CLA 2013

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

CLA Conference Series

`cla.inf.upol.cz`

Laboratory L3i, University of La Rochelle, France

ISBN 978–2–7466–6566–8

Laboratory L3i, University of La Rochelle, France

The Tenth International Conference on Concept Lattices and Their Applications

CLA 2013

La Rochelle, France
October 15–18, 2013

Edited by

Manuel Ojeda-Aciego
Jan Outrata

CLA 2013

Technical Editor:
Jan Outrata, `jan.outrata@upol.cz`

Cover design:
Elodie Carel, `elodie.carel@univ-lr.fr`

Page count: xii+306
Impression: 50
Edition: 1st
First published: 2013

Published and printed by:
Laboratory L3i, University of La Rochelle, France

Organization

CLA 2013 was organized by the Laboratory L3i, University of La Rochelle.

Steering Committee

Radim Belohlavek	Palacký 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
Engelbert Mephu Nguifo	LIMOS, University Blaise Pascal, Clermont-Ferrand, France
Amedeo Napoli	INRIA NGE/LORIA, Nancy, France

Program Chairs

Manuel Ojeda-Aciego	Universidad de Málaga, Spain
Jan Outrata	Palacký University, Olomouc, Czech Republic

Program Committee

Cristina Alcalde	Univ del Pais Vasco, San Sebastián, Spain
Jaume Baixeries	Polytechnical University of Catalonia, Spain
Radim Belohlavek	Palacký University, Olomouc, Czech Republic
Sadok Ben Yahia	Faculté des Sciences de Tunis, Tunisia
Anne Berry	LIMOS, Université de Clermont Ferrand, France
Karell Bertet	L3i, Université de La Rochelle, France
Ana Burusco	Universidad de Navarra, Pamplona, Spain
Claudio Carpineto	Fondazione Ugo Bordoni, Roma, Italy
Pablo Cordero	Universidad de Málaga, Spain
Jean Diatta	Université de la Réunion, France
Felix Distel	TU Dresden, Germany
Vincent Duquenne	Université Pierre et Marie Curie, Paris, France
Sebastien Ferre	Irisa/Université de Rennes 1, France
Bernhard Ganter	TU-Dresden, Germany
Alain Gély	University of Metz, France
Robert Godin	Univeristy of Montreal, Canada
Marianne Huchard	LIRMM, Montpellier, France
Dmitry I. Ignatov	State University HSE, Moscow, Russia
Vassilis G. Kaburlasos	TEI, Kavala, Greece
Jan Konecny	Palacký University, Olomouc, Czech Republic
Stanislav Krajci	University of P.J. Safarik, Kosice, Slovakia

Sergei O. Kuznetsov	State University HSE, Moscow, Russia
Léonard Kwuida	Bern University of Applied Science, Switzerland
Jesús Medina	Universidad de Cádiz, Spain
Engelbert Mephu Nguifo	LIMOS, University Blaise Pascal, Clermont Ferrand, France
Rokia Missaoui	UQO, Gatineau, Canada
Amedeo Napoli	INRIA NGE/LORIA, Nancy, France
Lhouari Nourine	LIMOS, Université de Clermont Ferrand, France
Sergei Obiedkov	State University HSE, Moscow, Russia
Uta Priss	Ostfalia University of Applied Sciences, Wolfenbüttel, Germany
Olivier Raynaud	LIMOS, University of Clermont Ferrand, France
Sebastian Rudolph	Institute AIFB, University of Karlsruhe, Germany
Baris Sertkaya	SAP Research Center, Dresden, Germany
Laszlo Szathmary	University of Debrecen, Hungary
Petko Valtchev	Université du Québec à Montréal, Canada
Francisco Valverde	Universidad Nacional de Educacin a Distancia, UNED Spain

Additional Reviewers

Michel Liquière	LIRMM, Montpellier, France
María Eugenia Cornejo Piñero	Universidad de Cádiz, Spain
Eloisa Ramírez Poussa	Universidad de Cádiz, Spain
Alexis Irlande	Universidad Nacional de Colombia, Colombia
Juan Carlos Díaz Moreno	Universidad de Cádiz, Spain
Nader Mohamed Jelassi	LIMOS, Clermont Université, France & URPAH, Faculty of Tunis, Tunisia

