

Factor analysis of sports data via decomposition of matrices with grades*

Radim Belohlavek, Marketa Krmelova

Data Analysis and Modeling Lab (DAMOL)
Dept. Computer Science, Palacky University, Olomouc
`radim.belohlavek@acm.org`, `marketa.krmelova@gmail.com`

Abstract. The aim of this paper is to present experimental results on a recently developed method of factor analysis of data with graded, or fuzzy, attributes. The method utilizes formal concepts of data with graded attributes. In our previous papers, we described the factor model, the method, an algorithm to compute factors, and provided basic examples. In this paper, we perform a more extensive experimentation with this method. In particular, we apply the method to factor analysis of sports data. The aim of the paper is to demonstrate that the method yields reasonable factors, explain in detail how the factor model and the factors are to be understood, and to put forward new issues relevant to the method.

1 Introduction

1.1 Aim of This Paper

Recently, a considerable effort was devoted to the development of factor analysis and related methods for new types of data such as Boolean (binary) or ordinal. In our previous papers, we developed a method of factor analysis of Boolean data [5], i.e. data with Boolean attributes, and extended the problem and method to data with graded attributes [4, 6]. In the present paper, we use the method as well as the algorithm from [4, 6]. Due to the limited scope and the aim of this paper, we only provide a brief, mainly informal overview of the key notions involved, illustrate these notions by examples and refer the reader to [4, 6] for technical details. Our aim is to provide information sufficient to understand the experiments described in this paper. A full version of this paper will contain a detailed description of the method, a more comprehensive experimental section, formal treatment of some issues that we only discuss informally in this paper (cf. also Section 3), as well as a section putting the method being discussed into perspective of related methods of data analysis.

* We acknowledge support by the Grant No. P202/10/0262 of the Czech Science Foundation (R. Belohlavek); and by the IGA of Palacky University, No. PrF_20124029 (M. Krmelova).

1.2 The Method, Factors, and Their Interpretation

In a broad sense, our method may be considered as implementing the general idea of factor analysis [1, 12]: Given an $n \times m$ object-attribute matrix I, one finds a decomposition

$$I = A \circ B \tag{1}$$

of I into a product of an $n \times k$ object-factor matrix A, a $k \times m$ matrix B, revealing thus k factors, i.e. new, possibly more fundamental attributes (or variables), which explain the original m attributes. We want $k < m$ and, in fact, k as small as possible to achieve parsimony: The n objects described by m attributes via I may then be described by k factors via A, with B representing a relationship between the original attributes and the factors. Contrary to classic factor analysis, which uses the calculus of real-valued matrices, we use the calculus of matrices over residuated lattices. That is, the entries of matrices involved are elements of a residuated lattice $\mathbf{L} = \langle L, \wedge, \vee, \otimes, \rightarrow, 0, 1\rangle$, i.e. $I_{ij}, A_{il}, B_{lj} \in L$. The elements of L represent truth degrees, 0 and 1 are the smallest and largest one and correspond to "(fully) false" and "(fully) true"; $\wedge$ and $\vee$ denote the infimum and supremum, and $\otimes$ and $\rightarrow$ denote the truth functions on many-valued logic connectives of conjunction and implication. The product $\circ$ in (1) is defined by

$$(A \circ B)_{ij} = \textstyle\bigvee_{l=1}^{k} A_{il} \otimes B_{lj}. \tag{2}$$

Importantly, the entries of I, A, and B are interpreted the following way:

I_{ij} is the truth degree of the proposition "object i has attribute j",
A_{il} is the truth degree of the proposition "factor l applies to object i",
B_{lj} is the truth degree of "attribute j is one of the manifestations of factor l".

For the moment, think of i, j, and l as a particular athlete (object), good performance in long jump (attribute), and good speeding ability (factor). Using the principles of fuzzy logic [11], (2) and hence the whole factor model has the following meaning (this is even easy to see using intuition knowing that "exists" and "and" are modeled by $\bigvee$ and $\otimes$):

object i has attribute j if and only if
there exists factor l such that i has l (or, l applies to i) (3)
and j is one of the particular manifestations of l.

In principle, our method works as follows. We compute from I, using a greedy approximation algorithm from [6], a set

$$\mathcal{F} = \{\langle C_1, D_1\rangle, \ldots, \langle C_k, D_k\rangle\} \subseteq \mathcal{B}(X, Y, I) \tag{4}$$

of formal fuzzy concepts of I, which gives us the decomposition as follows. Put

$$(A_{\mathcal{F}})_{il} = (C_l)(i) \quad \text{and} \quad (B_{\mathcal{F}})_{lj} = (D_l)(j), \tag{5}$$

i.e. $A_{\mathcal{F}}$ is an $n \times k$ matrix in which the lth column consists of grades assigned to objects by the lth concept extent C_l and $B_{\mathcal{F}}$ is a $k \times m$ matrix in which the lth row consists of grades assigned to attributes by the lth intent D_l. Then $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$, i.e. the matrices $A_{\mathcal{F}}$ and $B_{\mathcal{F}}$ induced by $\mathcal{F}$ provide us with a decomposition of I. Moreover, the optimal decompositions (i.e. with least k possible) may in principle be obtained this way, in which sense using formal concepts as factors is an optimal strategy. Note however, that our algorithm computes suboptimal decompositions since the problem to compute an optimal decomposition is an NP-hard optimization problem.

We revisit these notions, particularly in Section 2.1, where we explain the notions involved using a particular example.

2 Examples and Experiments

Our aim in this section is to present the results of selected analyses, and thus to demonstrate a usefulness of the method, as well as to explain in detail the process of analysis, possibly even for a user who is not familiar with the technicalities of the method.

First, we point out some features common to the examples presented below. As the complete residuated lattice, we use as five-element Łukasiewicz chain. That is, the matrix degrees are taken from the set

$$L = \{0, 0.25, 0.5, 0.75, 1\}$$

and the operation $\otimes$ is given by

$$a \otimes b = \max(0, a + b - 1).$$

Many other choices are available, see e.g. [10]. We represent the degrees by shades of gray as follows (this also emphasizes the fact that the truth degrees have a symbolic, rather than numerical, meaning):

□	░	▒	▓	■
0.00	0.25	0.50	0.75	1.00 .

Note that due to the well-known Miller's 7 ± 2 phenomenon [15], small scales with up to $7{\pm}2$ degrees are preferable to use because humans can understand and use such scales easily. For a reader not familiar with basics of many-valued logics let us note that the Łukasiewicz $\otimes$ (such as other many-valued conjunction) may be seen as a natural conjunction-like aggregation: the higher the truth values a and b of propositions A and B, the higher the truth value $a \otimes b$ of the conjunction $A \& B$.

2.1 2004 Olymphic Games Decathlon—Top 5

We start with a detailed description of factor analysis of top 5 athletes in the 2004 Olympic Decathlon and use this example as a reference example in the subsequent sections (this data is also used in [6], but our analysis here is slightly different since we use a different transformation of the athletes' results to grades). Our method is particularly suitable for analyzing such data for the following reasons. The raw data, i.e. the actual results in the ten disciplines of decathlon, can naturally be transformed to data with graded attributes, i.e. to a matrix I. Namely, for every discipline d, one may consider a graded attribute "good performance in d". That is, such an attribute applies to an athlete (object) to a degree to which we consider the performance of the athlete a good performance. This is a natural, generally applicable idea. However, in our case, the IAAF (International Association of Athletics Federations) provides us with decathlon scoring tables (http://www.iaaf.org, IAAF Scoring Tables for Combined Events) using which one transforms the actual results to scores from an ordinal scale, namely the interval of integers $[0, 1, \dots, 1400]$, which is common to all disciplines. For example, the result of 10.75sec in 100m gets 962 points, the result of 204cm in high jump gets 927 points, etc. A table with actual scores may then be transformed to a matrix I with graded attributes using an appropriate set L of truth degrees and an appropriate transformation function.

The top table in Tab. 1 contains the results of top 5 athletes according to the IAAF scoring tables. The second table from the top contains the corresponding matrix I, i.e. the matrix with degrees from the five-element scale L, and the bottom table contains its graphical representation. The transformation from the table with scores to the matrix with degrees from $L = \{0, 0.25, 0.5, 0.75, 1\}$ is accomplished using functions

$$s_j : [0, \dots, 1400] \to L \text{ defined by } s_j(p) = \text{round}\left(\frac{p - L_j}{H_j - L_j}\right)$$

where j is an attribute (discipline), and L_j and H_j are the lowest and the highest scores achieved by all the athletes (i.e. not only the top 5) who participated in the competition, and round is the function rounding the numbers in $[0, 1]$ to their closest values in L. Note that in this competition, we have $L_{10} = 746$, $L_{lj} = 723$, $L_{sp} = 657$, $L_{hj} = 644$, $L_{40} = 673$, $L_{hu} = 755$, $L_{dt} = 622$, $L_{pv} = 673$, $L_{jt} = 598$, $L_{15} = 466$, and $H_{10} = 989$, $H_{lj} = 1050$, $H_{sp} = 873$, $H_{hj} = 944$, $H_{40} = 968$, $H_{hu} = 978$, $H_{dt} = 905$, $H_{pv} = 1035$, $H_{jt} = 897$, $H_{15} = 791$. Therefore, the degree assigned to Sebrle in 400m is $\text{round}(\frac{892-673}{968-673}) = \text{round}(0.74\dots) = 0.75$. The matrix I allows us to interpret the athletes' results verbally. Namely, assigning to the degrees from L linguistic labels such as "not at all" to 0, "little bit" to 0.25, "half" to 0.5, "quite" to 0.75, and "fully" to 1, or the like, one may say that Sebrle's performance in 400m was quite good. Even though we lose some information using such rounding to five degrees, the information preserved still allows us to perform a reasonable analysis, which is shown next.

The algorithm from [6] found a decomposition of I using six factors depicted in Fig. 2. The corresponding decomposition $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$ is depicted in Fig. 1. As explained in Section 1, cf. (5), the columns of $A_{\mathcal{F}}$ corresponding to $F_l = \langle C_l, D_l \rangle$ contain the degrees assigned to the athletes by C_l; likewise for the rows of $B_{\mathcal{F}}$, the attributes, and D_l.

Fig. 2 shows rectangular patterns using which the factors may be visualized. Each rectangular pattern labeled F_l is actually the matrix J_l resulting as the Cartesian product of the extent C_l and the intent D_l of F_l, i.e. we have $(J_l)_{ij} = C_l(i) \otimes D_l(j)$. (For readers familiar with the ordinary FCA, let us note that these patterns are the rectangles corresponding to formal concepts and that in the general situation with degrees, the concepts cannot be uniquely restored from these patterns.)

Table 1. 2004 Olympic Games Decathlon

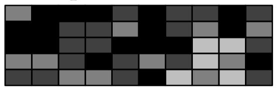
Scores of Top 5 Athletes

	10	*lj*	*sp*	*hj*	40	*hu*	*di*	*pv*	*ja*	15
Sebrle	894	1020	873	915	892	968	844	910	897	680
Clay	989	1050	804	859	852	958	873	880	885	668
Karpov	975	1012	847	887	968	978	905	790	671	692
Macey	885	927	835	944	863	903	836	731	715	775
Warners	947	995	758	776	911	973	741	880	669	693

Matrix I with Graded Attributes (input to the method)

	10	*lj*	*sp*	*hj*	40	*hu*	*di*	*pv*	*ja*	15
Sebrle	0.50	1.00	1.00	1.00	0.75	1.00	0.75	0.75	1.00	0.75
Clay	1.00	1.00	0.75	0.75	0.50	1.00	1.00	0.50	1.00	0.50
Karpov	1.00	1.00	1.00	0.75	1.00	1.00	1.00	0.25	0.25	0.75
Macey	0.50	0.50	0.75	1.00	0.75	0.75	0.75	0.25	0.50	1.00
Warners	0.75	0.75	0.50	0.50	0.75	1.00	0.25	0.50	0.25	0.75

Graphical Representation of Matrix I



Legend: 10—100 meters sprint race; *lj*—long jump; *sp*—shot put; *hj*—high jump; 40—400 meters sprint race; *hu*—110 meters hurdles; *di*—discus throw; *pv*—pole vault; *ja*—javelin throw; 15—1500 meters run.

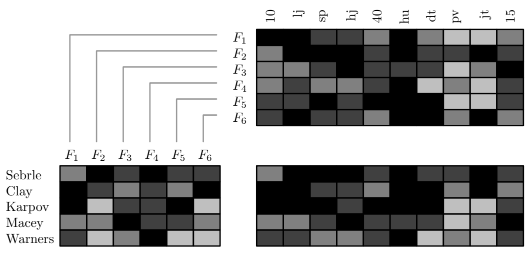


Fig. 1. Decomposition $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$. I, $A_{\mathcal{F}}$, and $B_{\mathcal{F}}$ are the bottom-right, bottom-left, and top matrix, respectively.

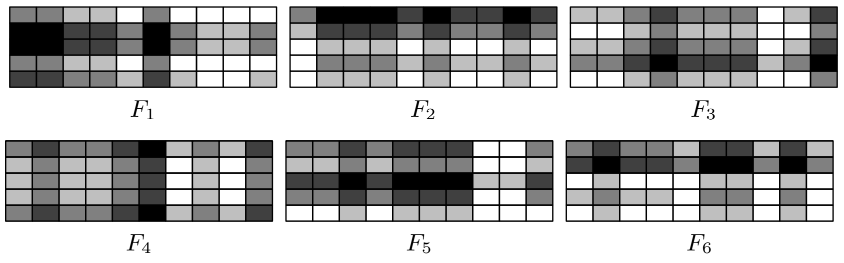


Fig. 2. Factor Concepts as Rectangular Patterns.

Table 2. Factor Concepts

F_i	*Extent*	*Intent*
F_1	{$^{.5}$/Sebrle, Clay, Karpov, $^{.5}$/Macey, $^{.75}$/Warners}	{10, lj, $^{.75}$/sp, $^{.75}$/hj, $^{.5}$/40, *hu*, $^{.5}$/di, $^{.25}$/pv, $^{.25}$/ja, $^{.5}$/15}
F_2	{Sebrle, $^{.75}$/Clay, $^{.25}$/Karpov, $^{.5}$/Macey, $^{.25}$/Warners}	{$^{.5}$/10, lj, sp, hj, $^{.75}$/40, *hu*, $^{.75}$/di, $^{.75}$/pv, ja, $^{.75}$/15}
F_3	{$^{.75}$/Sebrle, $^{.5}$/Clay, $^{.75}$/Karpov, Macey, $^{.5}$/Warners}	{$^{.5}$/10, $^{.5}$/lj, $^{.75}$/sp, hj, $^{.75}$/40, $^{.75}$/*hu*, $^{.75}$/di, $^{.25}$/pv, $^{.5}$/ja, 15}
F_4	{Sebrle, $^{.75}$/Clay, $^{.75}$/Karpov, $^{.75}$/Macey, Warners}	{$^{.5}$/10, $^{.75}$/lj, $^{.5}$/sp, $^{.5}$/hj, $^{.75}$/40, *hu*, $^{.25}$/di, $^{.5}$/pv, $^{.25}$/ja, $^{.75}$/15}
F_5	{$^{.75}$/Sebrle, $^{.5}$/Clay, Karpov, $^{.75}$/Macey, $^{.25}$/Warners}	{$^{.75}$/10, $^{.75}$/lj, sp, $^{.75}$/hj, 40, *hu*, di, $^{.25}$/pv, $^{.25}$/ja, $^{.75}$/15}
F_6	{$^{.75}$/Sebrle, Clay, $^{.25}$/Karpov, $^{.5}$/Macey, $^{.25}$/Warners}	{$^{.75}$/10, lj, $^{.75}$/sp, $^{.75}$/hj, $^{.5}$/40, hu, di, $^{.5}$/pv, ja, $^{.5}$/15}

Fig. 3 demonstrates what portion of matrix I is explained using the first l factors for $l = 1, \dots, k$. In particular, the matrix labeled 56% just shows the rectangular pattern J_1 corresponding to F_1. The number indicates that 56% of the entries in I have the same value as in J_1, i.e. 56% of the data is explained by the first factor. The second matrix contains $J_1 \vee J_2$, i.e. it illustrates what happens when we add the second factor. As we can see, 82% of the data is explained by the first two factors. Since the first three factors explain 91% of the data, one might say that the first three factors account for most of the data, are most important, and the rest of the factors may be omitted. Nevertheless, adding further the factors we see that the first four, five, and six factors explain 95%, 98%, and 100% of the data (the latter fact is obviously true because the factors completely decompose matrix I). Note also, that several of the 18% = 100% − 82% of the entries not explained by the first two factors have values close to the corresponding entries of I, so a measure of closeness of J_l and I which takes into account also close entries, rather than exactly equal ones only, would yield a number larger than 82%. In any case, we can conclude from the visual inspection of the matrices that already the first two or three factors explain the data reasonably well. Note that the fact that the revealed factors are reasonable was confirmed to us by an experienced decathlon coach who also pointed out to us that F_2 (explosiveness) is known to be well-developed by the Czech school of decathlon (hence Sebrle).

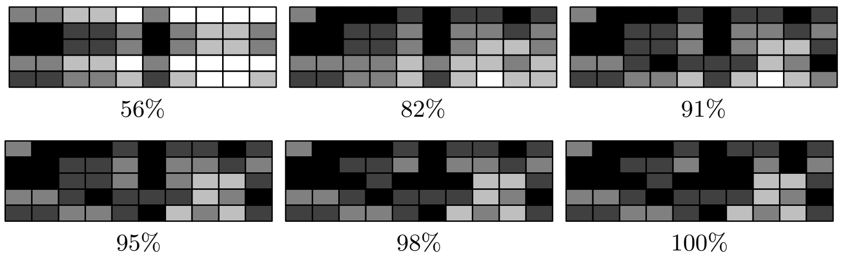


Fig. 3. $\bigvee$-superposition of Factor Concepts

Let us turn to the interpretation of the factors. For this purpose, Fig. 2 is crucial since it contains all the information about the factors. Note however that Fig. 2 is also helpful as it shows the clusters corresponding to the factor concepts which draw together the athletes and their performances in the events. Factor F_1: F_1 applies to Sebrle to degree 0.5, to both Clay and Karpov to degree 1, to Macey to degree 0.5, and to Warners to degree 0.75. Furthermore, this factor applies to attribute 10 (100 m) to degree 1, to attribute lj (long jump) to degree 1, to attribute sp (shot put) to degree 0.75, etc. This means that an excellent performance (degree 1) in 100 m, an excellent performance in long jump, a very good performance (degree 0.75) in shot put, etc. are particular manifestations of this factor. On the other hand, only a relatively weak performance (degree 0.25) in javelin throw and pole vault are manifestations of this factor. All the manifestations of this factor with degree 1 are 100 m, long jump, and 110 m hurdles. This factor can be interpreted as the ability to run fast for short distances. Note that this factor applies particularly to Clay and Karpov which is well known in the world of decathlon. Factor F_2: Similarly, since the manifestations of this factor with degree 1 are long jump, shot put, high jump, and javelin, F_2 can be interpreted as the ability to apply very high force in a very short term (explosiveness). F_2 applies particularly to Sebrle, and then to Clay, who are known for this ability. Factor F_3: Manifestations with grade 1 are high jump and 1500 m. This factor is typical for lighter, not very muscular athletes. Macey, who is evidently that type among decathletes (196 cm and 98 kg) is the athlete to whom the factor applies to degree 1. These are the most important factors behind data matrix I.

2.2 2004 Olympic Games Decathlon Top 5 By Their Best Results

In this example, we take the top 5 athletes of the 2004 Olympic Decathlon but we take their best performances during their decathlon competitions, instead of their actual performances in a single event such as the 2004 Olympics. Taking best performances may be reasonable if we want to avoid a possible bad luck in a particular discipline such as a bad start in 100 m. Tab. 3 contains the scores. The corresponding matrix I and its decomposition into $A_{\mathcal{F}} \circ B_{\mathcal{F}}$ is depicted in Fig. 4. Here, the transformation from points to degrees is defined as follows. For discipline j, we put

$$s_j(p) = \begin{cases} 1 & \text{for } p \in [H_j, H_j - 100), \\ 0.75 & \text{for } p \in [H_j - 100, H_j - 200), \\ 0.5 & \text{for } p \in [H_j - 200, H_j - 300), \\ 0.25 & \text{for } p \in [H_j - 300, H_j - 400), \\ 0 & \text{for } p \leq H_j - 400, \end{cases}$$

where H_j is the highest score ever achieved during a decathlon competition for discipline j. Note that $H_{10} = 1042$; $H_{lj} = 1117$; $H_{sp} = 1048$; $H_{hj} = 1061$; $H_{40} = 1025$; $H_{hu} = 1064$; $H_{di} = 993$; $H_{pv} = 1152$; $H_{ja} = 1040$; $H_{15} = 963$.

It seems natural that the factors in this case are different from those in the example in Section 2.1. Nevertheless, we can see that F_1 applies to degree 1 to Clay and Karpov in both examples and applies to the other athletes to similar degrees in both examples as well. Nevertheless, the intents of the first factor are different although a reasonable similarity is apparent as well (presence of long

Table 3. 2004 Olympic Games Decathlon

Scores of Top 5 Athletes

	10	*lj*	*sp*	*hj*	40	*hu*	*di*	*pv*	*ja*	15
Sebrle	942	1089	880	944	921	1002	859	972	907	798
Clay	1010	1050	868	887	944	1022	993	941	920	670
Karpov	931	1073	910	915	968	984	929	1004	743	729
Macey	940	1002	841	944	998	931	836	849	799	990
Warners	947	1022	800	831	978	973	824	886	692	693

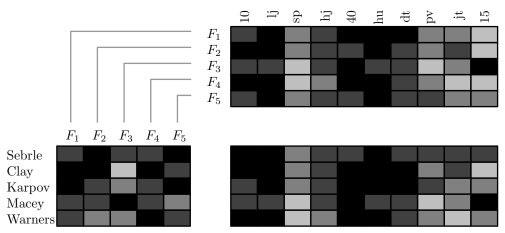


Fig. 4. Decomposition $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$.

jump and hurdles to degree 1, presence of 100 m and high jump to high degrees). A similar observation can be made on F_2 (connects Sebrle and Clay) and F_3 which is typical of Macey.