Organization Committee

Karell Bertet (chair)	L3i, Université de La Rochelle, France
Dwiajeng Andayani	Université de La Rochelle, France
Romain Bertrand	L3i, Université de La Rochelle, France
Mickaël Coustaty	L3i, Université de La Rochelle, France
Ngoc Bich Dao	L3i, Université de La Rochelle, France
Christophe Demko	L3i, Université de La Rochelle, France
Cyril Faucher	L3i, Université de La Rochelle, France
Nathalie Girard	LI, Université de Tours, France
Clément Gurin	L3i, Université de La Rochelle, France
Muhammad Muzzamil Luqman	L3i, Université de La Rochelle, France
Jean-Marc Ogier	L3i, Université de La Rochelle, France
Christophe Rigaud	L3i, Université de La Rochelle, France
Kathy Theuil	L3i, Université de La Rochelle, France
Muriel Visani	L3i, Université de La Rochelle, France

Table of Contents

Preface

Invited Contributions

Full Papers

Short Papers

Preface

Formal concept analysis has, for many years, laid claim to providing a formal basis for an applied lattice theory. With the many different formalisms and implementations, and their applications available today, this claim is stronger than ever, as witnessed by increasing amount and range of publications in the area.

The International Conference "Concept Lattices and Their Applications (CLA)" is being organized since 2002 with the aim of bringing together researchers working on various approaches for studying and practically applying concept lattices. The main aim of CLA is to bring together researchers (students, professors, engineers) involved in all aspects of the study of concept lattices, from theory to implementations and practical applications. As the diversity of the selected papers shows, there is a wide range of theoretical and practical research directions, ranging from algebra and logic to pattern recognition and knowledge discovery.

The Tenth edition of CLA was held in La Rochelle, France from October 15th to October 18th, 2013. The event was organized and hosted by the Laboratory L3i, University of La Rochelle.

This volume includes the selected papers and the abstracts of 4 invited talks. This year there were initially 37 submissions from which 22 included papers were accepted as full papers and 5 as short papers. Thus, the program of the conference consisted of four keynote talks given by the following distinguished researchers: Ralph Freese, Bart Goethals, Michel Grabisch and Vincent Duquenne, together with twenty-seven communications, including the full and short papers, authored by researchers from thirteen countries (Belgium, Chile, Cyprus, Czech Republic, Djibouti, Estonia, France, Germany, Mexico, Russia, Slovakia, Spain and USA).

The papers were reviewed by members of the Program Committee with the help of the additional reviewers listed overleaf. We would like to thank them all for their valuable assistance. It is planned that a selection of extended versions of the best papers will be published in a renowned journal, after being subjected again to a peer review.

The success of such an event is mainly due to the hard work and dedication of a number of people, and the collaboration of several institutions. We want to thank the contributing authors, who submitted high quality works, to acknowledge the help of members of the CLA Steering Committee, who gave us the opportunity of chairing this edition, and to thank the Program Committee, the additional reviewers, and the local Organization Committee. All of them deserve many thanks for having helped to attain the goal of providing a balanced event with a high level of scientific exchange and a pleasant environment.

We would also like to thank the following institutions, which have helped the organization of the 10th CLA International Conference: Region of Poitou-Charentes,

Department of Charente Maritime, City of La Rochelle and University of La Rochelle.
Last but not least, most of our bureaucratic tasks related to paper submission, selection, and reviewing have been minimized thanks to the EasyChair conference system, and we should therefore not forget to mention its help after the list of "official sponsors".

October 2013

Manuel Ojeda-Aciego
Jan Outrata
Program Chairs of CLA 2013

Projective Lattices

Ralph Freese

Department of Mathematics, University of Hawaii
Honolulu, HI 96822, USA
`ralph@math.hawaii.edu`

A lattice **L** is *projective* in a variety $\mathcal{V}$ of lattices if whenever

$$f : \mathbf{K} \twoheadrightarrow \mathbf{L} \tag{1}$$

is an epimorphism, there is a homomorphism

$$g : \mathbf{L} \to \mathbf{K} \tag{2}$$

such that $f(g(a)) = a$ for all $a \in L$.

Projective lattices are characterized in [3] by four conditions. This talk will discuss two of them that are of current interest.

If g in (2) is only required to be order-preserving, it is called an *isotone section* of the epimorphism (1). We will characterize which lattices **L** have an isotope section for every epimorphism (1). We will use this to characterize when the ordinal (linear) sum of two projective lattices in $\mathcal{V}$ will be projective and give some surprising examples.

The second of the four conditions characterizing projectivity we will discuss is join refinement and the dependency relation; the so-called D-relation. This condition and some closely related concepts are used in many parts of lattice theory. Besides free lattice, projective lattices and finitely presented lattices, it has applications to transferable lattices, congruence lattices of lattices, representing finite lattices as congruence lattices of finite algebras, and ordered direct bases in database theory [1, 2].