2.3 2004 Olympic Games Decathlon—Top 10

The results of the 5th–10th athletes in the 2004 Olympic Decathlon are depicted in Tab. 4. The matrix I corresponding to the top 10 athletes, along with a decomposition $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$ computed by the algorithm is depicted in Fig. 5. The same transformation from scores to degrees was used as in Section 2.1.

Table 4. 2004 Olympic Games Decathlon

Scores of the 5th–10th Athletes

	10	*lj*	*sp*	*hj*	40	*hu*	*di*	*pv*	*ja*	15
Zsivoczky	881	847	809	915	842	856	780	819	790	748
Hernu	867	859	768	831	874	942	761	849	704	782
Nool	906	942	744	698	870	874	706	1035	758	704
Bernard	931	930	777	915	855	953	762	731	667	704
Schwarzl	865	932	729	749	826	942	714	941	683	721

Compared to the factors from Section 2.1, the factors in this example are generally different although some similarities are apparent. For example, factor

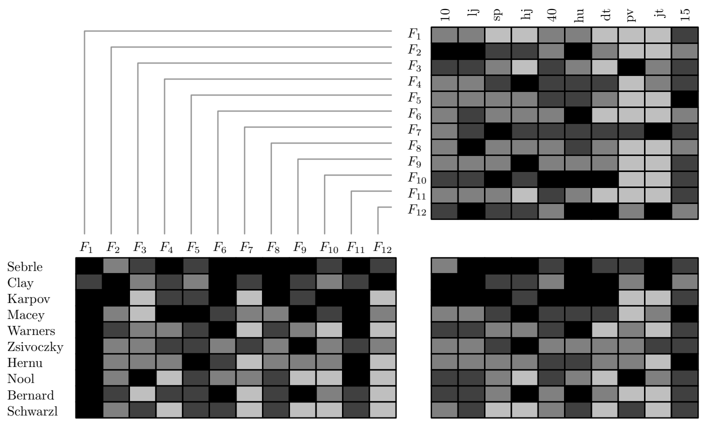


Fig. 5. Decomposition $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$.

F_2 here is exactly the same (has same intent) as F_1 in Section 2.1, F_{12} is the same as F_6 in Section 2.1, and F_4 is almost the same as F_3 in Section 2.1.

We might nevertheless be interested in the question of how well the factors from Section 2.1 explain the new dataset regarding the top 10 athletes. A reasonable way to proceed is the following. Consider the set of 6 concepts of the new, 10×10 matrix I, namely,

$$\mathcal{G} = \{G_1 = \langle P_1, Q_1 \rangle, \ldots, G_6 = \langle P_6, Q_6 \rangle\}$$

obtained from the factors $F_1 = \langle C_1, D_1 \rangle, \ldots, F_6 = \langle C_6, D_6 \rangle$ by

$$P_1 = D_1^{\downarrow}, Q_1 = P_1^{\uparrow}, \ldots, P_6 = D_6^{\downarrow}, Q_6 = P_6^{\uparrow},$$

i.e. every factor G_l is the concept of the 10×10 matrix I generated by the intent of F_l. This way, we do not have $I = A_{\mathcal{G}} \circ B_{\mathcal{G}}$ in general, as can be seen from this example. Nevertheless, the first factor G_1 explains 50% of the data, the first two factors 69%, the first three factors 80%, the first four factors 86%, the first five factors 89%, and all factors in $\mathcal{G}$ explain 91% of the data. Hence, one may conclude that the factors of the top 5 athletes explain reasonably well also the results of all the top 10 athletes. The matrices involved are depicted in Fig. 6. Note that one may clearly observe the similarity between I (the original matrix) and $A_{\mathcal{G}} \circ B_{\mathcal{G}}$ (the matrix reconstructed from the factors in $\mathcal{G}$).

2.4 2004 Olympic Games Modern Pentathlon

Another sport that contains several disciplines and may be interesting for factor analysis is the modern pentathlon. The five disciplines are, however, rather diverse and it is therefore challenging to think of natural factors in this sport. Recall that modern pentathlon consists of pistol shooting, fencing, 200 m freestyle swimming, show jumping, and a 3 km cross-country run. Except for the fencing competition, athletes do not directly compete against one another in the five

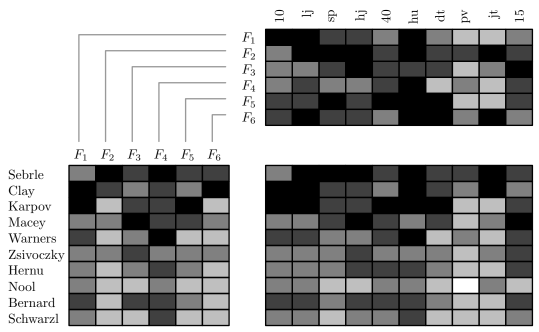


Fig. 6. Matrices $A_{\mathcal{G}} \circ B_{\mathcal{G}}$ (bottom-right), $A_{\mathcal{G}}$ (bottom-left), and $B_{\mathcal{G}}$ (top).

events. Instead, a better absolute performance results in a higher score and the sum of all the scores for the disciplines gives the overall total score of a given athlete.

Tab. 5 contains the results of the 2004 Olympic Games modern pentathlon of the top 10 athletes. The corresponding matrix I and its decomposition into $A_{\mathcal{F}} \circ B_{\mathcal{F}}$ is depicted in Fig. 7. To transform the scores of discipline j to degrees, we used the function

$$s_j(p) = \begin{cases} 1 & \text{for } p \in [H_j, H_j - \frac{1}{5}(H_j - L_j)), \\ 0.75 & \text{for } p \in [H_j - \frac{1}{5}(H_j - L_j), H_j - \frac{2}{5}(H_j - L_j)), \\ 0.5 & \text{for } p \in [H_j - \frac{2}{5}(H_j - L_j), H_j - \frac{3}{5}(H_j - L_j)), \\ 0.25 & \text{for } p \in [H_j - \frac{3}{5}(H_j - L_j), H_j - \frac{4}{5}(H_j - L_j)), \\ 0 & \text{for } p \leq H_j - \frac{4}{5}(H_j - L_j), \end{cases}$$

where H_j and L_j are the highest and the lowest score achieve in discipline j in the 2004 Olympic Games modern pentathlon. Note that $H_{sh} = 1168$, $L_{sh} = 892$; $H_{fe} = 1000$, $L_{fe} = 664$; $H_{sw} = 1376$, $L_{sw} = 1140$; $H_{ri} = 1172$, $L_{ri} = 584$; $H_{ru} = 1116$, $L_{ru} = 752$.

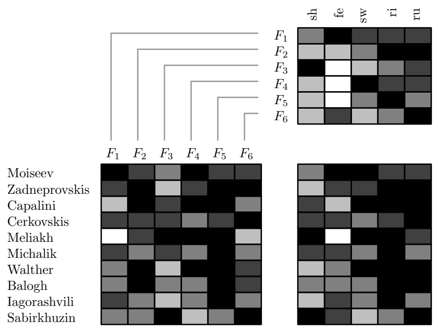


Fig. 7. Decomposition $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$.

Table 5. 2004 Olympic Games Modern Pentathlon

Scores of Top 10 Athletes

	sh	*fe*	*sw*	*ri*	*ru*
Moiseev	1036	1000	1376	1032	1036
Zadneprovskis	1000	916	1308	1088	1116
Capalini	1084	776	1336	1116	1080
Cerkovskis	1096	916	1252	1004	1088
Meliakh	1168	692	1332	1144	1004
Michalik	1108	888	1260	1144	932
Walther	952	832	1336	1116	1084
Balogh	1036	804	1240	1172	1044
Iagorashvili	988	916	1252	1172	948
Sabirkhuzin	1156	888	1216	908	1034

Legend: *sh*—shooting; *fe*—fencing; *sw*—swimming; *ri*—riding; *ru*—running.

Note that of all the factors computed, F_2 is probably most interesting because it is actually known in the world of modern pentathlon. Namely, F_2's manifestations are riding and cross-country run which is typical for athletes who are in a good physical shape and have good endurance. Each of the other factors more or less corresponds to a single discipline which corresponds to the intuitive idea that the disciplines are diverse and require diverse skills.

3 Concusions, Further Issues and Future Work

We presented several examples of factor analysis of sports data using a recently developed method that utilizes formal concepts as factors. Our main aim was to explain the method, to illustrate the key notions used in the method, and to demonstrate how one can understand the results of the method. It turns out from the examples that the method yields reasonable factors and that the results of the method are easy to understand.

Due to the limited scope of this paper, we presented only a limited number of examples and limited comments on the presented examples. In addition to the examples presented in this paper, we performed factor analyses of further decathlon data (namely, the World Championships), figure skating, and ice hockey players performance. We refrained from formalizing some of the issues involved, such as "explanation of data by factors", "similarity of factors", how well the factors of one dataset serve as good factors of another dataset, etc., and used these notions with their informal meaning only. We therefore also skipped theoretical results regarding these notions, as well as further notions and results that may help us answer further natural questions that arise in the context of the presented method, such as the influence of the choice of the scale of degrees, the operation $\otimes$, the influence of the transformation from scores to degrees, and the like.

More examples with detailed comments as well as a detailed treatment of some of issues mentioned above will appear in the full version of this paper. An interesting question that is to be a subject of our future research is a comparison, experimental and possibly also theoretical, of relationships of the presented method with related methods that involve matrix decomposition, notable the non-negative matrix factorization [8, 14].

References

1. Bartholomew D. J., Knott M.: *Latent Variable Models and Factor Analysis, 2nd Ed.*, London, Arnold, 1999.
2. Bartl E., Belohlavek R., Konecny J.: Optimal decompositions of matrices with grades into binary and graded matrices. Annals of Mathematics and Artificial Intelligence 59(2)(2010), 151–167.
3. Belohlavek R.: Concept lattices and order in fuzzy logic. *Annals of Pure and Applied Logic* **128**(1–3)(2004), 277–298.
4. Belohlavek R.: Optimal decompositions of matrices with entries from residuated lattices. Journal of Logic and Computation, doi: 10.1093/logcom/exr023, online: September 7, 2011.
5. Belohlavek R., Vychodil V.: Discovery of optimal factors in binary data via a novel method of matrix decomposition. Journal of Computer and System Sciences 76(1)(2010), 3–20.
6. Belohlavek R., Vychodil V.: Factor analysis of incidence data via novel decomposition of matrices. LNAI 5548(2009), 83-97.
7. Ganter B., Wille R.: *Formal Concept Analysis. Mathematical Foundations.* Springer, Berlin, 1999.
8. Golub G., Van Loan C.: *Matrix Computations.* Johns Hopkins University Press, 1996.
9. J. A. Goguen. The logic of inexact concepts. *Synthese* **18** (1968–9), 325–373.
10. Gottwald S.: *A Treatise on Many-Valued Logics.* Research Studies Press, Baldock, Hertfordshire, England, 2001.
11. Hájek P.: *Metamathematics of Fuzzy Logic.* Kluwer, Dordrecht, 1998.
12. Harman H. H. : *Modern Factor Analysis, 2nd Ed.* The Univ. Chicago Press, Chicago, 1970.
13. Krantz H. H., Luce R. D., Suppes P., Tversky A.: *Foundations of Measurement.* Vol. I (Additive and Polynomial Representations), Vol. II (Geometric, Threshold, and Probabilistic Represenations), Vol. III (Represenations, Axiomatization, and Invariance). Dover Edition, 2007.
14. Lee D., Seung H.: Learning the parts of objects by non-negative matrix factorization. *Nature* **401**(1999), 788–791.
15. Miller G. A.: The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychol. Rev.* **63**(1956), 81–97.
16. Stockmeyer L. J.: The set basis problem is NP-complete. IBM Research Report RC5431, Yorktown Heights, NY, 1975.
17. Ward M., Dilworth R. P.: Residuated lattices. *Trans. Amer. Math. Soc.* **45** (1939), 335–354.
18. Zadeh L. A.: Fuzzy sets. *Inf. Control* **8**(1965), 338–353.

Impact of Boolean factorization as preprocessing methods for classification of Boolean data[⋆]

Radim Belohlavek, Jan Outrata, Martin Trnecka

Data Analysis and Modeling Lab (DAMOL)
Dept. Computer Science, Palacky University, Olomouc
radim.belohlavek@acm.org, jan.outrata@upol.cz, martin.trnecka@gmail.com

Abstract. The paper explores a utilization of Boolean factorization as a method for data preprocessing in classification of Boolean data. In previous papers, we demonstrated that data preprocessing consisting in replacing the original Boolean attributes by factors, i.e. new Boolean attributes that are obtained from the original ones by Boolean factorization, improves the quality of classification. The aim of this paper is to explore the question of how the various Boolean factorization methods that were proposed in the literature impact the quality of classification. In particular, we compare three factorization methods, present experimental results, and outline issues for future research.

1 Problem Setting

In classification of Boolean data, the objects to classify are described by Boolean (binary, yes-no) attributes. As with the other classification problems, one may be interested in preprocessing of the input attributes to improve the quality of classification. With Boolean input attributes, we might want to limit ourselves to preprocessing with a clear semantics. Namely, as it is known, see e.g. [2, 8], applying to Boolean data the methods designed originally for real-valued data distorts the meaning of the data and leads generally to results difficult to interpret. In [9, 10], we proposed a method for preprocessing Boolean data based on the Boolean matrix factorization (BMF) method, i.e. a decomposition method for Boolean matrices, developed in [2]. The method consists in using for classification of the objects new Boolean attributes. The new attributes are actually the factors computed from the original attributes. Since the factors are essentially (some of the) formal concepts [4] associated to the input data, they have a clear meaning and are easy to interpret [2]. Moreover, there exists a natural transformation of the objects between the space of the original attributes and the space of the factors [2] which is conveniently utilized by the method. It has been demonstrated in [9, 10] that such preprocessing makes it possible to classify

⋆ We acknowledge support by the ESF project No. CZ.1.07/2.3.00/20.0059, the project is co-financed by the European Social Fund and the state budget of the Czech Republic (R. Belohlavek); Grant No. 202/10/P360 of the Czech Science Foundation (J. Outrata); and by the IGA of Palacky University, No. PrF_20124029 (M. Trnecka).

using a smaller number of input variables (factors instead of original attributes) and yet improve the quality of classification. In addition to the method from [2], there exist several other BMF methods described in the literature. In the present paper, we therefore look at the question of how these methods influence the quality of classification. In particular, we focus on three such methods and provide an experimental evaluation using basically the same scenario as in [9, 10]. Doing so, we emphasize the need to consider not only coverage and the number of extracted factors, but also additional criteria regarding quality of the proposed BMF methods.

We use the following notation. We denote by $X = \{1, \ldots, n\}$ a set of objects which are given along with their input Boolean attributes which form the set $Y = \{1, \ldots, m\}$, and a class attribute c. The input attributes are described by an $n \times m$ Boolean matrix I with entries I_{ij} (entry at row i and column j), i.e. $I_{ij} \in \{0, 1\}$ for every i, j. Alternatively, I may be considered as a representation of a binary relation between X and Y and, hence, we may speak of a formal context $\langle X, Y, I\rangle$, etc. Since there is no danger of confusion, we conveniently switch between the matrix and relational way of looking at things. The class attribute c may be conceived as a mapping $c : X \to C$ assigning to every object $i \in X$ its class label $c(i)$ in the set C of all class tables (note that C may contain more than two labels).

The preprocessing method along with the three particular methods of Boolean matrix factorization is described in Section 2. Section 3 describes the experiments and provides their results. In Section 4 we conclude the paper and provide some directions for future research.

2 Boolean Matrix Factorization and Its Utilization

2.1 General BMF Problem

We denote by $\{0, 1\}^{n \times m}$ the set of all $n \times m$ Boolean matrices and by $I_{i\_}$ and $I_{\_j}$ the ith row and jth column, respectively, of matrix I. In BMF, the general aim is to find for a given $I \in \{0, 1\}^{n \times m}$ (and possibly other parameters, see Problem 1 and Problem 2) matrices $A \in \{0, 1\}^{n \times k}$ and $B \in \{0, 1\}^{k \times m}$ for which

$$I \text{ is (approximately) equal to } A \circ B, \tag{1}$$

with $\circ$ being the Boolean matrix product given by

$$(A \circ B)_{ij} = \bigvee_{l=1}^{k} A_{il} \cdot B_{lj}, \tag{2}$$

where $\bigvee$ denotes the maximum and $\cdot$ the ordinary product. Such an exact or approximate decomposition of I into $A \circ B$ corresponds to a discovery of k factors (new Boolean variables) that exactly or approximately explain the data. Namely, factor $l = 1, \ldots, k$, may be represented by $A_{\_l}$ (column l of A) and $B_{l\_}$ (row l of B): $A_{il} = 1$ indicates that factor l applies to object i while B_{lj} indicates that attribute j is a particular manifestation of factor l (think of person A as object, "being fluent in English" as attribute, and "having good education" as factor). The least k for which an exact decomposition $I = A \circ B$ exists is called the Boolean (or Schein) rank of I [2, 5, 8]. Then, according to (2), the factor model reads "object i has attribute j if and only if there exists factor l such that l applies to i and j is a particular manifestation of l".

The matrices I, A, and B are usually called the object-attribute matrix, the object-factor (or usage) matrix, and the factor-attribute (or basis vector) matrix [2, 8]. The methods described in the literature are usually designed for two particular problems. Consider the matrix metric [5, 8] (arising from the L_1-norm $||\cdot||$ of matrices, or Hamming weight in case of Boolean matrices) given by

$$E(C, D) = ||C - D|| = \textstyle\sum_{i=1,j=1}^{m,n} |C_{ij} - D_{ij}|. \tag{3}$$

$E(I, A \circ B)$ may be used to asses how well the product $A \circ B$ approximates the input matrix I.

Problem 1

input: $I \in \{0, 1\}^{n\times m}$, positive integer k
output: $A \in \{0, 1\}^{n\times k}$ and $B \in \{0, 1\}^{k\times m}$ minimizing $||I - A \circ B||$.

This problem is called the *discrete basis problem* (DBP) in [8]. In [2], the following problem is considered:

Problem 2

input: $I \in \{0, 1\}^{n\times m}$, positive integer ε
output: $A \in \{0, 1\}^{n\times k}$ and $B \in \{0, 1\}^{k\times m}$ with k as small as possible such that $||I - A \circ B|| \leq \varepsilon$.

The two problems reflect two important views on BMF, the first one emphasizing the importance of the first k (presumably most important) factors, the second one emphasizing the need to account for (and thus to explain) a prescribed portion of data. Note that the problem of finding an exact decomposition of I with the least number k of factors possible is a particular instance of Problem 2 (put $\varepsilon = 0$). Note also that it follows from the known results that both Problem 1 and Problem 2 are NP-hard optimization problems, see e.g. [2, 8], and hence approximation algorithms are needed to obtain (suboptimal) solutions.

2.2 Use of BMF in Preprocessing of Boolean Data

The idea may be described as follows. For a given set X of objects, set Y of attributes, Boolean matrix I, and class attribute c, we compute $n \times k$ and $k \times m$ Boolean matrices A and B, respectively, for which $A \circ B$ approximates I reasonably well (either according to the scenario given by Problem 1 or Problem 2). Then, instead of the original instance $\langle X, Y, I, c\rangle$ of the classification problem, we consider a new instance given by $\langle X, F, A, c\rangle$, with $F = \{1, \ldots, k\}$ denoting the factors, i.e. new Boolean attributes. Any classification model developed for $\langle X, F, A, c\rangle$ may then be used to classify the objects described by the original

Boolean attributes from Y. Namely, one may utilize natural transformations $g : \{0,1\}^m \to \{0,1\}^k$ and $h : \{0,1\}^k \to \{0,1\}^m$ between the space of the original attributes and the space of factors which are given by

$$(g(P))_l = \bigwedge_{j=1}^{m}(B_{lj} \to P_j) \quad \text{and} \quad (h(Q))_j = \bigvee_{l=1}^{k}(Q_l \cdot B_{lj})$$

for $P \in \{0,1\}^m$ and $Q \in \{0,1\}^k$ ($\bigwedge$ and $\to$ denote minimum and implication). These transformations are described in [2] to which we refer for more information. In particular, given an object represented by $P \in \{0,1\}^m$ (vector of values of the m input attributes), we apply the classification method developed for $\langle X, F, A, c\rangle$ to $g(P)$, i.e. to the object representation in the space of factors. Any classification model $M_F : \{0,1\}^k \to C$ for $\langle X, F, A, c\rangle$ therefore induces a classification model $M_Y : \{0,1\}^m \to C$ by $M_Y(P) = M_F(g(P))$ for any $P \in \{0,1\}^m$.

Note that since the number k of factors of I is usually smaller than the number m of attributes (see [2], which means a reduction of dimensionality of data) and the transformation of objects from the attribute space to the factor space is not an injective mapping, we need to solve the problem of assigning a class label to objects in $\langle X, F, A, c\rangle$ with equal $g(P)$ representations transformed from objects in $\langle X, Y, I, c\rangle$ with different P representations and different assigned class labels. We adopt the common solution of assigning to such objects in $\langle X, F, A, c\rangle$ the majority class label of class labels assigned to the objects in $\langle X, Y, I, c\rangle$.

2.3 Three Methods for Boolean Matrix Factorization Used in Our Experiments

Asso [8] works as follows. From the input $n \times m$ matrix I, the required number k of factors, and parameters τ, w^+, and w^-, the algorithm computes an $m \times m$ matrix C in which $C_{ij} = 1$ if the confidence of the association rule $\{i\} \Rightarrow \{j\}$ is at least τ. The rows of C are then the candidate rows for matrix B. The actual k rows of B are selected from the rows of C in a greedy manner using parameters w^+ and w^-. During the greedy selection, the k columns of A are selected along with the k rows of B. This way, one obtains from I two matrices A and B such that $A \circ B$ approximates I. Asso is designed for Problem 1. There is no guarantee that Asso computes an exact factorization of I even for $k = m$, see [3]. In our experiments, we used $\tau = 1$ and $w^+ = w^- = 1$ because such choice guarantees that for $k = m$ all the 1s in I will be covered by the computed factors.