References

1. K. Adaricheva, J.B. Nation, and R. Rand. rdered direct implicational basis of a finite closure system. *Discrete Applied Math.*, 161:707–723, 2013.
2. K. Bertet and B. Monjardet. The multiple facets of the canonical direct unit implicational basis. *Theoret. Comput. Sci.*, 411(22-24):2155–2166, 2010.
3. Ralph Freese and J. B. Nation. Projective lattices. *Pacific J. Math.*, 75:93–106, 1978.

Cartification: from Similarities to Itemset Frequencies

Bart Goethals

University of Antwerp, Belgium
`bart.goethals@ua.ac.be`

Abstract. We propose a transformation method to circumvent the problems with high dimensional data. For each object in the data, we create an itemset of the k-nearest neighbors of that object, not just for one of the dimensions, but for many views of the data. On the resulting collection of sets, we can mine frequent itemsets; that is, sets of points that are frequently seen together in some of the views on the data. Experimentation shows that finding clusters, outliers, cluster centers, or even subspace clustering becomes easy on the cartified dataset using state-of-the-art techniques in mining interesting itemsets.

Cooperative Games on Lattices

Michel Grabisch

Paris School of Economics
Université Paris I
106-112, Bd de l'Hôpital, 75013 Paris
`michel.grabisch@univ-paris1.fr`

In cooperative game theory, for a given set of players N, TU-games are functions $v : 2^N \to \mathbb{R}$ which express for each nonempty coalition $S \subseteq N$ of players the best they can achieve by cooperation.

In the classical setting, every coalition may form without any restriction, i.e., the domain of v is indeed 2^N. In practice, this assumption is often unrealistic, since some coalitions may not be feasible for various reasons, e.g., players are political parties with divergent opinions, or have restricted communication abilities, or a hierarchy exists among players, and the formation of coalitions must respect the hierarchy, etc.

Many studies have been done on games defined on specific subdomains of 2^N, e.g., antimatroids [1], convex geometries [3,4], distributive lattices [6], or others [2,5]. In this paper, we mainly deal with the case of distributive lattices. To this end, we assume that there exists some partial order $\preceq$ on N describing some hierarchy or precedence constraint among players, as in [6]. We say that a coalition S is feasible if the coalition contains all its subordinates, i.e., $i \in S$ implies that any $j \preceq i$ belongs to S as well. Then feasible coalitions are downsets, and by Birkhoff's theorem, form a distributive lattice. From now on, we denote by $\mathcal{F}$ the set of feasible coalitions, assuming that $\emptyset, N \in \mathcal{F}$.

The main problem in cooperative game theory is to define a rational solution of the game, that is, supposing that the grand coalition N will form, how to share among its members the total worth $v(N)$. The core is the most popular solution concept, since it ensures stability of the game, in the sense that no coalition has an incentive to deviate from the grand coalition. For a game v on a family $\mathcal{F}$ of feasible coalitions, the core is defined by

$$\mathcal{C}(v) = \{x \in \mathbb{R}^n \mid x(S) \geq v(S), \forall S \in \mathcal{F}, x(N) = v(N)\}$$

where $x(S)$ is a shorthand for $\sum_{i \in S} x_i$. When $\mathcal{F} = 2^N$, the core is either empty or a convex bounded polyhedron. However, for games whose cooperation is restricted, the study of the core becomes much more complex, since it may be unbounded or even contain no vertices (see a survey in [7]). For the case of games with precedence constraints, it is known that the core is *always* unbounded or empty, but contains no line (i.e., it has vertices). The problem arises then, to select a significant bounded part of the core as a reasonable concept of solution, since unbounded payments make no sense. We propose to select a bounded face of the core. A systematic study of bounded faces is done through the concept of normal collections.

We also present some results when $\mathcal{F}$ is not a distributive lattice, but a set lattice closed under intersection, or a regular set system.

Lastly, we introduce games on concept lattices, show that this induces in fact two games, and give some results on the core.