GreConD This algorithm, described in [2] where it is called Algorithm 2, utilizes formal concepts of I as factors. Namely, the algorithm is selecting formal concepts of I, one by one, until a decomposition of I into $A \circ B$ is obtained. The selected formal concepts are utilized in a simple way: The (characteristic vectors of the) extents and intents of the concepts form the columns and rows of A and B. The algorithm may be stopped after computing the first k concepts or whenever $||I - A \circ B|| \leq \varepsilon$, i.e. the algorithm may be used for solving Problem 1 as well as Problem 2. The formal concepts are selected in a greedy manner to maximize the drop of the error function, in particular, on demand way, whence the name Gre(edy)Con(concepts on)D(emand).

GreEssQ This algorithm [3] utilizes formal concepts of I in the same way as GreConD. The concepts are selected in a greedy manner, but contrary to GreConD, the concepts are selected using a particular heuristic that is based on the information provided by certain intervals in the concept lattice of I. As with GreConD, GreEssQ may be used to solve both Problem 1 and 2.

3 Experiments

We performed a series of experiments to evaluate the impact of the three Boolean matrix factorization methods described in Section 2.3 on classification of Boolean data when using factors as new attributes. The experiments consisted in comparing the classification accuracy of learning models created by selected machine learning (ML) algorithms from the data with the original attributes replaced by factors. The factors are computed from the input data by the three selected factorization methods. The ML algorithms used in the comparison are: the reference decision tree algorithms ID3 and C4.5 (entropy and information gain based), an instance based learning method (Nearest Neighbor, NN), Naive Bayes learning (NB) and a multilayer perceptron neural network trained by back propagation (MLP) [7, 11]. The algorithms were borrowed and run from Weka[1], a software package that contains implementations of machine learning and data mining algorithms in Java. Default Weka's parameters were used for the algorithms.

Table 1. Characteristics of datasets used in experiments

Dataset	*No. of attributes (binary)*	*No. of objects*	*Class distribution*
breast-cancer	9(51)	277	196/81
kr-vs-kp	36(74)	3196	1669/1527
mushroom	22(125)	282	187/95
vote	16(32)	232	124/108
zoo	15(30)	101	41/20/5/13/4/8/10

The experiments were done on selected public real-world datasets from UCI Machine Learning Repository [1]. The selected datasets are from different areas (medicine, biology, zoology, politics, games). All the datasets contain only categorical attributes with one class attribute and the datasets were cleared of objects containing missing values. Basic characteristics of the datasets are depicted in Table 1 (note that the mushroom dataset was shrunk in the number of objects due to computation time reasons). Note that "9(51)" means 9 categorical and 51 binary attributes obtained by nominal scaling. The classification accuracy is evaluated using the 10-fold stratified cross-validation test [6] and the

[1] Waikato Environment for Knowledge Analysis, available at `http://www.cs.waikato.ac.nz/ml/weka/`

following results are based on averaging 10 execution runs on each dataset with randomly ordered objects.

The results are depicted in Figures 1 to 5. Each figure contains five graphs for the five ML algorithms used. The graphs show the average percentage rates of correct classifications on the preprocessed data, i.e. the data $\langle X, F, A, c\rangle$ described by factors instead of $\langle X, Y, I, c\rangle$ described by the original attributes, cf. Sections 2.2 and 2.3 for each of the three Boolean matrix factorization methods. The percentage rates of GreEssQ, GreConD, and Asso are depicted by the dashed, dot-and-dashed, and dotted lines, respectively. The x-axis corresponds to the factor decompositions obtained by the algorithms and, in particular, measures the quantity

$$\frac{|\langle i, j\rangle\,;\, I_{ij} = 1 \text{ and } (A \circ B)_{ij} = 0|}{|\langle i, j\rangle\,;\, I_{ij} = 1|},$$

i.e. the relative error w.r.t. 1s of the input matrix I that are left uncovered in $A \circ B$ for the computed factorization given by A and B. The values on the x-axis range from 0.9 (corresponding to a factorization with a small number of factors that leave 90 % of the 1s in I uncovered) to 0 (corresponding to the number of factors which decompose I exactly, i.e. $I = A \circ B$). The average percentage rate of correct classification for the original data $\langle X, Y, I, c\rangle$ is depicted in each graph by a constant solid line. All the graphs are computed for the testing parts of the datasets used in the evaluation of classification only.

We can clearly see from the graphs for all datasets but breast-cancer that the best results (average percentage rates of correct classifications) for preprocessed data are obtained, for all ML algorithms used, by the GreEssQ algorithm, outperforming both GreConD and, quite significantly, the Asso algorithm. GreConD outperforms the Asso algorithm, again for all ML algorithms used, for datasets kr-v-kp and mushroom, but not for the vote dataset. We can also see from the graphs that sometimes the preprocessed data lead to a better classification accuracy than the original data even with a few factors covering less than 100 % of input data. This can be seen for instance for the kr-vs-kp dataset and Nearest Neighbor and MLP or the mushroom dataset and ID.3, Naive Bayes and MLP. See [9, 10] for indications of when, i.e. for which datasets and ML algorithms, the data with original attributes replaced by factors (computed by GreConD) covering 100 % of input data leads to a better classification accuracy compared to the original data.

Particularly interesting seem the results for the breast-cancer dataset. As we can see, the preprocessed data with factors instead of the original attributes are (much) better classified compared to the original data and that this is observable for all ML algorithms used except for Naive Bayes. Furthermore, the number of factors leading to the best average percentage rates of correct classifications is such that the factors cover just 40 % (which corresponds to 0.6 on the x-axis) of input data! This indicates either many superfluous attributes or large noise in the input data that is overcome by using the factors. The GreEssQ and GreConD algorithms are of comparable performance here, both outperforming the Asso algorithm.

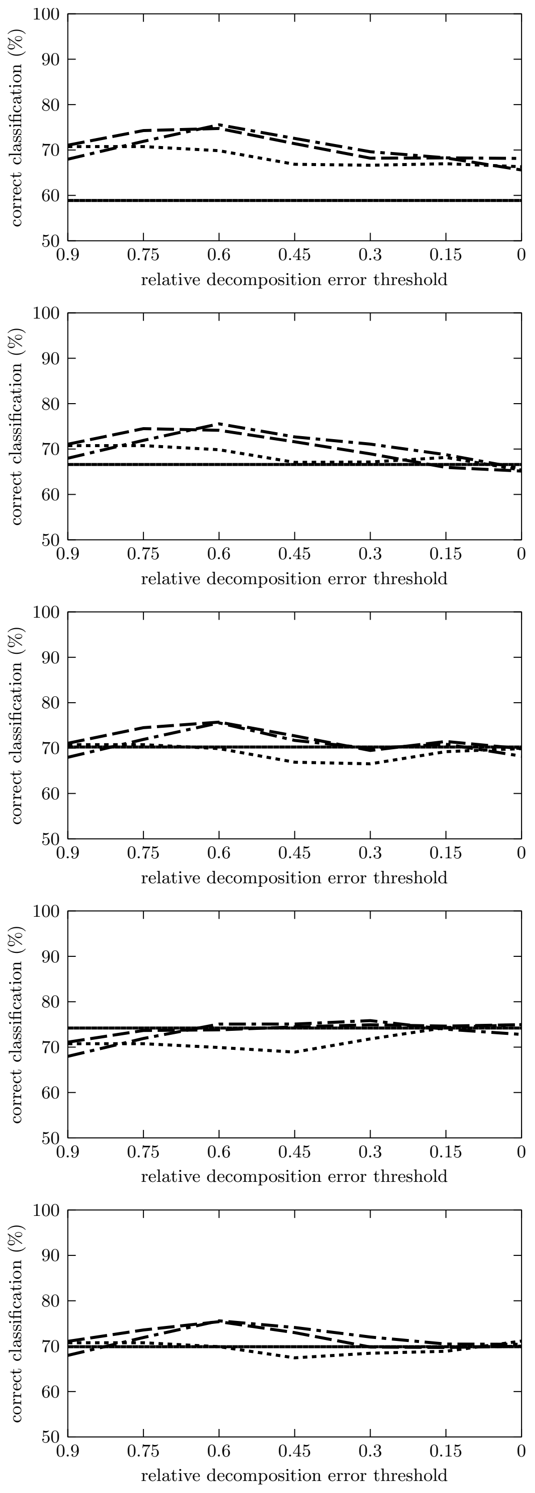


Fig. 1. Classification accuracy for breast-cancer dataset, for (from top to bottom) ML algorithms ID.3, C4.5, NN, NB and MLP

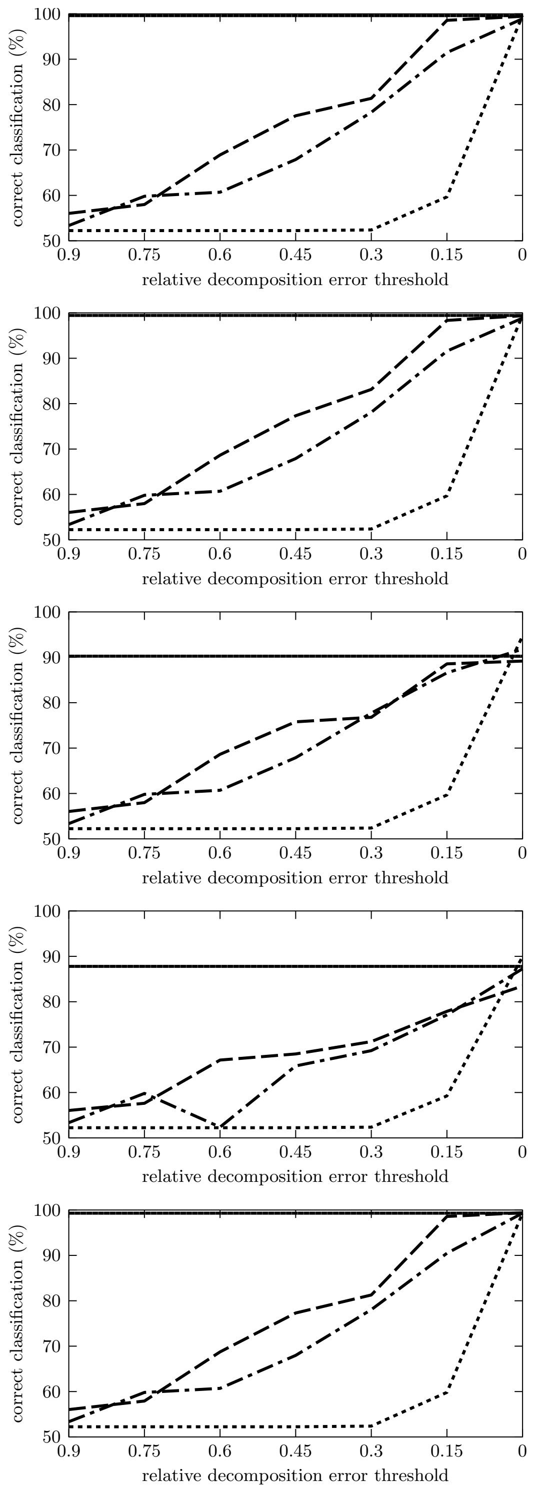


Fig. 2. Classification accuracy for kr-vs-kp dataset, for (from top to bottom) ML algorithms ID.3, C4.5, NN, NB and MLP

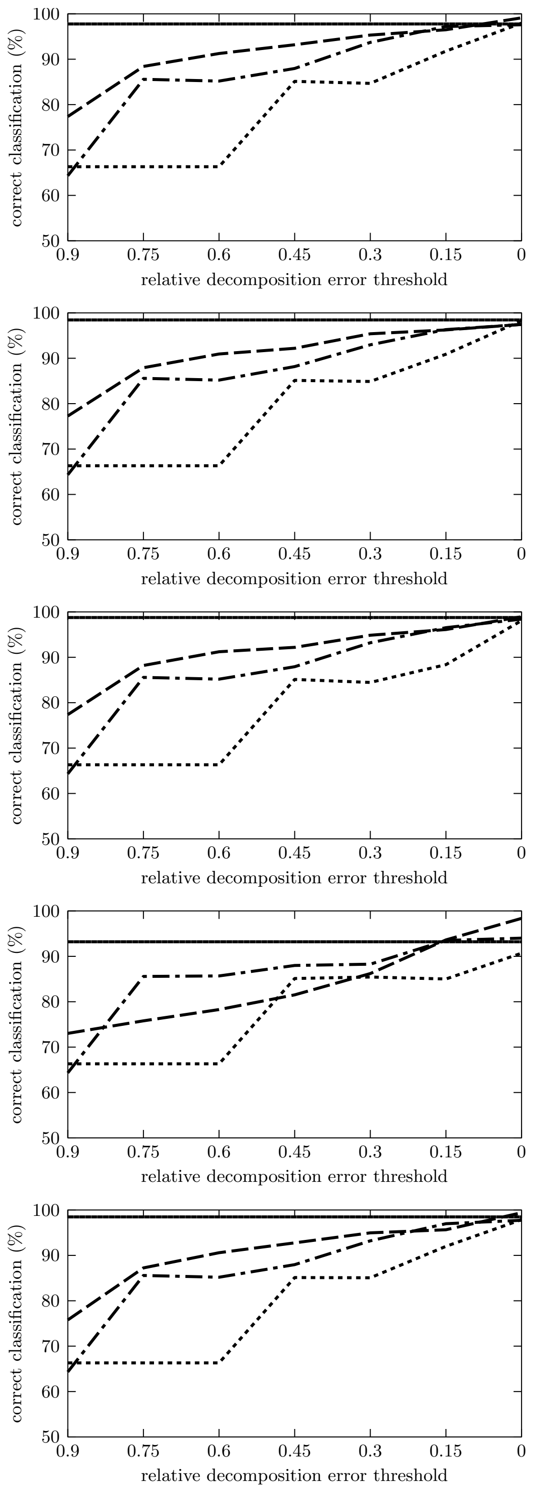


Fig. 3. Classification accuracy for mushroom dataset, for (from top to bottom) ML algorithms ID.3, C4.5, NN, NB and MLP

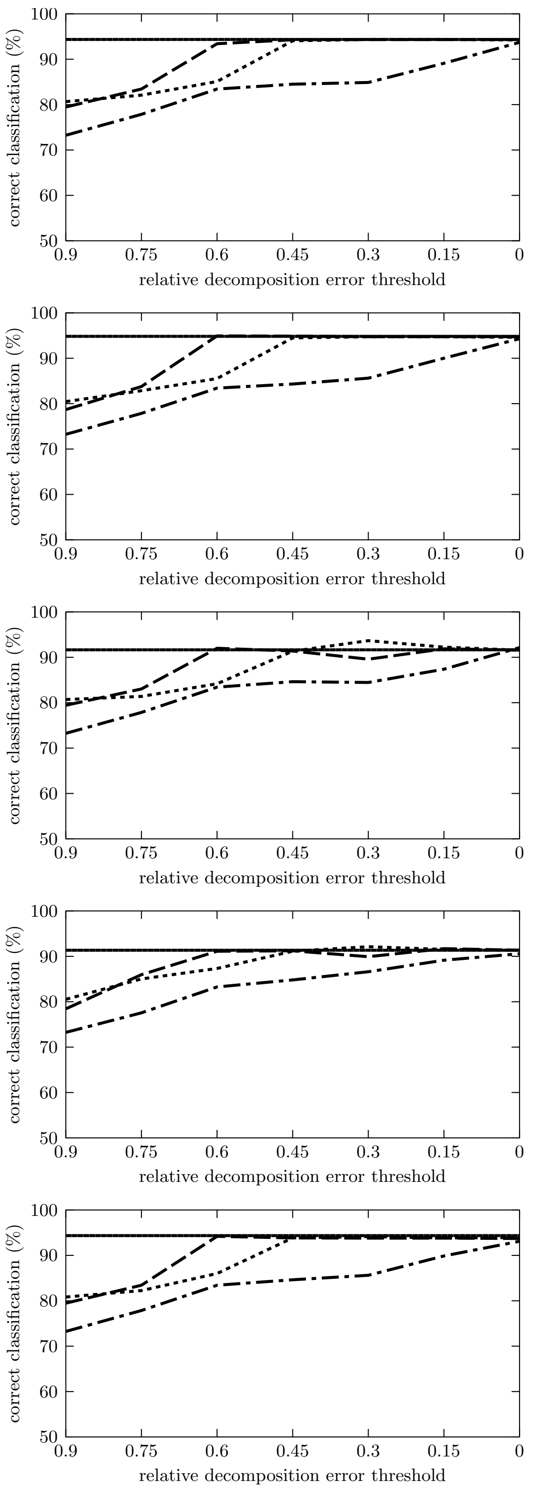


Fig. 4. Classification accuracy for vote dataset, for (from top to bottom) ML algorithms ID.3, C4.5, NN, NB and MLP

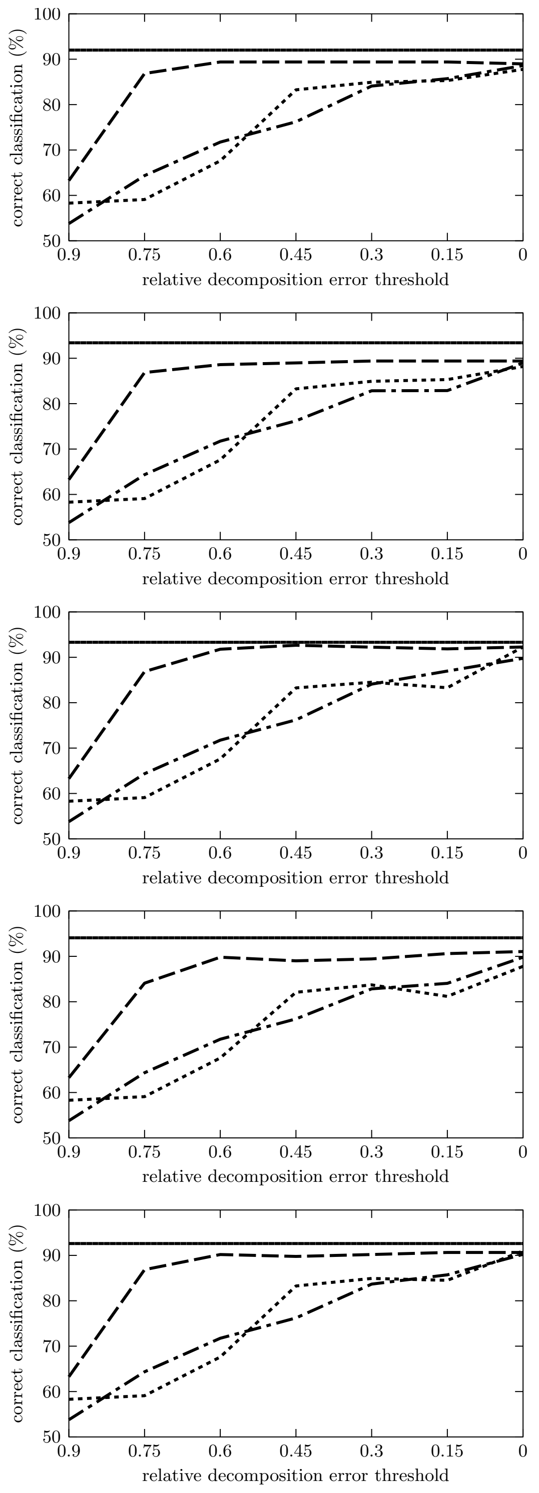


Fig. 5. Classification accuracy for zoo dataset, for (from top to bottom) ML algorithms ID.3, C4.5, NN, NB and MLP

4 Conclusions

We presented an experimental study which shows that when Boolean matrix factorization is used as a preprocessing technique in Boolean data classification in the scenario proposed in [9, 10], the particular factorization algorithms impact in a significant way the accuracy of classification. For this purpose, we compared three such algorithms from the literature. In addition to demonstrating further the usefulness of Boolean factorization for classification of Boolean data, the paper emphasizes Boolean factorization as a data dimensionality reduction technique that may be utilized in a similar way as the matrix-decomposition-based methods designed for real-valued data.

An extended version of this paper will include further factorization algorithms in the experimental comparison (let us note in this respect that a technical problem with some such algorithms is that they are poorly described in the literature). Furthermore, we intend to investigate and utilize further appropriate transformation functions between the attribute and the factor spaces, in particular those suitable for approximate factorizations. A comparison with other data dimensionality techniques, see e.g. the references in [12], in the presented scenario is also an important topic for future research. In this respect, both the impact on the classification accuracy as well as the transparency of the resulting classification model are important aspects to be evaluated in such a comparison.

References

1. Asuncion A., Newman D. J., *UCI Machine Learning Repository.* University of California, Irvine, School of Information and Computer Sciences, 2007. `http://www.ics.uci.edu/~mlearn/MLRepository.html`
2. Belohlavek R., Vychodil V.: Discovery of optimal factors in binary data via a novel method of matrix decomposition. J. Computer and System Sci. 76(1)(2010), 3–20.
3. Belohlavek R., Trnecka M.: From-below approximations in Boolean matrix factorization: geometry, heuristics, and new BMF algorithm (submitted).
4. Ganter B., Wille R.: *Formal Concept Analysis. Mathematical Foundations.* Springer, Berlin, 1999.
5. Kim K. H.: *Boolean Matrix Theory and Applications.* M. Dekker, 1982.
6. Kohavi R.: A Study on Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. Proc. IJCAI1995, pp. 1137–1145.
7. Mitchell T. M.: *Machine Learning.* McGraw-Hill, 1997.
8. Miettinen P., Mielikäinen T., Gionis A., Das G., Mannila H., The Discrete Basis Problem. IEEE Trans. Knowledge and Data Eng. 20(10)(2008), 1348–1362 (preliminary version in PKDD 2006, pp. 335–346.)
9. Outrata J.: Preprocessing input data for machine learning by FCA. Proc. CLA 2010, pp. 187–198, Sevilla, Spain.
10. Outrata J.: Boolean factor analysis for data preprocessing in machine learning. Proc. ICML 2010, pp. 899-902, Washington, D.C., USA.
11. Quinlan J. R.: *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.
12. Tatti N., Mielikäinen T., Gionis A., Mannila H.: What is the dimension of your binary data? Proc. ICDM 2006, pp. 603–612.

The lattice of all betweenness relations : Structure and properties

Laurent Beaudou, Mamadou Moustapha Kanté, and Lhouari Nourine

Clermont Université, Université Blaise Pascal, LIMOS, CNRS, France
`laurent.beaudou@univ-bpclermont.fr, {mamadou.kante,nourine}@isima.fr`

Abstract. We consider implication bases with premises of size exactly 2, which are also known as *betweenness* relations. Our motivations is that several problems in graph theory can be modelled using betweenness relations, e.g. hull number, maximal cliques. In this paper we characterize the lattice of all betweenness relations by giving its poset of *irreducible* elements. Moreover, we show that this lattice is a meet-sublattice of the lattice of all *closure systems.*

1 Introduction

A *convexity space* on a ground set X is a subset of 2^X that is closed under intersection. Convexity spaces were studied in [13] and are sometimes called *Closure systems.* The members of a convexity space are called *convex sets.* Since the paper [13], convexity spaces are studied by several authors who describe several of their properties (see the joint paper [8] of Edelman and Jamison for a list of publications during the eighties), in particular the set of convexity spaces forms a *lattice.*

In this paper we deal with *betweenness* relations which are special cases of convexity spaces. The notion of betweenness relation has appeared in the early twentieth century when mathematicians focused on fundamental geometry [4]. A *betweenness* relation B on a finite set X is a set of triples $(x, y, z) \in X^3$. The most intuitive betweenness relations are those coming from metric spaces (a point y is between x and z if they satisfy the triangular equality). A *convex set* of a betweenness relation B is a subset Y of X such that for all $(x, y, z) \in B$, if $\{x, z\} \subseteq Y$, then $y \in Y$. It is well-known that the set of convex sets of a betweenness relation is a convexity space. Betweenness relations have been thoroughly studied by Menger and his students [16]. Other betweenness relations have arisen in research fields as probability theory with the work of Reichenbach [18]. Betweenness relations have been also studied in graphs in order to generalize geometrical theorems [1,2,9] (see the survey [17] for a non exhaustive list of betweenness relations on graphs).

We are interested in describing the set of all betweenness relations on a ground set X. We prove that the set of all convexity spaces on X derived from betweenness relations on X forms a lattice and is in fact a *meet-sublattice* of the lattice of all convexity spaces on X (we give an example showing that it is not a sub-lattice). We also describe the set of *meet* and *join-irreducible* elements of the lattice. We conclude by showing that the set of convexity spaces obtained from *clique betweenness* relations on graphs is a sublattice of the lattice of all convexity spaces of betweenness relations.

This paper is motivated by understanding links between several parameters that are considered in different areas such as FCA, database, logic and graph theory.

Summary. Notations and definitions are given in Section 2. The description of the lattice of convexity spaces of betweenness relations is given in Section 3. The convexity spaces of clique betweenness relations is described in Section 4. Some questions arising from algorithmic aspects are given in Section 5.

2 Preliminaries

Let X be finite set. A *partially ordered set* on X (or *poset*) is a reflexive, anti-symmetric and transitive binary relation denoted by $P := (X, \leq)$. For $x, y \in X$, we say that y covers x, denoted by $x \prec y$, if for any $z \in X$ with $x \leq z \leq y$ we have $x = z$ or $y = z$. A *lattice* $L := (X, \leq)$ is a partially ordered set with the following properties:

1. for all $x, y \in X$ there exists a unique z, denoted by $x \vee y$, such that for all $t \in X$, $t \geq x$ and $t \geq y$ implies $z \leq t$. (Upper bound property.)
2. for all $x, y \in X$ there exists a unique z, denoted by $x \wedge y$, such that for all $t \in X$, $t \leq x$ and $t \leq y$ implies $z \geq t$. (Lower bound property.)

Let $L = (X, \leq)$ be a lattice. An element $x \in X$ is called *join-irreducible* (resp. *meet-irreducible*) if $x = y \vee z$ (resp. $x = y \wedge z$) implies $x = y$ or $x = z$. A join-irreducible (resp. meet-irreducible) element covers (resp. is covered by) exactly one element. We denote by J_L and M_L the set of all join-irreducible and meet-irreducible elements of L respectively.

The poset of irreducible elements of a lattice $L = (X, \leq)$ is a representation of L by a bipartite poset $Bip(L) = (J_L, M_L, \leq)$. The concept lattice of $Bip(L)$ is isomorphic to L (for more details see the books of Davey and Priestley [3], and Ganter and Wille [10]).

An implication on X is an ordered pair (A, B) of subsets of X, denoted by $A \to B$. The set A is called the premise and the set B the conclusion of the implication $A \to B$. Let Σ be a set of implications on X. A subset $Y \subseteq X$ is Σ-closed if for each implication $A \to B$ in Σ, $A \subseteq Y$ implies $B \subseteq Y$. The closure of a set S by Σ, denoted by S^{Σ}, is the smallest Σ-closed set containing S.

Let S be a subset of X. Algorithm 1 computes the closure of S by a betweenness relation Σ. It is known as *forward chaining procedure* or *chase procedure* [11].

The set of Σ-closed subsets of X, denoted by F_{Σ}, is a *closure system* on X (i.e closed under set-intersection), and when ordered under inclusion is a lattice.

Algorithm 1: Set Closure(S, Σ)

```
Data: A set $S \subseteq X$ and $\Sigma$ a betweenness
Result: The closure $S^\Sigma$
begin
    Let $S^\Sigma := S$;
    while $\exists xy \to z \in \Sigma$ s.t. $\{x, y\} \subseteq S^\Sigma$ and $z \notin S^\Sigma$ do
        $S^\Sigma = S^\Sigma \cup \{z\}$;
end
```

Conversely, given a closure system $\mathcal{F}$ on X, a family Σ of implications on X is called an implicational basis for $\mathcal{F}$ if $\mathcal{F} = F_\Sigma$. A subset $K \in X$ is called a *key* if $K^\Sigma = X$ and K is minimal under inclusion with this property. The name key comes from database theory [15].

Definition 1. *An implication set Σ on X is called a* betweenness relation *if for all $A \to B \in \Sigma$, $|A| = 2$.*

Two betweenness relations Σ_1 and Σ_2 are said to be *equivalent*, denoted by $\Sigma_1 \equiv \Sigma_2$, if $F_{\Sigma_1} = F_{\Sigma_2}$. We define the *closure* of a betweenness relation Σ by $\Sigma^c = \{ab \to c \mid a, b, c \in X \text{ and } \Sigma \equiv \Sigma \cup \{ab \to c\}\}$. Note that Σ^c is the unique maximal betweenness relation equivalent to Σ. In each equivalence class we distinguish two types of betweenness relations:

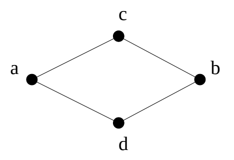
Canonical A *canonical betweenness* is the maximum in its equivalence class.
Optimal A betweenness Σ is *optimal* if for any betweenness relation Σ' equivalent to Σ, we have $|\Sigma| \leq |\Sigma'|$.

A graph G is a pair $(V(G), E(G))$, where $V(G)$ is the set of vertices and $E(G)$ is the set of edges. We consider simple graphs (for further definitions see the book [6]). Examples of betweenness relations arising from graph theory are $\Sigma_G = \{xy \to z \mid z$ lies in a shortest path from x to y$\}$ and $\Sigma_G = \{xy \to V(G) \mid xy \in E(G)\}$. Several other notions of convexity spaces are defined on graphs (see [17]). Figure 1 gives an example of a graph and its convex sets for the shortest path betweenness.

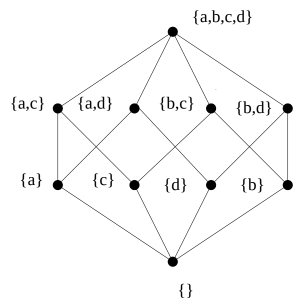
3 The Lattice of all Betweenness Relations

Let X be a finite set and Σ a betweenness relation on X. Given two sets A and C in 2^X such that $A \subseteq C$, we define the *set interval* $[A, C]$ as the family of all sets B in 2^X such that $A \subseteq B \subseteq C$.

Demetrovics *et al.* [5] gave a characterization of convex sets of an implication basis. Proposition 1 is restricted to betweenness relations.



(a) A graph G



(b) Convex sets of G for shortest path betweenness

Fig. 1. A graph and its convex sets for the shortest path betweenness relation

Proposition 1. *[5] Let Σ be a betweenness relation on a set X. Then,*

$$F_\Sigma = 2^X \setminus \bigcup_{ab\to c\in\Sigma} [\{a,b\}, X \setminus \{c\}].$$

We denote by $\mathbb{F}_X := \{F_\Sigma \mid \Sigma \text{ is a betweenness relation on } X\}$ the family of all betweenness relations on X.

Theorem 1. *$\mathbb{F}_X$ is a closure system and therefore a lattice when structured under inclusion.*

Proof. We have to prove that this structure is closed under the intersection and it contains a unique maximal element.

Let $F_1, F_2 \in \mathbb{F}_X$, then there exist Σ_1 and Σ_2 inducing these families of convex sets on X. Let $F = F_1 \cap F_2$ and Σ be the betweenness defined by $ab \to c \in \Sigma$ if $ab \to c \in \Sigma_1$ or $ab \to c \in \Sigma_2$. Then we claim that $F = F_\Sigma$.

Let C be a set in F. It is convex for Σ_1 and for Σ_2. Therefore, for any a, b in C, every c such that $ab \to c \in \Sigma_1$ or $ab \to c \in \Sigma_2$ is in C. From this, we derive that for any a, b in C, every c such that $ab \to c \in \Sigma$ is in C, so that C is convex for Σ.

Reciprocally, let C be a convex set for Σ. We will show that it is convex for Σ_1 and Σ_2. Let a, b, c be elements of X such that a and b are in C and $ab \to c \in \Sigma_1$. Then $ab \to c \in \Sigma$ and since C is convex for Σ, b is in C. Therefore, C is in F_1. Similarly it is in F_2 so that it is in F.

Since $\Sigma = \Sigma_1 \cup \Sigma_2$, we conclude that Σ is a betweenness relation and therefore the set $\mathbb{F}_X$ is closed under intersection.

The family 2^X is in $\mathbb{F}_X$. It arises when Σ is the empty betweenness relation.

Proposition 2. *Given two families F_1 and F_2 in $\mathbb{F}_X$ and their canonical betweenness relations Σ_1 and Σ_2. Then*

– $F_1 \wedge F_2 = F_{\Sigma_1 \cup \Sigma_2}$.
– $F_1 \vee F_2 = F_{\Sigma_1 \cap \Sigma_2}$.

Proof. See the proof of Theorem 1 for the first point. Noww, note that Σ_1 and Σ_2 are canonical betweenness relations. Suppose that $F_1 \vee F_2$ is not $F_{\Sigma_1 \cap \Sigma_2}$. By Proposition 1, we have $F_1 \vee F_2 \subseteq F_{\Sigma_1 \cap \Sigma_2}$.

Now call $\Sigma_\vee$ the canonical betweenness relation related to $F_1 \vee F_2$. If $ab \to c \in \Sigma_\vee$ for some a, b and c in X, then we know that $ab \to c \in \Sigma_i$ because $F_i \subseteq F_1 \vee F_2$ for $i = 1, 2$ (in the canonical form, we have every $ab \to c$ such that the corresponding interval has no intersection with F). Therefore, every $ab \to c \in \Sigma_\vee$ is true for Σ, and $F_{\Sigma_1 \cap \Sigma_2} = F_1 \vee F_2$.

Corollary 1. *$\mathbb{F}_X$ is a meet-sublattice of the lattice of all closure systems on X.*

Proof. Let F_1 and F_2 be two families in $\mathbb{F}_X$. Since $F_1 \cap F_2$ is the closure system of a betweenness relation then the meet is preserved and thus $\mathbb{F}_X$ is a a meet-sublattice of the lattice of all closure systems on X.

Remark 1. Notice that $\mathbb{F}_X$ is not a sublattice of the lattice of all closure systems on X. It suffices to consider the example where $X = \{1, 2, 3, 4\}$. Take the co-atoms $F_1 = 2^X \setminus [\{1, 2\}, \{1, 2, 3\}]$ defined by the betweenness relation restricted to $12 \to 4$ and $F_2 = 2^X \setminus [\{2, 3\}, \{1, 2, 3\}]$ defined by the betweenness relation restricted to $23 \to 4$. Then $F_1 \cup F_2 = 2^X \setminus \{\{1, 2, 3\}\}$ and is a closure system, while $F_1 \vee F_2$ is the top element, 2^X.

In the following, we give a characterization of the poset of irreducible elements of $\mathbb{F}_X$.

Proposition 3. *The poset of irreducible elements of $\mathbb{F}_X$ is the bipartite poset $Bip(\mathbb{F}_X) = (J_{\mathbb{F}_X}, M_{\mathbb{F}_X}, \subseteq)$ where*

$$J_{\mathbb{F}_X} := \{F_\perp \cup \{S\} \mid S \in 2^X \setminus F_\perp\} \text{ where } F_\perp = \{\emptyset, X\} \cup \{\{x\} \mid x \in X\}$$
$$M_{\mathbb{F}_X} := \{2^X \setminus [ab, X \setminus \{c\}] \mid a, b, c \in X\}.$$

Proof. We prove this proposition point by point.

Consider the maximal betweenness relation $\Sigma = \{ab \to c \mid a, b, c \in X\}$. Then $F_\perp = F_\Sigma = 2^X \setminus \bigcup_{ab \to c \in \Sigma} [\{a, b\}, X \setminus \{c\}]$ (see Proposition 1). Thus $F_\perp = \{\emptyset, X\} \cup \{\{x\} \mid x \in X\}$.

For meet-irreducible elements, we will first consider co-atoms. Let Σ be a betweenness relation such that F_Σ is a co-atom. Since Σ is non-empty, then there exists a set which is not convex. Thus Σ must contain an implication $ab \to c$, with $a, b, c \in X$, which corresponds to the maximal closure system in $\mathbb{F}_X$ and different from 2^X. Namely, we remove from 2^X the convex sets of the interval $[\{a, b\}, X \setminus \{c\}]$.

Now suppose there exists another meet-irreducible element F which is not a co-atom. Call Σ_1 a betweenness relation such that F is F_{Σ_1}. Also, call F' the only successor of F_{Σ_1} and Σ_2 a betweenness such that F' is F_{Σ_2}. By Proposition 1, we know that from F' to F, we remove at least an interval of the form $[\{a,b\}, X \setminus \{c\}]$. Thus, the co-atom associated to the implication $ab \to c$ is not above F' but it is above F, so that F has at least two successors. We conclude that all meet-irreducible elements are co-atoms of $\mathbb{F}_X$.

For join-irreducible elements, we will first characterize atoms of $\mathbb{F}_X$. Pick any subset S of X which is not empty, a singleton or the whole of X. Define the betweenness relation Σ_S such that $ab \to c \in \Sigma_S$ for every a, b, c in X except those where a and b are in S and c is not in S. Then F_{Σ_S} is $F_\perp \cup \{S\}$. Now suppose there exists a join-irreducible $F \in \mathbb{F}_X$ that is not an atom. F contains at least one set S which is not in the unique closure system $F' \in \mathbb{F}_X$ that it covers. But there is an atom which contains exactly $F" = F_\perp \cup \{S\}$, with $F" \subseteq F$ and $F" \not\subseteq F'$. Thus F covers at least two elements, and thus F is not a join-irreducible element.

Corollary 2. *$\mathbb{F}_X$ contains $\binom{n}{2}(n-2)2^{n-3}$ meet-irreducible and $2^n - (n+2)$ join-irreducible elements.*

Proof. Every co-atom is of the form $2^X \setminus [\{a,b\}, X \setminus \{c\}]$ and is above every atom formed by $F_\perp$ and any set not in the forbidden interval. This makes $2^n - (n+2) - 2^{n-3}$ atoms below it.

An atom of the form $F_\perp \cup S$ where S is a set on X of size 2 to $n-1$, is below every co-atom that does not forbid S. This number equals $\binom{n}{2}(n-2)2^{n-3} - [\binom{|S|}{2}(n-|S|)]$.

Figure 2 shows the irreducible poset for $X = \{1,2,3,4\}$ where every atom is represented by the set S added to $F_\perp$ and every co-atom is represented by the removed implication $xy \to z$.

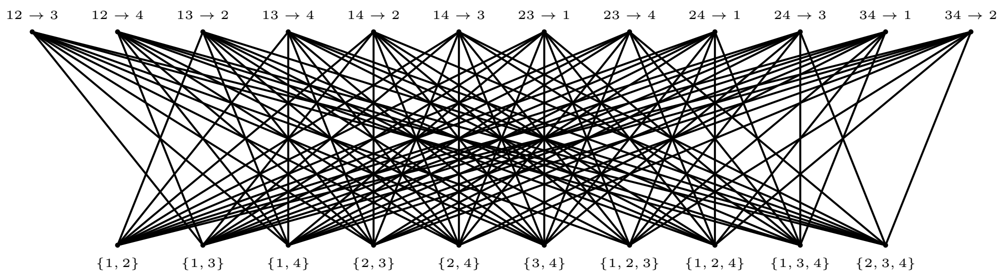


Fig. 2. Irreducible poset for $n = 4$

4 Clique Betweenness Relations

In this section we deal with a special betweenness relation defined through a graph in a specific way. Many other betweenness relations on graphs can be defined in the same way, e.g. independent sets, set covers.

Given a graph G, we define the betweenness relation

$$\Sigma_G := \{ab \to c \mid c \in X,\ ab \notin E(G)\}.$$

The convex sets of Σ_G are exactly the cliques of G. Notice that for any graph G, there is a corresponding betweenness relation Σ_G. In the following, we characterize the lattice of all Σ_G where G is a graph with vertex-set X. We denote by $\mathbb{F}_X^K = \{F_{\Sigma_G} \mid G \text{ is a graph on } X\}$.

Proposition 4. *$(\mathbb{F}_X^K, \subseteq)$ is a lattice.*

Proof. The bottom (resp. top) element of $\mathbb{F}_X$ corresponds to F_{Σ_G} where G is a stable (resp. clique) on X.

Moreover, $\mathbb{F}_X^K$ is closed under intersection (the cliques of the intersection of two graphs are exactly the ones in the intersection of both families of cliques). Therefore, $(\mathbb{F}_X^K, \subseteq)$ is a lattice.

Since Σ_G is a betweenness relation, then $\mathbb{F}_X^K \subset \mathbb{F}_X$ for any graph G defined on X.

Proposition 5. *The lattice $(\mathbb{F}_X^K, \subseteq)$ is a sublattice of $(\mathbb{F}_X, \subseteq)$.*

Proof. It is easy to see that it is a meet-sublattice, the meet of two families is the intersection of both families in both structures.

In order to prove that it is a join-sublattice of $(\mathbb{F}_X, \subseteq)$, consider two graphs G_1 and G_2 and their clique families F_1 and F_2. Call G the graph union of G_1 and G_2 and F the family of its cliques. The related betweenness relation is obtained by taking the intersection of betweenness relations related to G_1 and G_2 (non-edges in G are exactly non-edges in G_1 and in G_2). Therefore it is the join of F_1 and F_2 in $(\mathbb{F}_X, \subseteq)$.

Corollary 3. *The lattice $(\mathbb{F}_X^K, \subset)$ is a boolean lattice with $\binom{n}{2}$ atoms.*

Proof. For each graph G, its corresponding canonical betweenness relation is the set $\Sigma_G^c := \{ab \to X \mid ab \notin E(G)\}$. Note that any super-set of Σ_G^c corresponds to a betweenness relation of a partial graph of G, by deleting edges ab from G which corresponds to adding $ab \to X$ in Σ_G^c. Since any atom of $(\mathbb{F}_X^K, \subset)$ corresponds to a betweenness relation which contains exactly an implication $ab \to X$, we have exactly $\binom{n}{2}$ atoms.

5 Algorithmic Aspects of Betweenness Relations

In this section, we recall some optimization problems related to betweenness relations.

Minimum Key (MK)
Input: Σ a betweenness relation on X and k an integer.
Question: Is there a set $K \subseteq X$ such that $|K| \leq k$ and $K^{\Sigma} = X$?

These problems have been studied in the several domains and specially in database theory, and they have been proved NP-complete [5,14,15] for general implication bases. The problem MK has been proved NP-complete for particular cases of betweenness relations (see for instance [7] for the shortest path betweenness relation on graphs). It is known as the *hull number* of a betweenness relation. Therefore, we have the following.

Proposition 6. *MK is NP-complete.*

Recently, Kanté and Nourine [12] have shown that MK is polynomial for shortest path betweenness relations of chordal and distance hereditary graphs by using database techniques. Can we use the lattice structure of $\mathbb{F}_X$ to get new polynomial time algorithms for the MK problem in new graph classes?

Now we consider the problem which computes an optimal cover of a betweenness relation. This problem is to find an optimal betweenness relation which is equivalent to a given betweenness relation. Several works have been done in the general case known as *Horn minimization* [11].

Optimal Cover (OC)
Input: Σ a betweenness relation on X and k an integer.
Question: Is there a betweenness relation Σ' equivalent to Σ such that $|\Sigma'| \leq k$?

The size of an optimal cover is known as the *hydra number* [19]. The computational complexity of the hydra number is open for betweenness relations, but is NP-complete for the general case [11,15]. We hope that the lattice structure of $\mathbb{F}_X$ could help to address this question.

6 Conclusion

In this paper, we characterize the context of the lattice of all betweenness relations on a finite set, which is a meet-sublattice of the lattice of all closure systems on the same set. We are convinced that the structure of lattice can help to understand some problems of graph theory such as hull number and hydra number. In the future we will investigate the link of these parameters and the structure of the lattice.

References

1. Xiaomin Chen and Vašek Chvátal. Problems related to a de Bruijn-Erdős theorem. *Discrete Applied Mathematics*, 156(11):2101–2108, 2008.
2. Vašek Chvátal. Antimatroids, betweenness, convexity. In *Cook, W.J., Lováz, L., Vygen, J. (eds.) Research Trends in Combinatorial Optimization*, pages 57–64, 2019.
3. B. A. Davey and A. Priestley. *Introduction to Lattices and Order.* Cambridge University Press, 2002.
4. G. de Beauregard Robinson. *The foundations of geometry.* Mathematical expositions. University of Toronto Press, 1952.
5. János Demetrovics, Leonid Libkin, and Ilya B. Muchnik. Functional dependencies in relational databases: A lattice point of view. *Discrete Applied Mathematics*, 40(2):155–185, 1992.
6. Reinhardt Diestel. *Graph Theory.* Springer-Verlag, 3rd edition, 2005.
7. Mitre Costa Dourado, John G. Gimbel, Jan Kratochvíl, Fábio Protti, and Jayme Luiz Szwarcfiter. On the computation of the hull number of a graph. *Discrete Mathematics*, 309(18):5668–5674, 2009.
8. P.H. Edelman and R.E. Jamison. The theory of convex geometries. In R. Rustin, editor, *Geometriae Dedicata, vol 19, Ni3*, pages 247–270. springer, 1985.
9. Martin Farber and Robert E. Jamison. Convexity in graphs and hypergraphs. *SIAM Journal on Algebraic and Discrete Methods*, 7:433–444, 1986.
10. B. Ganter and R. Wille. *Formal concept analysis: Mathematical foundations.* Springer (Berlin and New York), 1999.
11. Peter L. Hammer and Alexander Kogan. Optimal compression of propositional horn knowledge bases: Complexity and approximation. *Artif. Intell.*, 64(1):131–145, 1993.
12. M.M. Kanté and L. Nourine. Polynomial time algorithms for computing a minimum hull set in distance-hereditary and chordal graphs. *Submitted*, 2012.
13. D.C. Kay and E.W. Womble. Axiomatic convexity theory and relationships between the Carathéodory, Helly, and Radon numbers. *Pacific Journal of Mathematics*, 38:471–485, 1971.
14. Claudio L. Lucchesi and Sylvia L. Osborn. Candidate keys for relations. *Journal of Computer and System Sciences*, 17(2):270 – 279, 1978.
15. David Maier. Minimum covers in relational database model. *J. ACM*, 27(4):664–674, 1980.
16. Karl Menger. Untersuchungen über allgemeine metrik. *Mathematische Annalen*, 100:75–163, 1928. 10.1007/BF01448840.
17. Ignacio M. Pelayo. On convexity in graphs. Technical report, Universitat Politècnica de Catalunya, http://www-ma3.upc.es/users/pelayo/research/Definitions.pdf, 2004.
18. H. Reichenbach and M. Reichenbach. *The Direction of Time.* California Library Reprint Series. University of California Press, 1956.
19. Despina Stasi, Robert H. Sloan, and György Turán. Hydra formulas and directed hypergraphs: A preliminary report. In *ISAIM*, 2012.

Using Taxonomies on Objects and Attributes to Discover Generalized Patterns

Léonard Kwuida[1], Rokia Missaoui[2*], Jean Vaillancourt[2]

[1] Bern University of Applied Sciences
kwuida@gmail.com
[2] Université du Québec en Outaouais
{rokia.missaoui,jean.vaillancourt}@uqo.ca

Abstract. In this paper, we show how the existence of taxonomies on objects and/or attributes can be used in formal concept analysis to help discover *generalized* patterns in the form of concepts. To that end, we analyze three generalization cases and different scenarios of a simultaneous generalization on both objects and attributes.
We also contrast the number of generalized patterns against the number of simple patterns.

1 Introduction

In many real-life applications and research trends in Computer Science, the semantics of data can be advantageously exploited to better retrieve and efficiently manage information and discover unexpected and relevant patterns which are a concise and semantically rich representation of data. Patterns can be clusters, concepts, association rules, outliers, and so on. In this work we analyze some possible ways to abstract or group objects and/or attributes together to get generalized concepts by using taxonomies on attributes and/or objects.

Formal Concept Analysis (FCA) is a formalism for knowledge representation which is based on the formalization of "concepts" and "concept hierarchies" [6]. One recurrent problem in FCA is the number of concepts that can be exponential in the size of the context. To handle this problem many techniques have been proposed [2] to use or produce a taxonomy on attributes or objects to control the size of the context and the corresponding concept lattice.

The rest of this contribution is organized as follows. In Section 2 we introduce the basic notions of FCA. Section 3 presents three different generalization schemes, and discusses different scenarios of generalizing both objects and attributes. In Section 4 we discuss the visualization issue of generalized patterns and provide the real meaning of the three generalization cases. In Section 5 the size of the generalized concept set is compared to the size of the initial (*i.e.*, before generalization) concept set. Finally, existing work about combining FCA with ontology is briefly described in Section 6.

* The second author acknowledges the financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC).

2 Formal Concept Analysis and Data Mining

2.1 Elementary information systems, contexts and concepts

$\mathbb{K}$	a	b	c	d	e	f	g	h
1	×				×		×	
2	×				×	×		×
3	×	×			×	×	×	
4		×			×	×	×	×
5	×		×	×				
6	×	×	×	×				
7		×	×				×	
8		×	×	×			×	

Fig. 1: A formal context

In formal concept analysis, a context is a triple $\mathbb{K} := (G, M, I)$ where G, M and I stand for a set of objects, a set of attributes, and a binary relation between G and M respectively. A formal concept is a pair (A, B) such that B is exactly the set of all properties shared by the objects in A and A is the set of all objects that have all the properties in B. We set $A' := \{m \in M \mid a\mathrm{I}m \text{ for all } a \in A\}$ and $B' := \{g \in G \mid g\mathrm{I}b \text{ for all } b \in B\}$. Then (A, B) is a concept of $\mathbb{K}$ iff $A' = B$ and $B' = A$. The extent of the concept (A, B) is A while its intent is B. We denote by $\mathfrak{B}(\mathbb{K})$, $\mathrm{Int}(\mathbb{K})$ and $\mathrm{Ext}(\mathbb{K})$ the set of concepts, intents and extents of the formal context $\mathbb{K}$, respectively. A subset X is closed if $X'' = X$. Closed subsets of G are exactly extents while closed subsets of M are intents of $\mathbb{K}$. Figure 1 describes items $a, \ldots, h$ that appear in eight transactions (customers) of a *market basket analysis* application. Such a setting defines a binary relation I between the set G of objects/transactions and the set M of properties/items.

The concept hierarchy is formalized with a relation $\leq$ defined on $\mathfrak{B}(\mathbb{K})$ by $A \subseteq C \iff: (A, B) \leq (C, D) :\iff B \supseteq D$. This is an order relation, and is also called a *specialization/generalization* relation on concepts. In fact, a concept (A, B) is called a specialization of a concept (C, D), or (C, D) is a generalization of (A, B) iff $(A, B) \leq (C, D)$ holds. For any list $\mathcal{C}$ of concepts of $\mathbb{K}$, there is a concept $\mathfrak{u}$ of $\mathbb{K}$ which is more general than every concept in $\mathcal{C}$ and is a specialization of every generalization of all concepts in $\mathcal{C}$ ($\mathfrak{u}$ is the *supremum* of $\mathcal{C}$ and is denoted by $\bigvee \mathcal{C}$), and there is a concept $\mathfrak{v}$ of $\mathbb{K}$ which is a specialization of every concept in $\mathcal{C}$ and a generalization of every specialization of all concepts in $\mathcal{C}$ ($\mathfrak{v}$ is the *infimum* of $\mathcal{C}$ and is denoted by $\bigwedge \mathcal{C}$)[3]. Hence, $\mathfrak{B}(\mathbb{K})$ is a *complete lattice* called the concept lattice of the context $\mathbb{K}$.

For $g \in G$ and $m \in M$ we set $g' := \{g\}'$ and $m' := \{m\}'$. The object concepts $(\gamma g := (g'', g'))_{g \in G}$ and the attribute concepts $(\mu m := (m', m''))_{m \in M}$ form the "building blocks" of $\mathfrak{B}(\mathbb{K})$. In fact, every concept of $\mathbb{K}$ is a supremum of some γg's and infimum of some μm's[4]. Thus, the set $\{\gamma g \mid g \in G\}$ is $\bigvee$-dense and the set $\{\mu m \mid m \in M\}$ is $\bigwedge$-dense in $\mathfrak{B}(G, M, \mathrm{I})$.

The size of a concept lattice can be extremely large, even exponential in the size of the context. To handle such large sets of concepts many techniques have been proposed [6], based on context decomposition or lattice pruning/reduction (atlas decomposition, direct or subdirect decomposition, iceberg concept lattices,

[3] For two concepts x_1 and x_2 we set $x_1 \vee x_2 := \bigvee\{x_1, x_2\}$ and $x_1 \wedge x_2 := \bigwedge\{x_1, x_2\}$.

[4] For $(A, B) \in \mathfrak{B}(G, M, \mathrm{I})$ we have $\bigvee\limits_{g \in A} \gamma g = (A, B) = \bigwedge\limits_{m \in B} \mu m$.

nested line diagrams, ...). We believe that using taxonomies on objects and attributes can contribute to the extraction of unexpected and relevant generalized patterns and in most cases to the reduction of the size of discovered patterns.

2.2 Labeled line diagrams of concept lattices

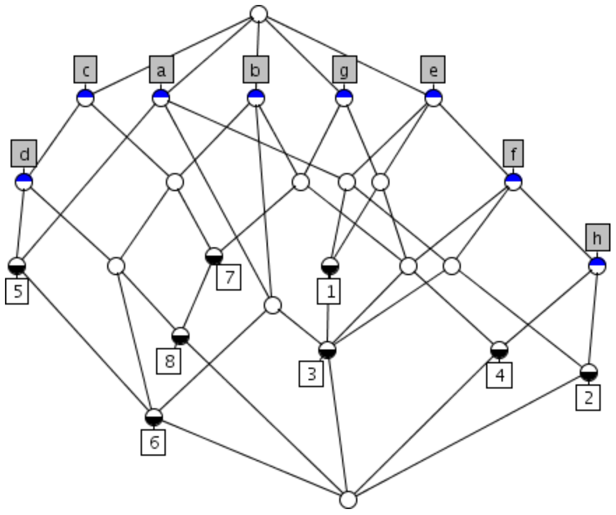


Fig. 2: Concept lattice of the context in Fig 1

One of the strengths of FCA is the ability to pictorially display knowledge, at least for contexts of reasonable size. Finite concept lattices can be represented by reduced labeled Hasse diagrams (see Figure 2). Each node represents a concept. The label g is written below γg and m above μm. The extent of a concept represented by a node a is given by all labels in G from the node a downwards, and the intent by all labels in M from a upwards. For example, the label 5 in the left side of Figure 2 represents the object concept $\gamma 5 = (\{5, 6\}, \{a, c, d\})$. Diagrams are valuable tools for visualizing data. However drawing a good diagram for complex structures is a big challenge. Therefore, we need tools to abstract the output by reducing the size of the input, making the structure nicer, or by exploring the diagram layer by layer. For the last case, FCA offers nested line diagrams as a means to visualize the concepts level-wise [6].

Before we move to generalized patterns, let us see how data are transformed into binary contexts, the suitable format for our data.

2.3 Information Systems

Frequently, data are not directly encoded in a "binary" form, but rather as a many-valued context, *i.e.*, a tuple (G, M, W, I) such that G is the set of objects, M the set of attribute names, W the set of attribute values, $I \subseteq G \times M \times W$ and every $m \in M$ is a partial map from G to W with $(g, m, w) \in I$ iff $m(g) = w$. Many-valued contexts can be transformed into binary contexts, via conceptual scaling. A conceptual scale for an attribute m of (G, M, W, I) is a binary context $\mathbb{S}_m := (G_m, M_m, I_m)$ such that $m(G) \subseteq G_m$. Intuitively, M_m discretizes or groups the attribute values into $m(G)$, and I_m describes how each attribute value $m(g)$ is related to the elements in M_m. For an attribute m of (G, M, W, I) and a conceptual scale $\mathbb{S}_m$ we derive a binary context $\mathbb{K}_m := (G, M_m, I^m)$ with

$gI^m s_m :\iff m(g)I_m s_m$, where $s_m \in M_m$. This means that an object $g \in G$ is in relation with a scaled attribute s_m iff the value of m on g is in relation with s_m in $\mathbb{S}_m$. With a conceptual scale for each attribute we get the derived context $\mathbb{K}^S := (G, N, I^S)$ where $N := \bigcup\{M_m \mid m \in M\}$ and $gI^S s_m \iff m(g)I^m s_m$. In practice, the set of objects remains unchanged; each attribute name m is replaced by the scaled attributes $s_m \in M_m$. The choice of a suitable set of scales depends on the interpretation, and is usually done with the help of a domain expert. A *Conceptual Information System* is a many-valued context together with a set of conceptual scales [9, 12]. The methods presented in Section 3 are actually a form of scaling.

3 Generalized Patterns

In the field of data mining, generalized patterns are pieces of knowledge extracted from data when an ontology is used. In the following we formalize the way generalized patterns are produced. Let $\mathbb{K} := (G, M, I)$ be a context. The attributes of $\mathbb{K}$ can be grouped together to form another set of attributes, namely S, to get a context where the attributes are more general than in $\mathbb{K}$. For the basket market analysis example, items/products can be generalized into product lines and then product categories, and customers may be generalized to groups according to some specific features (e.g., income, education). The context $\mathbb{K}$ is then replaced with a context (G, S, J) as in the scaling process where S can be seen as an index set such that $\{m_s \mid s \in S\}$ covers M. We will usually identify the group m_s with the index s.

3.1 Types of Generalization

There are mainly three ways to express the relation J:

($\exists$) $g\,\mathrm{J}\,s :\iff \exists m \in s,\ g\,\mathrm{I}\,m$. Consider an information table describing companies and their branches in USA. We first set up a context whose objects are companies and whose attributes are the cities where these companies have branches. If there are too many cities, we can decide to group them into states to reduce the number of attributes. Then, the (new) set of attributes is now a set S whose elements are states. It is quite natural to assert that a company g has a branch in a state s if g has a branch in a city m which belongs to the state s.

($\forall$) $g\,\mathrm{J}\,s :\iff \forall m \in s,\ g\,\mathrm{I}\,m$. Consider an information system about Ph.D. students and the components of the comprehensive exam (CE). Assume that components are: the written part, the oral part, and the thesis proposal, and a student succeeds in his exam if he succeeds in the three components of that exam. The objects of the context are Ph.D. students and the attributes are the different exams taken by students. If we group together the different components, for example

$$CE.written, CE.oral, CE.proposal \mapsto CE.exam,$$

then it becomes natural to state that a student g succeeds in his comprehensive exam $CE.exam$ if he succeeds in *all* the exam parts of CE.

(α%) $g\,\mathrm{J}\,s :\iff \frac{|\{m\in s \mid g\,\mathrm{I}\,m\}|}{|s|} \geq \alpha_s$ where α_s is a threshold set by the user for the generalized attribute s. This case generalizes the ($\exists$)-case ($\alpha = \frac{1}{|M|}$) and the ($\forall$)-case ($\alpha = 1$). To illustrate this case, let us consider a context describing different specializations in a given Master degree program. For each program there is a set of mandatory courses and a set of optional ones. Moreover, there is a predefined number of courses that a student should succeed to get a degree in a given specialization. Assume that to get a Master in Computer Science with a specialization in "computational logic", a student must succeed seven courses from a set s_1 of mandatory courses and three courses from a set s_2 of optional ones. Then, we can introduce two generalized attributes s_1 and s_2 so that a student g succeeds in the group s_1 if he succeeds in at least seven courses from s_1, and succeeds in s_2 if he succeeds in at least three courses from s_2. So, $\alpha_{s_1} := \frac{7}{|s_1|}$, $\alpha_{s_2} := \frac{3}{|s_2|}$, and

$$g\,\mathrm{J}\,s_i \iff \frac{|\{m \in s_i \mid g\,\mathrm{I}\,m\}|}{|s_i|} \geq \alpha_{s_i},\ 1 \leq i \leq 2.$$

$\mathbb{K}_{\exists-\forall-\alpha}$	a	b	c	d	e	f	g	h	A	B	C	D	S	T	U	V	E	F	H
1	×				×		×		×		×		×						
2	×				×	×		×	×		×	×				×		×	
3	×	×			×	×	×		×	×	×	×	×				×	×	
4		×			×	×	×	×	×	×		×	×			×		×	×
5	×		×	×						×	×				×		×		
6	×	×	×	×						×	×			×	×		×		
7		×	×				×		×	×				×			×		
8		×	×	×			×		×	×	×			×			×		

Fig. 3: Three generalizations of the context in Fig. 1 (see Subsection 3.1). The $\exists$-generalized attributes are $A := \{e,g\}$, $B := \{b,c\}$, $C := \{a,d\}$ and $D := \{f,h\}$. The $\forall$-generalized attributes are $S := \{e,g\}$, $T := \{b,c\}$, $U := \{a,d\}$ and $V := \{f,h\}$. The α-generalized attributes are $E := \{a,b,c\}$, $F := \{d,e,f\}$ and $H := \{g,h\}$ with $\alpha = 60\%$.

The α-case discussed here generalizes the alpha Galois lattices investigated by Ventos *et al* [14, 10]. In fact, if the set S forms a partition of M and all α_s are equal, then the generalization is an alpha Galois lattice.

Attribute generalization reduces the number of attributes. One may therefore expect a reduction in the number of concepts. Unfortunately, this is not always the case. Therefore, it is interesting to investigate under which condition generalizing patterns leads to a "generalized" lattice of smaller size than the initial one. Moreover, finding the connections between the implications and more generally association rules of the generalized context and the initial one is also an important problem to be considered.

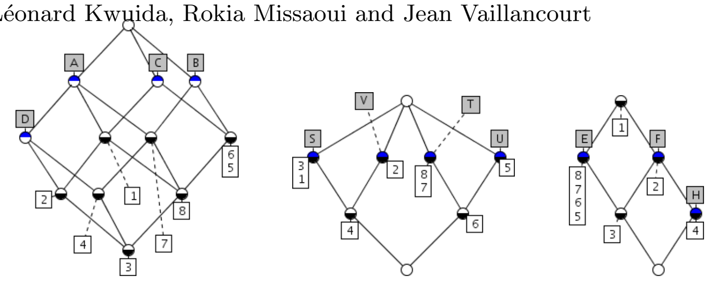


Fig. 4: Lattices of contexts in Fig 3. ∃-generalization (left), ∀-generalization (middle) and α-generalization (right)

If data represent customers (transactions) and items (products), the usage of a taxonomy on attributes leads to new useful patterns that could not be seen before generalizing attributes. For example, the ∃-case (see Figure 4, left) helps the user acquire the following knowledge:

- Customer 3 (at the bottom of the lattice) buys at least one item from each product line
- Whenever a customer buys at least one item from the product line D, then he/she buys at least one item from the product line A.

From the ∀-case in Figure 4 (middle), one may learn for example that Customers 4 and 6 have distinct behaviors in the sense that the former buys all the items of the product lines V and S while the latter purchases all the items of the product lines U and T.

To illustrate the α-case, we put the attributes of M in three groups $E := \{a, b, c\}$, $F := \{d, e, f\}$ and $H := \{g, h\}$ and set $\alpha := 60\%$ for all groups. This α-generalization on the attributes of M is presented in Figure 4 (right). Note that if all groups have two elements, then any α-generalization would be either an ∃-generalization ($\alpha \leq 0.5$) or a ∀-generalization ($\alpha > 0.5$). From the lattice in Figure 4 (right) one can see that any customer buying at least 60% of items in H necessarily purchases at least 60% of items in F. Moreover, the product line E (respectively H) seems to be the most (resp. the less) popular among the four product lines since five out of eight customers (resp. only one customer) bought at least 60% of items in E (resp. H).

Generalization can also be conducted on objects to replace some (or all) of them with generalized objects, or even more, can be done simultaneously on objects and attributes.

3.2 Generalization on Objects and Attributes

Done simultaneously on attributes and on objects, the generalization will give a kind of *hypercontext* (similar to hypergraphs [1]), since the objects are subsets of G and attributes are subsets of M. Let $\mathcal{A}$ be a group of objects and $\mathcal{B}$ be a group of attributes related to a context $\mathbb{K}$. Then, the relation J can be defined using one or a combination of the following cases:

1. $\mathcal{A} \,\mathrm{J}\, \mathcal{B}$ iff $\exists a \in \mathcal{A}$, $\exists b \in \mathcal{B}$ such that $a \,\mathrm{I}\, b$, i.e. some objects from the group $\mathcal{A}$ are in relation with some attributes in the group $\mathcal{B}$;

2. $\mathcal{A}\,\mathrm{J}\,\mathcal{B}$ iff $\forall a \in \mathcal{A}$, $\forall b \in \mathcal{B}$ $a\,\mathrm{I}\,b$, i.e. every object in the group $\mathcal{A}$ is in relation with every attribute in the group $\mathcal{B}$;
3. $\mathcal{A}\,\mathrm{J}\,\mathcal{B}$ iff $\forall a \in \mathcal{A}$, $\exists b \in \mathcal{B}$ such that $a\,\mathrm{I}\,b$, i.e. every object in the group $\mathcal{A}$ has at least one attribute from the group $\mathcal{B}$;
4. $\mathcal{A}\,\mathrm{J}\,\mathcal{B}$ iff $\exists b \in \mathcal{B}$ such that $\forall a \in \mathcal{A}$ $a\,\mathrm{I}\,b$, i.e. there is an attribute in the group $\mathcal{B}$ that belongs to all objects of the group $\mathcal{A}$;
5. $\mathcal{A}\,\mathrm{J}\,\mathcal{B}$ iff $\forall b \in \mathcal{B}$, $\exists a \in \mathcal{A}$ such that $a\,\mathrm{I}\,b$, i.e. every property in the group $\mathcal{B}$ is satisfied by at least one object of the group $\mathcal{A}$;
6. $\mathcal{A}\,\mathrm{J}\,\mathcal{B}$ iff $\exists a \in \mathcal{A}$ such that $\forall b \in \mathcal{B}$ $a\,\mathrm{I}\,b$, there is an object in the group $\mathcal{A}$ that has all the attributes in the group $\mathcal{B}$;
7. $\mathcal{A}\,\mathrm{J}\,\mathcal{B}$ iff $\frac{\left|\{a\in\mathcal{A} \mid \frac{|\{b\in\mathcal{B} \mid a\,\mathrm{I}\,b\}|}{|\mathcal{B}|} \geq \beta_{\mathcal{B}}\}\right|}{|\mathcal{A}|} \geq \alpha_{\mathcal{A}}$, i.e. at least $\alpha_{\mathcal{A}}$ fraction of objects in the group $\mathcal{A}$ have each at least $\beta_{\mathcal{B}}$ fraction of the attributes in the group $\mathcal{B}$;
8. $\mathcal{A}\,\mathrm{J}\,\mathcal{B}$ iff $\frac{\left|\left\{b\in\mathcal{B} \mid \frac{|\{a\in\mathcal{A} \mid a\,\mathrm{I}\,b\}|}{|\mathcal{A}|} \geq \alpha_{\mathcal{A}}\right\}\right|}{|\mathcal{B}|} \geq \beta_{\mathcal{B}}$, i.e. at least $\beta_{\mathcal{B}}\%$ of attributes in the group $\mathcal{B}$ belong altogether to at least $\alpha_{\mathcal{A}}\%$ of objects in the group $\mathcal{A}$;
9. $\mathcal{A}\,\mathrm{J}\,\mathcal{B}$ iff $\frac{|\mathcal{A}\times\mathcal{B}\cap I|}{|\mathcal{A}\times\mathcal{B}|} \geq \alpha$, i.e. the density of the rectangle $\mathcal{A}\times\mathcal{B}$ is at least equal to α.

Remark 1. The cases 7 and 8 generalize Case 1 ($\alpha_{\mathcal{A}} := \frac{1}{|G|}$, $\beta_{\mathcal{B}} := \frac{1}{|M|}$ for all $\mathcal{A}$ and $\mathcal{B}$) and Case 2 ($\alpha_{\mathcal{A}} := 1$, $\beta_{\mathcal{B}} := 1$ for all $\mathcal{A}$ and $\mathcal{B}$). Moreover, Case 7 also generalizes Case 3 ($\alpha_{\mathcal{A}} := 1$, $\beta_{\mathcal{B}} := \frac{1}{|M|}$ for all $\mathcal{A}$ and $\mathcal{B}$) and Case 5 ($\alpha_{\mathcal{A}} := \frac{1}{|G|}$, $\beta_{\mathcal{B}} := 1$ for all $\mathcal{A}$ and $\mathcal{B}$). However, Cases 4 and 6 cannot be captured by Case 7, but are captured by Case 8 ($\alpha_{\mathcal{A}} := 1$, $\beta_{\mathcal{B}} := \frac{1}{|M|}$ for all $\mathcal{A}$ and $\mathcal{B}$ to get Case 4, and $\alpha_{\mathcal{A}} := \frac{1}{|G|}$, $\beta_{\mathcal{B}} := 1$ for all $\mathcal{A}$ and $\mathcal{B}$ to get Case 6).

An example of generalization on both objects and attributes would be one of customers grouped according to some common features and items grouped into product lines. We can also assign to each group all items bought by their members (an $\exists$-generalization) or only their common items (a $\forall$-generalization), or just some of the frequent items among their members (similar to an α-generalization).

4 Visualizing Generalized Patterns on Line diagrams

4.1 Visualization

Let $\mathbb{K}$ be a formal context and (G, S, J) a context obtained from $\mathbb{K}$ via a generalization on attributes. The usual action is to directly construct a line diagram of (G, S, J) which contains concepts with generalized attributes (See Fig 4). However, one may be interested, after getting (G, S, J) and constructing a line diagram for $\mathfrak{B}(G, S, J)$, to refine further on the attributes in M or recover the lattice constructed from $\mathbb{K}$.

When storage space is not a constraint, then the attributes in M and the generalized attributes can be kept altogether. This is done using the apposition of $\mathbb{K} := (G, M, \mathrm{I})$ and (G, S, J) to get $(G, M \cup S, I \cup J)$.

A nested line diagram [6] can be used to display the resulting lattice, with (G, S, J) at the first level and $\mathbb{K}$ at the second one, *i.e.*, we construct a line diagram for $\mathfrak{B}(G, S, J)$ with nodes large enough to contain copies of the line diagram of $\mathfrak{B}(\mathbb{K})$. The generalized patterns can then be visualized by conducting a projection (*i.e.*, a restricted view) on generalized attributes, and keeping track of the effects of the projection, i.e, we display the projection of the concept lattice $\mathfrak{B}(G, M \cup S, I \cup J)$ on S by marking the equivalence classes on $\mathfrak{B}(G, M \cup S, I \cup J)$. Note that two concepts (A, B) and (C, D) are equivalent with respect to the projection on S iff $B \cap S = D \cap S$ (*i.e.*, their intents have the same restriction on S [8]). This is illustrated by Figure 5.

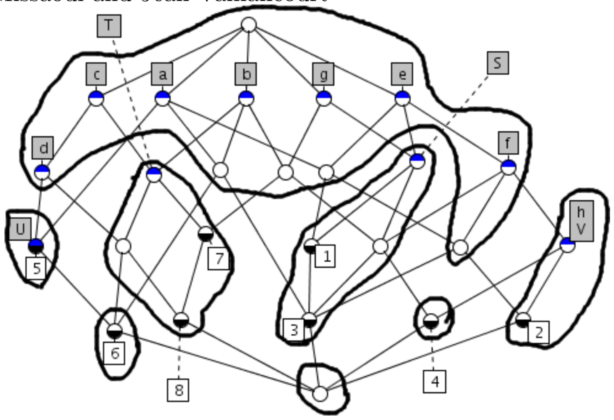


Fig. 5: Projection of the lattice in Figure 2 onto the ∀-generalized attributes.

4.2 Are generalized attributes really generalizations?

For attributes $a, b \in M \cup S$, we should normally assert that a is a generalization of b or b is a specialization of a whenever $\mu a \geq \mu b$. For the $\exists$-case we have, $m_s' = \bigcup\{m' \mid m \in m_s\}$. Thus, $\mu m_s \geq \mu m$ for all $m \in m_s$; i.e. m_s is really a *generalization* of the attributes $m \in m_s$.

For the $\forall$-case we have, $m_s' = \bigcap\{m' \mid m \in m_s\}$. Thus, $\mu m_s \leq \mu m$, $\forall m \in m_s$; i.e. m_s is rather a *specialization* of the attributes $m \in m_s$. For the α-case, $\frac{1}{|M|} < \alpha < 1$, an object $g \in G$ is in relation with an attribute m_s iff $\alpha \leq \frac{|\{m \in m_s | g \, \mathrm{I} \, m\}|}{|m_s|}$. The following situations can happen:

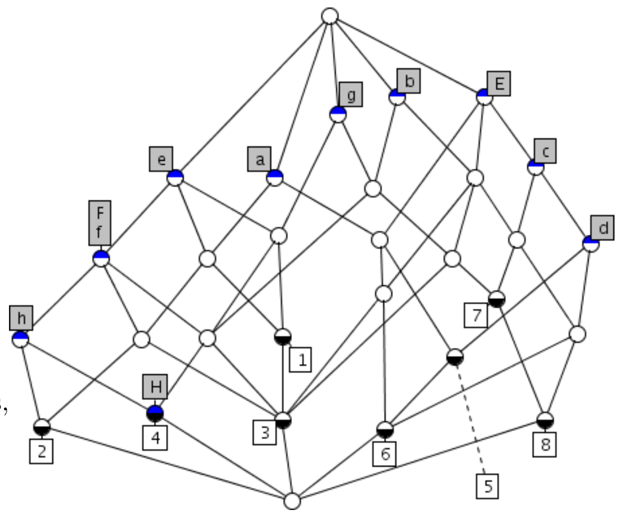


Fig. 6: α-generalization with $\mu E \| \mu b$. $E = \{a, b, c\}$, $F = \{d, e, f\}$, $H = \{g, h\}$, $\alpha = 0.6$.

– There is an α-generalized attribute $m_s \in S$ with at least one attribute $m \in m_s$ such that $g \not\mathrm{I} m$ and $g \, \mathrm{J} \, m_s$; hence $\mu m \not\leq \mu m_s$ in $\mathfrak{B}(G, M \cup S, I \cup J)$; i.e μm_s is not a *generalization* of μm.

– There is an α-generalized attribute $m_s \in S$ with at least one attribute $m \in m_s$ such that $g \,\mathrm{I}\, m$ and $g \not\mathrm{J} m_s$; hence $\mu m_s \not\leq \mu m$ in $\mathfrak{B}(G, M \cup S, I \cup J)$; i.e μm_s is not a *specialization* of μm.

Therefore, there are α-generalized attributes m_s that are neither a *generalization* of the m's nor a *specialization* of the m's. In Figure 6, the element b belongs to the group E, but μE is neither a specialization nor a generalization of μb, since $\mu b \not\leq \mu E$ and $\mu E \not\leq \mu b$. Thus, we should better call the α-case an attribute *approximation*, the $\forall$-case a *specialization* and only the $\exists$-case a *generalization.*

5 Controlling the size of generalized concepts

A generalized concept is a concept whose intent (or extent) contains generalized attributes (or objects). Figure 7 displays an $\exists$-generalization that leads to a larger number of concepts. The two concepts μm_1 and μm_2 will be put together. Although attributes m_1 and m_2 are replaced with m_{12}, the nodes γg_2 and γg_3 will remain since they will be obtained as $\mu m_{12} \wedge \mu m_4$ and $\mu m_{12} \wedge \mu m_3$ respectively. Then we get the configuration on Figure 7 (right) which has one concept more than the initial concept lattice shown in the left of the same figure.

In the following, we analyze the impact of $\exists$ and $\forall$ attribute generalizations on the size of the resulting set of generalized concepts.

5.1 An $\exists$-generalization on attributes

Let (G, M, I) be a context and (G, S, J) a context obtained from an $\exists$-generalization on attributes, i.e the elements of S are groups of attributes from M. We set $S = \{m_s \mid s \in S\}$, with $m_s \subseteq M$. Then, an object $g \in G$ is in relation with a generalized attribute m_s if there is an attribute m in m_s such that $g \,\mathrm{I}\, m$. To compare the size of the corresponding concept lattices, we can define some mappings. We assume that $(m_s)_{s\in S}$ forms a partition of M. Then for each $m \in M$ there is a unique generalized attribute m_s such that $m \in m_s$, and $g \,\mathrm{I}\, m$ implies $g \,\mathrm{J}\, m_s$, for every $g \in G$. To distinguish between derivations in (G, M, I) and in (G, S, J), we will replace $'$ by the name of the corresponding relation. For example $g^I = \{m \in M \mid g \,\mathrm{I}\, m\}$ and $g^J = \{s \in S \mid g \,\mathrm{J}\, s\}$. Two canonical maps α and β are defined as follows:

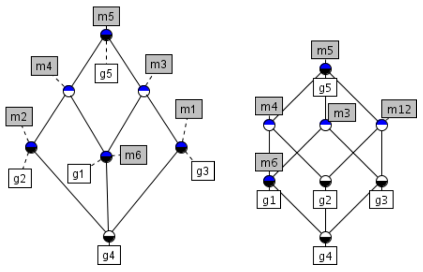


Fig. 7: An $\exists$-generalization (right) increasing the size of the initial lattice (left). $m_{12} = \{m_1, m_2\}$.

$$\begin{aligned} \alpha\colon G &\to \mathfrak{B}(G, S, J) \\ g &\mapsto \bar{\gamma} g := (g^{\mathrm{J}\,\mathrm{J}}, g^{\mathrm{J}}) \end{aligned} \quad \text{and} \quad \begin{aligned} \beta\colon M &\to \mathfrak{B}(G, S, J) \\ m &\mapsto \bar{\mu} m_s := (s^{\mathrm{J}}, s^{\mathrm{J}\,\mathrm{J}}), \text{ where } m \in m_s \end{aligned}$$

The maps α and β induce two order preserving maps φ and ψ defined by

$$\begin{aligned}\varphi : \mathfrak{B}(G,M,I) &\to \mathfrak{B}(G,S,J)\\ (A,B) &\mapsto \bigvee\{\alpha g \mid g \in A\}\end{aligned} \quad \text{and} \quad \begin{aligned}\psi : \mathfrak{B}(G,M,I) &\to \mathfrak{B}(G,S,J)\\ (A,B) &\mapsto \bigwedge\{\beta m \mid m \in B\}\end{aligned}$$

If φ or ψ is surjective, then the generalized context is of smaller cardinality. As we have seen on Figure 7 these maps can be both not surjective. Obviously $\varphi(A,B) \leq \psi(A,B)$ since $g\,\mathrm{I}\,m$ implies $g\,\mathrm{J}\,m_s$ and $\bar{\gamma} g \leq \mu \bar{m}_s$. When do we have the equality? Does the equality imply surjectivity?

Here are some special cases where the number of concepts does not increase after a generalization.

Case 1 Every m_s has a greatest element $\top_s$. Then the context (G,S,J) is a projection of (G,M,I) on the set $M_S := \{\top_s \mid s \in S\}$ of greatest elements of m_s. Thus $\mathfrak{B}(G,S,J) \cong \mathfrak{B}(G,M_S,I \cap (G \times M_S))$ and is a sub-order of $\mathfrak{B}(G,M,I)$. Hence $|\mathfrak{B}(G,S,J)| = |\mathfrak{B}(G,M_S,I \cap G \times M_S)| \leq |\mathfrak{B}(G,M,I)|$.

Case 2 $\bigcup\{m^{\mathrm{I}} \mid m \in m_s\}$ is an extent, for any $m_s \in S$. Then any grouping does not produce a new concept. Hence the number of concepts cannot increase.

The following result (Theorem 1) gives an important class of lattices for which the $\exists$-generalization does not increase the size of the lattice. A context is object reduced if no row can be obtained as the intersection of some other rows.

Theorem 1. *The $\exists$-generalizations on distributive concept lattices whose contexts are object reduced decrease the size of the concept lattice.*

Proof. Let (G,M,I) be an object reduced context such that $\mathfrak{B}(G,M,I)$ is a distributive lattice. Let (G,S,J) be a context obtained by an $\exists$-generalization on the attributes in M. Let m_s be a generalized attribute, i.e. a group of attributes of M. It is enough to prove that m_s^J is an extent of (G,M,I). By definition,

$$m_s^J = \bigcup\{m^I \mid m \in m_s\} \subseteq \left(\bigcup\{m^I \mid m \in m_s\}\right)^{II} = \text{ext}\left(\bigvee\{\mu m \mid m \in m_s\}\right)$$

For any $g \in \text{ext}(\bigvee\{\mu m \mid m \in m_s\})$ we have $\gamma g \leq \bigvee\{\mu m \mid m \in m_s\}$ and

$$\gamma g = \gamma m \wedge \bigvee\{\mu m \mid m \in m_s\} = \bigvee\{\gamma g \wedge \mu m \mid m \in m_s\} = \gamma g \wedge \mu m \text{ for some } m \in m_s.$$

Therefore $\gamma g \leq \mu m$, and $g \in m^I$. This proves that $\text{ext}(\bigvee\{\mu m \mid m \in m_s\}) \subseteq m_s^J$, and $m_s^J = \text{ext}\,(\bigvee\{\mu m \mid m \in m_s\})$.

Remark 2. The above discussed cases are not the only ones where the size does not increase. Therefore, it would be interesting to describe the classes of lattices on which $\exists$-generalizations do not increase the size.

5.2 A $\forall$-generalization on attributes

Let (G,S,J) be a context obtained from (G,M,I) by a $\forall$-generalization. In the context $(G, M \cup S, I \cup J)$, each attribute concept μm_s is reducible. This means that $m_s^{\mathrm{J}} = \bigcap\{m^{\mathrm{J}} \mid m \in m_s\} = \bigcap\{m^{\mathrm{I}} \mid m \in m_s\}$, and is an extent of (G,M,I). Therefore, $|\mathfrak{B}(G,S,J)| \leq |\mathfrak{B}(G,M \cup S, I \cup J)| = |\mathfrak{B}(G,M,I)|$.

Theorem 2. *The $\forall$-generalizations on attributes reduce the size of the concept lattice.*

6 Related work

There are a set of studies [2–5, 7, 13, 15] about the possible collaborations between formal concept analysis and ontology engineering (e.g., ontology merging and mapping) to let the two formalisms benefit from each other strengths. For example, starting from the observation that both domain ontologies and FCA aim at modeling concepts, [2] show how FCA can be exploited to support ontology engineering (e.g., ontology construction and exploration), and conversely how ontologies can be fruitfully used in FCA applications (e.g., extracting new knowledge). In [13], the authors propose a bottom-up approach called $FCA{-}MERGE$ for merging ontologies using a set of documents as input. The method relies on techniques from natural language processing and FCA to produce a lattice of concepts. Starting from a set of domain specific texts, [7] proposes a semi-automatic method for ontology extraction and design based on FCA and Horn clause model. [5] studies the role of FCA in reusing independently developed domain ontologies. To that end, an ontology-based method for evaluating similarity between FCA concepts is defined to perform some Semantic Web activities such as ontology merging and ontology mapping. In [15] an approach towards the construction of a domain ontology using FCA is proposed. The resulting ontology is represented as a concept lattice and expressed via the Semantic Web Rule Language to facilitate ontology sharing and reasoning.

In [4], a method for ontology mapping, called FCA-Mapping, is defined based on FCA and allows the identification of equal and subclass mapping relations. In [3], FCA is also used to propose an ontology mediation method for ontology merging. The resulting ontology includes new concepts not originally found in the input ontologies but excludes some redundant or irrelevant concepts.

In association rule mining, there are many efforts to integrate knowledge in the process of rule extraction to produce generalized patterns [11].

7 Conclusion

In this paper we have studied the problem of using a taxonomy on objects and/or attributes in the framework of formal concept analysis under three main cases of generalization ($\exists$, $\forall$, and α) and have shown that (i) the set of *generalized* concepts is in some cases smaller than the set of patterns extracted from the *original* set of attributes (before generalization), and (ii) the generalized concept lattice not only embeds new patterns on generalized attributes but also reveals particular features of objects and may unveil a new taxonomy on objects. A careful analysis of the three cases of attribute generalization led to the following conclusion: the α-case is an attribute *approximation*, the $\forall$-case is an attribute *specialization* while only the ∃-case is actually an attribute *generalization*. Different scenarios of a simultaneous generalization on objects and attributes are also discussed based on the three cases of generalization.

Since we focused our analysis on the integration of taxonomies in FCA to produce generalized concepts, our further research concerns the theoretical study of the mapping between a rule set on original attributes and a rule set of generalized attributes as well as the exploitation of other components of an ontology such as general links (other than *is-a* hierarchies) between concepts/entities.

References

1. Claude Berge. *Graphs and hypergraphs*, volume 6 of *North - Holland Mathematical Library*. North-Holland, Elsevier, Amsterdam, repr. of the 2., rev. ed. edition, 1979.
2. Philipp Cimiano, Andreas Hotho, Gerd Stumme, and Julien Tane. Conceptual knowledge processing with formal concept analysis and ontologies. In *ICFCA*, pages 189–207, 2004.
3. Olivier Curé and Robert Jeansoulin. An fca-based solution for ontology mediation. In *ONISW '08: Proceeding of the 2nd int. workshop on Ontologies and nformation systems for the semantic web*, pages 39–46, New York, NY, USA, 2008. ACM.
4. Liya Fan and Tianyuan Xiao. An automatic method for ontology mapping. In *Knowledge-Based Intelligent Information and Engineering Systems*, pages 661–669, 2007.
5. Anna Formica. Ontology-based concept similarity in formal concept analysis. *Inf. Sci.*, 176(18):2624–2641, 2006.
6. Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag New York, Inc., 1999. Translator-C. Franzke.
7. Hele-Mai Haav. A semi-automatic method to ontology design by using fca. In *CLA*, 2004.
8. Léonard Kwuida, Rokia Missaoui, Beligh Ben Amor, Lahcen Boumedjout, and Jean Vaillancourt. Restrictions on concept lattices for pattern management. In *Concept Lattices and Applications (CLA)*, pages 235–246, 2010.
9. Patrick Scheich, Martin Skorsky, Frank Vogt, Cornelia Wachter, and Rudolf Wille. Conceptual data systems. In O. Opitz, B. Lausen, and R. Klar, editors, *Information and Classification*, pages 72–84. Springer, Berlin-Heidelberg, 1993.
10. Henry Soldano and Véronique Ventos. Abstract concept lattices. In *ICFCA*, pages 235–250, 2011.
11. Ramakrishnan Srikant and Rakesh Agrawal. Mining generalized association rules. In *VLDB*, pages 407–419, 1995.
12. Gerd Stumme. Conceptual On-Line Analytical Processing. In K. Tanaka, S. Ghandeharizadeh, and Y. Kambayashi, editors, *Information Organization and Databases*, chapter 14, pages 191–203. Kluwer, 2000.
13. Gerd Stumme and Alexander Maedche. FCA-MERGE: Bottom-up merging of ontologies. In *IJCAI*, pages 225–234, 2001.
14. Véronique Ventos and Henry Soldano. Alpha galois lattices: An overview. In *ICFCA*, pages 299–314, 2005.
15. Jian Wang and Keqing He. Towards representing fca-based ontologies in semantic web rule language. In *CIT '06: Proceedings of the Sixth IEEE International Conference on Computer and Information Technology*, page 41, Washington, DC, USA, 2006. IEEE Computer Society.

Formal Concept Analysis Applied to Transcriptomic Data

Mehwish Alam[2,3], Adrien Coulet[2,3], Amedeo Napoli[1,2], and Malika Smaïl-Tabbone[2,3]

[1] CNRS, LORIA, UMR 7503, Vandoeuvre-lès-Nancy, F-54506, France
[2] Inria, Villers-lès-Nancy, F-54600, France
[3] Université de Lorraine, LORIA, UMR 7503, Vandoeuvre-lès-Nancy, F-54506, France
{mehwish.alam,adrien.coulet,amedeo.napoli,malika.smail@inria.fr}

Abstract. Identifying functions shared by genes responsible for cancer is a challenging task. This paper describes the preparation work for applying Formal Concept Analysis (FCA) to complex biological data. We present here a preliminary experiment using these data on a core context with the addition of domain knowledge. The resulting concept lattices are explored and some interesting concepts are discussed. Our study shows how FCA can help the domain experts in the exploration of complex data.

Keywords: Formal Concept Analysis, Knowledge Discovery, Transcriptomic Data.

1 Introduction

Over past few years, large volumes of transcriptomic data were produced but their analysis remains a challenging task because of the complexity of the biological background. Some earlier studies aimed at retrieving sets of genes sharing the same transcriptional behavior with the help of Formal Concept Analysis [1, 2]. Further studies analyze gene expression data by using gene annotations to determine whether a set of differentially expressed genes is enriched with biological attributes [3, 4]. Several efforts have been made for integrating heterogeneous data [5]. For example, at the Broad Institute, biological data were recently gathered from multiple resources to get thousands of predefined genesets stored in the Molecular Signature DataBase (MSigDB) [6]. A predefined geneset is a set of genes known to have a specific property such as their position on the genome, their involvement in a molecular pathway etc.

This paper focuses on the preparation of biological data to data mining guided by domain knowledge. The objective is to apply knowledge discovery techniques for analyzing a list of differentially expressed genes and identifying functions or pathways shared by these genes assumed to be responsible for cancer. Section 2 explains the proposed approach for FCA-based analysis of biological data. Section 3 focuses on the conducted experiment. Section 4 discusses the results. Section 5 concludes the paper.

2 The Proposed Framework

We rely on the standard definition of FCA fully described in [7] and adapt it according to the current problem. Let G be the set of genes $\{g_1, g_2, g_3, ..., g_n\}$, and M be a set of attributes of MSigDB for describing genes. M will be considered as a partition of three points of view, $M = M_1 \cup M_2 \cup M_3$, with $M_i \cap M_j = \emptyset$ whenever $i \neq j$.

The first set of attributes M_1 refers to four types of attributes, "Location", "Pathway", "Transcription Factors" and "GO Terms" (see Table 1). For our convenience we have named MSigDB categories as types of attributes and used only C_1, C_2, C_3 and C_5. The category C_4 was not used as it keeps information on sets of genes related to a certain kind of cancer, which is not useful for the current problem. Thus we have a first context $K_1 = (G, M_1, I_1)$ where I_1 denotes the relation stating that gene g_i has an attribute m_j in M_1.

Types of Attributes	**Description**	**Data Provenance**
C1: Positional Gene Sets	Location of the gene on the chromosome.	Broad Institute
C2: Curated Gene Sets	Pathway	KEGG, REACTOME, BIOCARTA
C3: Motif Gene Sets	Transcription Factors	Broad Institute
C4: Computational Gene Sets	Cancer Modules	Broad Institute
C5: Gene Ontology (GO) Gene Sets	Biological Process, Cellular Components, Molecular Functions	AmiGO

Table 1. Types of attributes from MSigDB

The second set of attributes M_2 is related to the so-called "categories" where a category makes reference to a set of attributes with the "Pathway" type. For example, "Cell Growth and Death" is an example of category (see Figure 1). The categories in M_2 determine a second context, $K_2 = (G, M_2, I_2)$ where M_2 is the set of categories and I_2 denotes the relation between a gene and a category. It can be noticed that the categories are only related to the "Pathway" type and that they can be considered as domain knowledge.

Moreover, the third set of attributes, namely M_3, refers to the so-called "upper categories", which are defined as groupings of categories. Actually, we have for the type "Pathway" a hierarchy of categories with two levels, categories and upper categories (see Figure 1). The upper categories in M_3 define a third context $K_3 = (G, M_3, I_3)$ where M_3 is the set of upper categories and I_3 denotes the relation between gene g_i and an upper level category m_j. Upper categories are also related to the "Pathway" type and as categories, they can be considered as domain knowledge too.

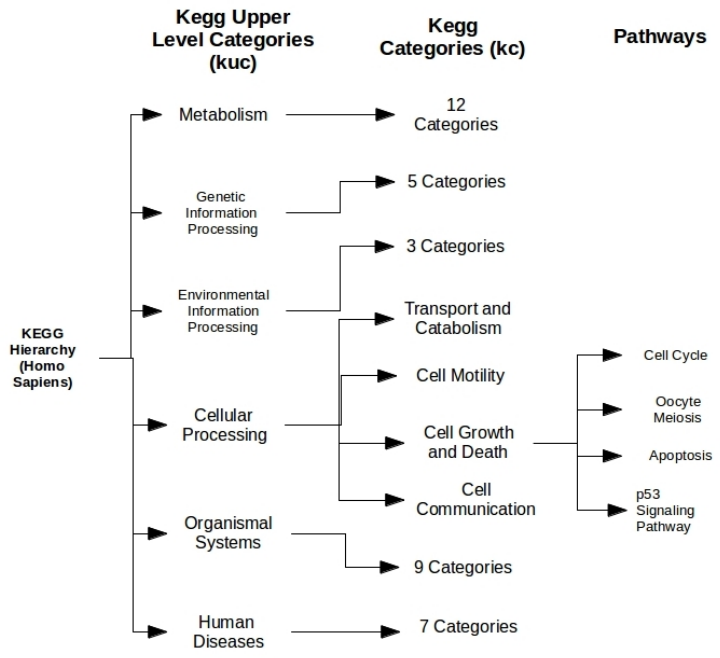


Fig. 1. Categories and upper categories in KEGG.

Now we consider the apposition of the three contexts K_1, K_2 and K_3, which yields the final context $K = (G, K_1 \cup K_2 \cup K_3, I_1 \cup I_2 \cup I_3)$. For example the context in Table 2 shows five genes described by attributes of M_1, M_2 and M_3.

3 Using FCA for Analyzing Genes

The framework described above was applied on three published sets of genes corresponding to Cancer Modules defined in [8]. Our test data are composed of three lists of genes corresponding to the so-called "Cancer Module 1" (Ovary Genes), "Cancer Module 2" (Dorsal Root Ganglia Genes), and "Cancer Module 5" (Lung Genes). For example, "PSPHL" is one gene with "Pathway" attribute as "PPAR Signaling" which belongs to category "kc:Endocrine System" and upper category "kuc:Organismal System". Considering the three lists of genes given by "Cancer Module 1", "Cancer Module 2" and "Cancer Module 5", we built three different contexts having the same form as the context in Table 2). Then we obtained three associated concept lattices with the help of the Coron Plate-form (`http://coron.loria.fr`). The concept lattice for Table 2 is given in Figure 2. The global characteristics of the three concept lattices are given in Table 3.

The exploration of a given concept lattice is carried out following the "Iceberg metaphor", i.e., the lattice is explored level by level according to the support of

Genes	ATP Binding (GO Term)	Serotonin Receptors (Pathway)	PPAR Signaling (Pathway)	V$POU3F2_02 (Transcription Factor)	Cellular Component Assembly(GO Term)	chr5q12 (Location)	kc:Endocrine System	kuc:Organismal Systems
BTB03	×				×	×		
PSPHL			×	×			×	×
CCT6A		×				×		
QNGPT1	×	×			×	×		
MYC	×			×				

Table 2. A toy example of formal context including domain knowledge.

each concept, where the the support of a concept is the cardinality of the extent. In addition, we also used stability for extracting interesting frequent and stable concepts [9].

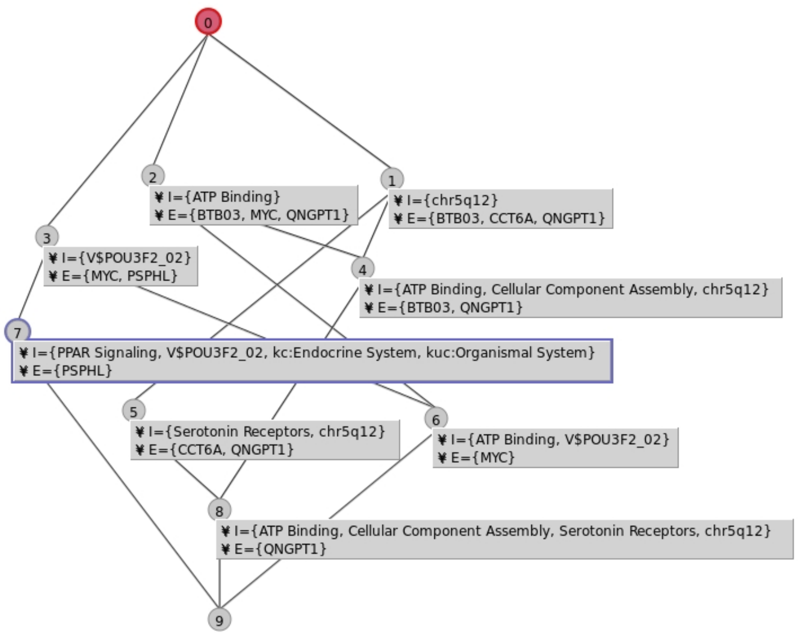

0
2
¥ I={ATP Binding}
¥ E={BTB03, MYC, QNGPT1}
1
¥ I={chr5q12}
¥ E={BTB03, CCT6A, QNGPT1}
3
¥ I={V$POU3F2_02}
¥ E={MYC, PSPHL}
4
¥ I={ATP Binding, Cellular Component Assembly, chr5q12}
¥ E={BTB03, QNGPT1}
7
¥ I={PPAR Signaling, V$POU3F2_02, kc:Endocrine System, kuc:Organismal System}
¥ E={PSPHL}
5
¥ I={Serotonin Receptors, chr5q12}
¥ E={CCT6A, QNGPT1}
6
¥ I={ATP Binding, V$POU3F2_02}
¥ E={MYC}
8
¥ I={ATP Binding, Cellular Component Assembly, Serotonin Receptors, chr5q12}
¥ E={QNGPT1}
9

Fig. 2. The concept lattice corresponding to table 2.

Data Sets	No. of Genes	No. of Attributes	No. of Concepts	Levels
Module 1	361	3496	9,588	12
Module 2	378	3496	6,508	11
Module 5	419	3496	5,004	12

Table 3. Concept lattice statistics for the cancer modules with domain knowledge.

4 Results

In this study, biologists are interested in links between the input genes in terms of pathways in which they participate, relationships between genes and their positions etc. We obtained concepts with shared transcription factors, pathways, locations of genes and GO terms. After the selection of concepts with a high support (≥ 10), we observed that there were some concepts with pathways either related to cell proliferation or apoptosis (expert interpretation). The addition of domain knowledge gives an opportunity to obtain the pathway categories shared by larger sets of genes (as categories and upper categories are there for maximizing the grouping of objects, see below).

Table 4 shows the top-ranked concepts found in each module. For example, in Table 4, we have the concept C_{4938}:*(KEGG Cytokine Cytokine Receptor Interaction, kc:Signaling Molecules and Interaction, kuc:Environmental Information Processing)* and the concept C_{4995}:*(kc:Signaling Molecules and Interaction, kuc:Environmental Information Processing)*. These two concepts are such as $C_{4938} \leq C_{4995}$, meaning that C_{4995} has greater support than C_{4938}. Moreover, we observed that the introduction of categories and upper categories in the global context allows us to consider concepts that otherwise would not be frequent. Actually, the role of categories and upper level categories is to facilitate the observation of sets of related genes.

This is a general way of obtaining larger sets of objects to interpret. When available, one can introduce a hierarchy of attributes –this is domain knowledge– and then insert the levels of each attribute in this hierarchy as a new attribute in the context. As a result, some classes of objects, that could not emerge before, will appear based on these hierarchical indications. Given the test data sets, the preliminary results obtained here constitute an interesting and positive control, and confirm that FCA-based analysis offers an efficient and practical procedure to explore complex and large sets of genes.

5 Conclusion

The preliminary study presented here shows how FCA can be applied to complex biological data and can give flexibility in using various types of attributes for analyzing a list of genes. In addition, domain knowledge can be introduced and guide the analysis.

Dataset	**Concept ID**	**Intents**	**Absolute Support**	**Stability**
Module 1	**9585**	GGGAGGRR _V$MAZ_Q6	51	0.99
	9571	GO Membrane Part	27	0.99
	9566	kc:Immune System, kuc:Organismal Systems	25	0.99
	9402	chr19q13	10	0.99
	9078	KEGG MAPK Signaling Pathway, kc:Signal Transduction, kuc:Environmental Information Processing	12	0.87
Module 2	**6502**	GGGAGGRR _V$MAZ_Q6	44	0.99
	6501	AACTTT _UNKNOWN	38	0.99
	6496	kc:Immune System, kuc:Organismal Systems	15	0.99
	6388	chr6p21	10	0.97
	6335	KEGG MAPK Signaling Pathway, kc:Signal Transduction, kuc:Environmental Information Processing	11	0.89
Module 5	**5002**	kuc:Cellular Processes	48	0.99
	5000	GGGAGGRR _V$MAZ_Q6	44	0.99
	4995	kc:Signaling Molecules and Interaction, kuc:Environmental Information Processing	26	0.99
	4933	chr19q13	11	0.99
	4985	kc:Immune System, kuc:Organismal Systems	11	0.99
	4938	KEGG Cytokine Cytokine Receptor Interaction, kc:Signaling Molecules and Interaction, kuc:Environmental Information Processing	11	0.87

Table 4. Top ranked concepts for each cancer module

As for future work, we plan to take into account relationships between genes and between terms (Gene Ontology relationships) and use the framework of relational concept analysis.

References

1. Kaytoue-Uberall, M., Duplessis, S., Kuznetsov, S.O., Napoli, A.: Two FCA-Based Methods for Mining Gene Expression Data. In Ferré, S., Rudolph, S., eds.: ICFCA. Volume 5548 of Lecture Notes in Computer Science., Springer (2009) 251–266
2. Rioult, F., Boulicaut, J.F., Crémilleux, B., Besson, J.: Using Transposition for Pattern Discovery from Microarray Data. In: DMKD. (2003) 73–79
3. Berriz, G.F., King, O.D., Bryant, B., Sander, C., Roth, F.P.: Characterizing gene sets with FuncAssociate. Bioinfo. **19**(18) (2003) 2502–2504
4. Doniger, S., Salomonis, N., Dahlquist, K., Vranizan, K., Lawlor, S., Conklin, B.: MAPPFinder: using Gene Ontology and GenMAPP to Create a Global Gene-expression Profile from Microarray Data. Genome Biology **4**(1) (2003) R7
5. Galperin, M.Y., Fernández-Suarez, X.M.: The 2012 Nucleic Acids Research Database Issue and the online Molecular Biology Database Collection. Nucleic Acids Research **40** (2012) 1–8
6. Liberzon, A.: Molecular Signatures Database (MSigDB) 3.0. Bioinfo. **27**(12) (2011) 1739–1740
7. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin/Heidelberg (1999)
8. Segal, E., Friedman, N., Koller, D., Regev, A.: A Module Map Showing Conditional Activity of Expression Modules in Cancer. Nat.Genet. **36** (2004) 1090–8
9. Kuznetsov, S.O.: On stability of a Formal Concept. Ann. Math. Artif. Intell. **49**(1-4) (2007) 101–115

WebGeneKFCA: an On-line Conceptual Analysis Tool for Genomic Expression Data

José María Fernández-Calabozo, Carmen Peláez-Moreno, and Francisco J. Valverde-Albacete [⋆]

Dpto. de Teoría de la Señal y de las Comunicaciones.
Universidad Carlos III de Madrid
Avda. de la Universidad, 30. Leganés 28911. Spain
`jmgc,carmen,fva@tsc.uc3m.es`

Abstract. In this paper we introduce a Web-based tool for the analysis of Genomic Expression (GE) data based in $\mathcal{K}$-Formal Concept Analysis (KFCA). First we present the task of analysing GE data and then we describe the tool implementing KFCA. As a second contribution, we present a mechanism to visualise a sequence of concept lattices by fixing the intents against the concept lattice of the contranominal scale of attributes $\mathfrak{B}(M, M, \neq)$. Derived from this we also propose a mechanism to explore the scope of objects in such sequences.

1 Introduction

The *transcriptome* of a species is the set of *gene expression products*, be they proteins or messenger RNA (mRNA) chains. DNA micro-arrays are a mechanism to take measures of such data in the form of an *expression profile*, a record of the concentration of different mRNA associated to a subset of the species genome with respect to a *condition*, a particular state or sequence of states undergone by the cells under study.

In this context, the concentration of the transcribed product (usually mRNA) is the *(gene) expression value*, and the expression values of a set of genes under the same condition, an *expression profile.* Therefore, given a *genome* —a set of *genes*—$G = \{g_i\}_{i=1}^{n}$ the *Gene Expression (GE) data* taken to analyse their functional influence consists of the expression value of every gene R_{ij}—an expression profile—under one condition m_j in a non-explicitly given set of conditions $M = \{m_j\}_{j=1}^{p}$. Under these premises, *co-regulation* refers to the increment (*up-regulation*) or decrement (*down-regulation*) of the expression value in a set of genes brought about by the change in expression value of other genes.

GE data exploration using Formal Concept Analysis includes the seminal work for of [1]. Later contributions essentially adhere to this paradigm, including our own [2] where we employ $\mathcal{K}$-Formal Concept Analysis (KFCA) [3].

In this paper we introduce WebGeneKFCA, an application supporting Exploratory Analysis of GE data using KFCA that attempts to embody the iterative process of exploration based on *inductive databases* suggested by Pensa et

[⋆] Corresponding Author.

al. [1]. Beyond the proof-of-concept nature of the work described in [2] WebGeneKFCA insists on providing tools for the practitioner to contextualise with domain knowledge as embodied in Gene Ontologies (GOs): a point-and-click interface seamlessly integrated with lattice visualisation enables the exploration of many different contextualised hypotheses.

Data procurement and normalisation is described in Sec. 2.1. Our extension of the state of the art in concept lattice (CL) visualisation that provides a representation for *sequences* of these is described in Sec. 2.2.

Relying on this property, we present yet another new visualisation feature whereby the related concepts in different CL of the sequence can be aggregated and their scope in terms of the value of a single continuous parameter explored (Sec. 2.3). The paper closes with a summary of contributions and further work.

2 Exploratory Analysis with WebGeneKFCA

In this Section we describe the Exploratory Analysis of Gene Expression Data using the inductive databases paradigm as embodied in WebGeneKFCA [1].

2.1 Data procurement and normalization

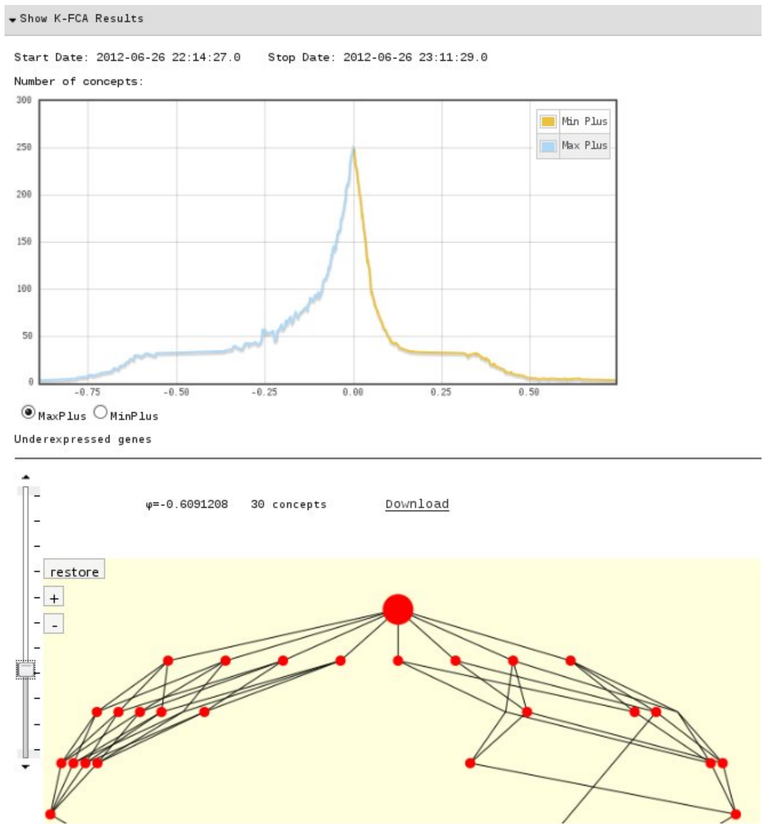
Before any new data can be analysed it must be uploaded to the platform in form of Affymetrix v4 CEL files [4]. Each experiment to analyse will consist of several tests each corresponding to a CEL file. Then, the `apt-summary-tool` is executed which is an open source tool provided by Affymetrix that creates matrix R_{ij} where each column represents a condition profile obtained from a CEL file and each row is a gene profile.

Prior to data analysis, the user must choose how to normalise the data to make it suitable for $\mathcal{K}$-Formal Concept Analysis. Currently we support four different types of normalisation schemes that make use of a special kind of profile called *control*: no normalisation, by the arithmetic mean, the geometric mean or the maximum value of the control profile.

2.2 Lattice Exploration and visualisation

As explained in [2], lattice exploration consists in sweeping all possible values of the data matrix R' in the $\overline{\mathbb{R}}_{\max,+}$ and $\overline{\mathbb{R}}_{\min,+}$ domains. The result of these two different exploring strategies can be shown together as in Fig. 1. The Structural Context that gives rise to the CL shown can also be obtained by clicking on the "Download" link just on the top of the graph in a standard CSV format.

[1] The server was built using Java 1.6 and the Spring framework (v. 3.1), and runs on Tomcat 6. The web view makes extensive use of *javascript* and *html5* and has been optimised for Chrome web browsers. To store the data the application currently uses *mysql* but it can be easily ported to any other SQL database. At least 1.5GB of free RAM is required. The tool is currently only available for in-house usage.


Show K-FCA Results

Start Date: 2012-06-26 22:14:27.0 Stop Date: 2012-06-26 23:11:29.0

Number of concepts:

[✓ MaxPlus] MinPlus

Underexpressed genes

φ=-0.6091208 30 concepts Download

Fig. 1: (Colour online) First data exploration screen. The top pane shows the number of concepts vs. φ (light blue, to the left of 0.0) and ϕ (drab green, right of 0.0) for the context being explored. The bottom pane shows the φ slider and (part of) the CL thus selected. In-between, the toggle for $\overline{\mathbb{R}}_{\max,+}$ or $\overline{\mathbb{R}}_{\min,+}$.

Several algorithms exist for the visualisation of CL, each with its advantages and disadvantages (see [5] for a review). However, since the use of $\mathcal{K}$-Formal Concept Analysis requires the visualisation of a *sequence* of CL, we propose a scheme having the distinctive feature that the Formal Concepts with the same intent belonging to different CL are always plotted in the same position. This means that the user can change the value of ϕ (φ) and will easily see how the extent of each concept evolves, increasing or decreasing until disappearing.

To ensure this property, the CL corresponding to a particular ϕ (φ) is drawn *over* the silhouette of the CL of a (virtual) contranominal scale involving all possible attributes, $\mathbb{N}^c_M = \underline{\mathfrak{B}}(M, M, \neq)$.

The rationale for this overlay is as follows: in the boolean lattice $\underline{\mathfrak{B}}(M, M, \neq)$, the top is a concept with no attributes. The next level of concepts from the top are those which only have one attribute, and so on. Call $|M| = p$, the number of conditions. Clearly, the total number of levels is the number of conditions plus one, and the number of concepts in each level l is $\binom{p}{l}$, whence $\underline{\mathfrak{B}}(M, M, \neq)$ has the maximum possible number of Formal Concepts for a given set of attributes M, a well-known result (see [6], p.48).

The important fact is that *any possible intent in any lattice with p attributes appears in this virtual order diagram*. Thus the locations of these Formal Concepts of the most complex CL related to M can serve as locations for those Formal Concepts of *any other CL with the same attribute set M*. Figure 2.(a) shows an example of such an overlay of a CL with relatively few concepts, while another example is shown in Fig. 2.(b) of a CL whose set of intents approaches that of the boolean lattice (despite conspicuous absences, e.g. in the atom set).

In this visualisation scheme, the size of each concept node is directly proportional to its extent size (but see 2.3). Making the mouse hover over each node, a dialog box showing the attributes of that concept and its number of objects appears. On clicking on the node, a floating window appears showing the full extent. Besides, since these objects are genes, each of them can be selected causing its associated information, obtained from the corresponding NetAffx Annotation file [7], to be displayed in a pane on the right side. This information pane also links with other external web pages that offer more information.

2.3 The Exploration of an Object Scope

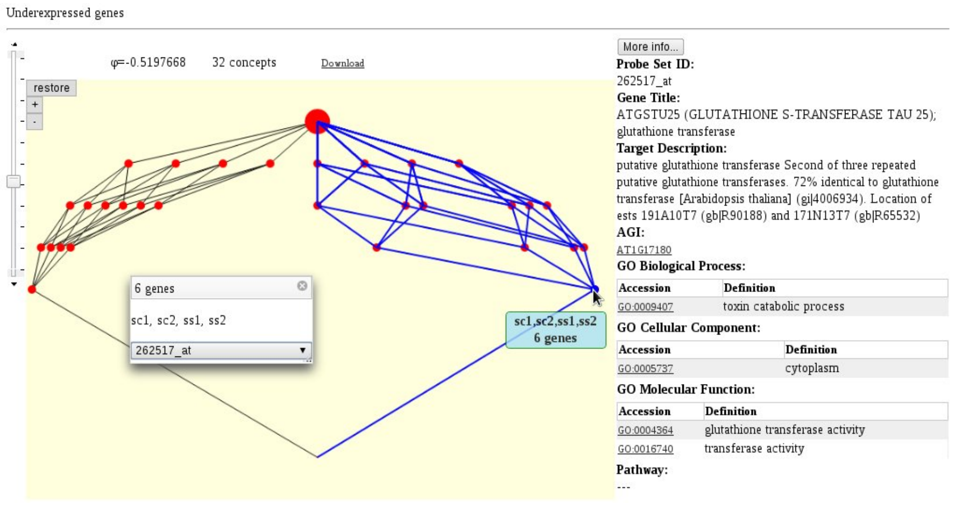
An additional property of each object (gene) can be observed when clicking the button with the legend "More info...". The new view that comes out (Fig. 3) displays the set of concepts the gene appears in through the CL full sequence. We call this the *scope* of the object (gene).

Against the backdrop of the boolean lattice, every concept appearing in any of the CL is rendered and a blue path connects all the concepts that the selected gene has ever belonged to. Hovering with the mouse over one of the blue dots from this path makes a tooltip appear showing the scope in φ (or ϕ) for the gene and that concept. The size of the dot is proportional to the width of the scope, the size of the φ interval. This information is also complemented with the data from the NetAffx Annotation file.

3 Contributions and Further Work

We have presented a Web-based tool to analyse GE data obtained from microarrays and to cluster the genes and test conditions by their similarity in either up- or down-regulation. The system sifts through many CL and saves its results in a database for later reading. The user can inspect through a visual, point-and-click interface the gene expression and related information in Gene Ontology-contextualised CL. Thanks to direct links, it is very easy to gather gene information from other Genomics sites.

Since the basic exploration mechanism is $\mathcal{K}$-Formal Concept Analysis, the user has a wealth of CL that make it difficult to propose and validate data-suggested hypotheses. To curb this exploration complexity we have proposed a



Probe Set ID:
262517_at
Gene Title:
ATGSTU25 (GLUTATHIONE S-TRANSFERASE TAU 25); glutathione transferase
Target Description:
putative glutathione transferase Second of three repeated putative glutathione transferases. 72% identical to glutathione transferase [Arabidopsis thaliana] (gi|4006934). Location of ests 191A10T7 (gb|R90188) and 171N13T7 (gb|R65532)
AGI:
AT1G17180
GO Biological Process:

Accession	Definition
GO:0009407	toxin catabolic process

GO Cellular Component:

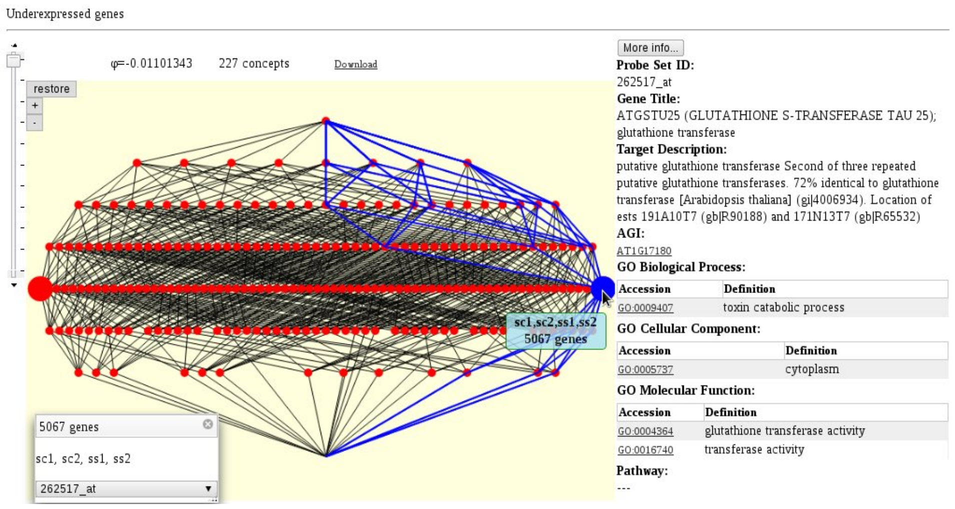
Accession	Definition
GO:0005737	cytoplasm

GO Molecular Function:

Accession	Definition
GO:0004364	glutathione transferase activity
GO:0016740	transferase activity

Pathway:

(a) $\varphi = -0.5197668$



Probe Set ID:
262517_at
Gene Title:
ATGSTU25 (GLUTATHIONE S-TRANSFERASE TAU 25); glutathione transferase
Target Description:
putative glutathione transferase Second of three repeated putative glutathione transferases. 72% identical to glutathione transferase [Arabidopsis thaliana] (gi|4006934). Location of ests 191A10T7 (gb|R90188) and 171N13T7 (gb|R65532)
AGI:
AT1G17180
GO Biological Process:

Accession	Definition
GO:0009407	toxin catabolic process

GO Cellular Component:

Accession	Definition
GO:0005737	cytoplasm

GO Molecular Function:

Accession	Definition
GO:0004364	glutathione transferase activity
GO:0016740	transferase activity

Pathway:

(b) $\varphi = -0.01101343$

Fig. 2: (Colour online) CL and gene description view for two different values of φ. Note the similarity of shapes.

novel visualisation scheme which amounts to the visual embedding of CL in the representation of the most complex lattice pertaining to a set of attributes, viz. the contranominal scale of attributes. It is conceivable that this representation could cater to some other interval-valued parameters like intervals of temporal continua.

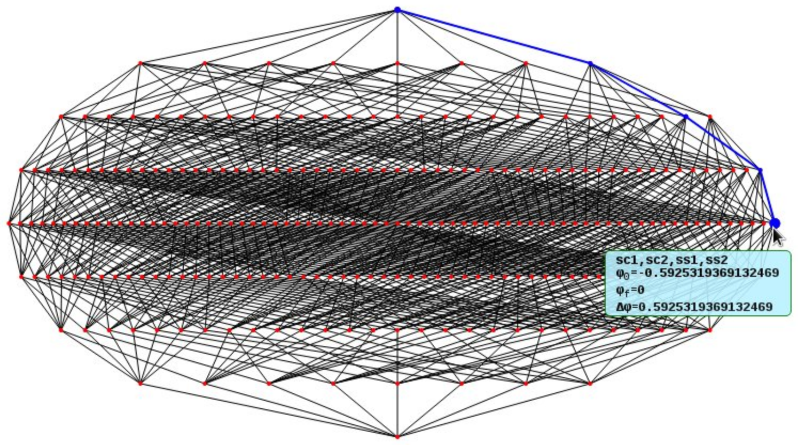


Fig. 3: (Colour online) Gene evolution in a sequence of CLs

We have also proposed a mechanism to aggregate the visualisation mechanisms above that enables the study of the robustness of gene appearance in concepts vs. the variation of thresholds, the scope. We believe that gene scope is an important tool to ascertain the quality of the CL with respect to the "noise" in gene expression values and will explore it in further work.

References

1. Pensa, R., Besson, J., Boulicaut, J.: A methodology for biologically relevant pattern discovery from gene expression data. In Suzuki, E., Arikawa, S., eds.: Discovery Science. Volume 3245 of LNAI., Springer (2004) 230—241
2. González-Calabozo, J., Peláez-Moreno, C., Valverde-Albacete, F.J.: Gene expression array exploration using $\mathcal{K}$-Formal Concept Analysis. In Valtchev, P., Jäschke, R., eds.: Proceedings of the ICFCA11. Volume 6628 of LNAI., Springer (2011) 119–134
3. Valverde-Albacete, F.J., Peláez-Moreno, C.: Extending conceptualisation modes for generalised Formal Concept Analysis. Information Sciences **181** (2011) 1888–1909
4. Affymetrix: Affymetrix CEL Data File Format (2009) http://www.affymetrix.com/support/developer/powertools/changelog/gcos-agcc/cel.html.
5. Eklund, P., Villerd, J.: A survey of hybrid representations of concept lattices in conceptual knowledge processing. In: Formal Concept Analysis:8^th International Conference, ICFCA 2010, Agadir, Morocco (2010) 296–311
6. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin, Heidelberg (1999)
7. Affymetrix: NetAffx Annotation file for *Arabidopsis thaliana* (2012) http://www.affymetrix.com/support/technical/byproduct.affx?product=arab.

Concept Lattices and Median Networks

Uta Priss

Ostfalia University of Applied Sciences
Wolfenbüttel, Germany
www.upriss.org.uk

Abstract. In phylogenetic analysis, median networks have been proposed as an improvement over tree representations. This paper argues that concept lattices represent a further improvement over median networks because FCA provides a detailed formal description and there are a number of existing software solutions for creating lattices. The purpose of this paper is to raise awareness in the FCA community for this interesting application area in bioinformatics.

1 Introduction

The field of phylogenetics tries to establish evolutionary relations among groups of organisms through molecular sequencing, for example, by sampling DNA from organisms and looking at differences. Reconstruction of phylogenetic trees is somewhat hypothetical because evolutionary relationships are established using DNA from currently living organisms. There are established means for inferring such trees using statistical means but in cases where parallel mutations or reversals occur, it is difficult to decide the exact sequence of the mutations. Therefore, instead of deciding which of the possible trees is more likely, one can use a graph which embeds all possible trees. This simplifies the analytic process and leads to more readable diagrams. Bandelt et al. (1995) develop the construction of such graphs into a method using median networks as explained in the next section. Sykes (2001) and Bandelt et al. (1995 and 2000) argue that using median networks is a significant improvement over construction of hypothetical trees using statistical methods.

Since trees can be embedded into lattices, the question arises as to whether Formal Concept Analysis[1] (FCA) can be used instead of or in addition to median networks. One advantage of using FCA is that FCA has a larger research community than median networks/graphs. Furthermore, there exist a variety of well-tested software tools for FCA[2] whereas Bandelt et al. (2000) discuss "manual construction" of median networks alongside some algorithms. For FCA researchers this establishes a further application domain in bioinformatics. The following section provides further details about median networks in phylogenetic analyses. Section 3 discusses how the phylogenetic data can be modelled with FCA and what is different or similar to how the data is modelled with median networks. The paper finishes with a concluding section.

[1] Because this conference is dedicated to FCA, this paper does not provide an introduction to FCA. Information about FCA can be found, for example, on-line (http://www.fcahome.org.uk) and in the main FCA textbook by Ganter & Wille (1999).

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

2 Median networks and phylogenetics

This section provides a very brief introduction to the application area of this paper[3]. Unfortunately, many of the papers in this application area are written for biologists and do not contain mathematically precise definitions of the terms and algorithms. Median graphs are undirected graphs where any three vertices have a unique median which is a vertex that belongs to shortest paths between any two of the three vertices. Examples of median graphs are trees or the Hasse diagrams of distributive lattices if considered as undirected graphs. Median networks are special kinds of median graphs where vertices represent species and parallel edges represent possible genetic changes.

In the field of phylogenetics, evolutionary trees are inferred from observed characteristics of species. Although DNA sequences can be of four values (A, G, C or T), it is unusual for more than one change to occur at the same site in a set of closely related species. Thus, characteristics can be considered binary by only recording whether or not a change occurred. In the case of parallel mutations (or the more rare reversals), it is difficult to know the sequence of the mutations. A median network summarises possible evolutionary trees. In particular, one is interested in “most parsimonious trees” which means that the number of times the endpoints of a tree edge have different values is minimal. Without parallelisms or reversals a median network is a tree. Considering the examples by Bandelt et al. (1995 and 2000), ordinary data sets tend to contain at least some parallelisms. Thus the generated median networks are not usually trees.

A median network is guaranteed to contain all most parsimonious trees (Bandelt et al., 1995). But if the sample size is large, an unmodified median network may be too complex to be graphically represented. Bandelt et al. (1995) suggest a method for reducing median networks based on weight and frequency (where “weight” and “frequency” are defined as follows). In order to construct a median network, one summarises all changes that occur simultaneously with respect to a set of sample species as “weight”. Graphically this can be represented by the length of edges. In the same manner, if several species have the exact same characteristics, one creates only one vertex for this group of species but records a higher frequency for this vertex. This can be graphically represented by a larger node for the vertex. Using frequencies and weights one can reduce the network by eliminating some of the edges which are less likely to have occurred. Bandelt et al. (1995) state that in all examples they considered so far even reduced networks still contained all most parsimonious trees, but there is no guarantee that that is always the case.

3 Modelling with FCA

One advantage of using FCA is the availability of established mathematical vocabulary for describing the phylogenetic phenomena. Important phylogenetic notions can be directly translated into FCA terminology. Series of evolutionary changes that are unambiguous correspond to attribute implications in the lattice. Latent species (which are implied by the data but for which no specimen have been found and which are “latent

[3] Based on Bandelt et al. (1995 and 2000), Sykes (2001) and the Wikipedia page on “Median graphs”.

vertices" in a median network) correspond to concepts that do not contain objects in their contingent but are the join of object concepts. Each meet-reducible concept in the lattice corresponds to a choice point between different possible trees.

An example is the mitochondrial data from Ward et al. (1991) which was also used by Bandelt et al. (1995). As mentioned above the data can be reduced to a single-valued context by using mitochondrial lineages as objects and sites of changes as attributes. Figure 1 shows a concept lattice for a data table discussed by Bandelt et al. 2000 (using HVS I data by Vigilant et al.). Two attributes are called "compatible" in Bandelt's terminology if they are lattice-theoretically comparable or their meet is the bottom node. Bandelt calls a set of attributes a "clique" if the attributes are pairwise compatible and the set is maximal with respect to inclusion. In other words, cliques are maximal trees. In Figure 1 one clique/tree contains all attributes except 16243 and another clique/tree contains all attributes except 16294 and 16239. These are the only two trees in Figure 1. Bandelt et al. describe a fairly complicated algorithm for deriving a median network using cliques, peripheral elements and torsos where the "torso" data matrix consists of the non-compatible attributes.

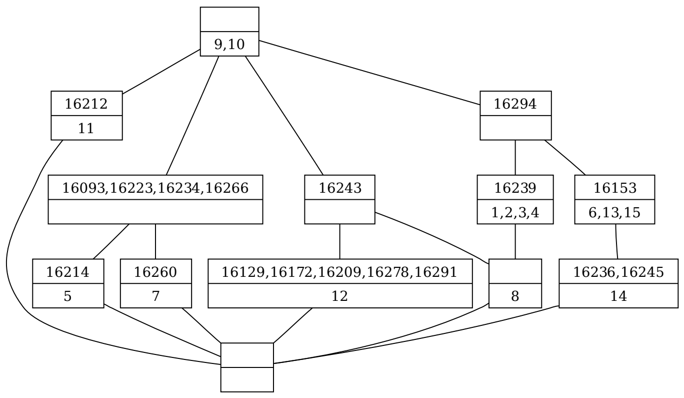


Fig. 1. Concept lattice for HVS I data of Vigilant used by Bandelt et al. (2000)

Figure 2 shows a median network for the data in Figure 1. In contrast to Bandelt et al. (2000), the attributes, frequencies and weights are omitted in the figure. This means that all nodes are of the same size and the length of the edges does not carry meaning. The lattice in Figure 1 and the median network contain essentially the same information apart from the fact that the lattice contains a bottom node and the median network contains a latent vertex in the torso (to the right of the vertex "8") which is due to the shortest path condition of median networks but not needed for lattices. Although we are not providing a formal proof at this point, based on similar construction algorithms it is to be expected that in general the concept lattice and the median network of a data table contain the same information apart from some latent vertices and the bottom node in the lattice.

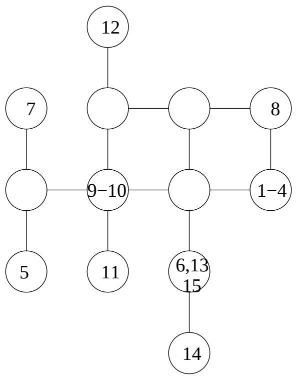


Fig. 2. The median network for Figure 1

4 Conclusion

The aim of this position paper is to stimulate further research into the application of FCA in the bioinformatics domain. It appears that FCA can improve on methods that are currently used in that area and can be used to derive a more consistent and precise terminology. Furthermore, from an FCA view, this application domain raises questions about using frequencies, weights, the construction of latent objects, tree embeddings and attribute splitting which could lead to future FCA research.

References

1. Bandelt, H. J.; Forster, P.; Sykes, B. C.; Richards, M. B. (1995). *Mitochondrial portraits of human populations using median networks.* Genetics, Oct, 141, 2, p. 743-753.
2. Bandelt, H.-J.; Macaulay, V.; Richards, M. (2000). *Median networks: Speedy construction and greedy reduction, one simulation, and two case studies from human mtDNA.* Molecular Phylogenetics and Evolution, 16, 1, p. 8-28.
3. Ganter, Bernhard; & Wille, Rudolf (1999). *Formal Concept Analysis. Mathematical Foundations.* Berlin-Heidelberg-New York: Springer.
4. Sykes, Bryan (2001). *The seven daughters of Eve.* Bantam Press.
5. Ward, R. H.; Frazier, B. L.; Dew-Jacer, K.; Pääbo, S. (1991). *Extensive mitochondrial diversity within a single Amerindian tribe.* Proc. Natl. Acad. Sci., USA, 88, p. 8720-8724.

Author Index

Editors:	Laszlo Szathmary, Uta Priss
Publisher & Print:	Universidad de Málaga (Dept. Matemática Aplicada) Spain
Title:	CLA 2012, Proceedings of the Ninth International Conference on Concept Lattices and Their Applications
Place, year, edition:	Fuengirola (Málaga), Spain, 2012, 1st
Page count:	xii+357
Impression:	75
Archived at:	`http://cla.inf.upol.cz`

Not for sale

ISBN 978–84–695–5252–0