References

1. E. Algaba, J. M. Bilbao, R. van den Brink, and A. Jiménez-Losada. Cooperative games on antimatroids. *Discrete Mathematics*, 282:1–15, 2004.
2. S. Béal, E. Rémila, and Ph. Solal. Rooted-tree solutions for tree games. *European Journal of Operational Research*, 203(2):404–408, 2010.
3. J. M. Bilbao. Axioms for the Shapley value on convex geometries. *European Journal of Operational Research*, 110:368–376, 1998.
4. J. M. Bilbao, E. Lebrón, and N. Jiménez. The core of games on convex geometries. *European Journal of Operational Research*, 119:365–372, 1999.
5. U. Faigle, M. Grabisch, and M. Heyne. Monge extensions of cooperation and communication structures. *European Journal of Operational Research*, 206:104–110, 2010. 10.1016/j.ejor.2010.01.043.
6. U. Faigle and W. Kern. The Shapley value for cooperative games under precedence constraints. *Int. J. of Game Theory*, 21:249–266, 1992.
7. M. Grabisch. The core of games on ordered structures and graphs. *Annals of Operations Research*, Vol. 204 (2013), 33-64.
8. M. Grabisch. Ensuring the boundedness of the core of games with restricted cooperation. *Annals of Operations Research*, 191:137–154, 2011.
9. M. Grabisch and P. Sudhölter. On the restricted cores and the bounded core of games on distributive lattices. Technical Report 2012.67, Centre d'Economie de la Sorbonne, Paris, 2012. http://ideas.repec.org/s/mse/cesdoc.html.
10. M. Grabisch and P. Sudhölter. The bounded core for games with precedence constraints. *Annals of Operations Research*, Vol. 201 (2012), 251-264. doi: 10.1007/s10479-012-1228-9.
11. M. Grabisch and L. J. Xie. The restricted core of games on distributive lattices: how to share benefits in a hierarchy. *Mathematical Methods of Operations Research*, 73:189–208, 2011.

Some applications of Lattice Analysis (1983-2013)

Vincent Duquenne

CNRS-IMJ / C&O, Université Pierre et Marie Curie,
4 place Jussieu, 75005 Paris, France

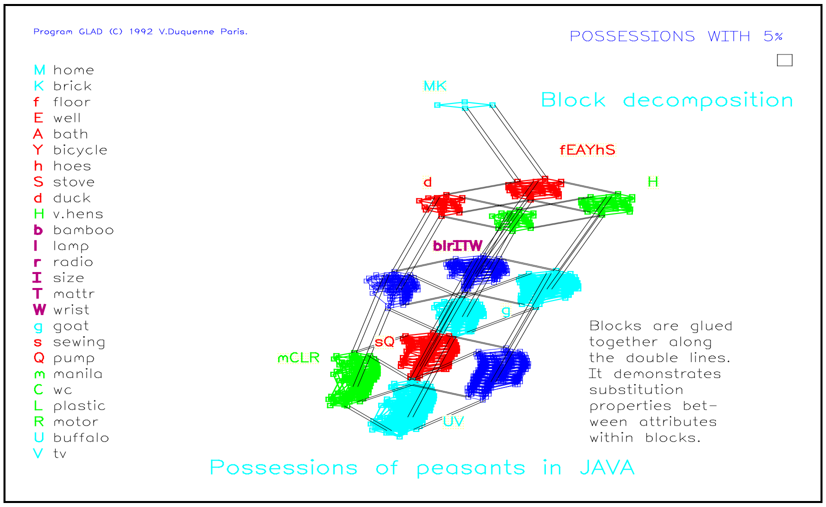
duquenne@math.jussieu.fr

Abstract. Following [1,2] we report on applications of Lattice Analysis either for deciphering data or for clarifying abstract lattices. Here, lattices are often considered as implication models that can be summarized with canonical basis [3,1,4,5] or (semi) lattice cores [1]. In a more symmetric way decompositions through lattice congruence / tolerance relations are used for real data analysis as well as for getting understandable structures of abstract lattices [6,7 and below]. As for the needed algorithms, many efforts have been done to "overtake" the NEXT-CLOSURE algorithms since their discovery in 1984 [5]. For implications the fees may involve an exponential explosion in memory. We will just try to give some visions of what could be next in doing with(-out) NEXT-CLOSURE. Hence in a fresh original spirit of the early eighties, for all these and further developments we still promote "more simplicity with more structure" (and tolerances ...) for deepening the concept systems and lattice applications.

Keywords: closure operator, lattice, canonical basis of implications, quasi / pseudo-closed, (semi) lattice cores, perspectivities / arrows, congruences / tolerances, combinatorial exhaustive enumeration, NEXT-CLOSURE algorithms.

References

1. Duquenne, V.: Contextual implications between attributes and some representation properties for finite lattices. In Ganter, B., Wille, R., Wolf, K. (eds.), Beitrage zur Begriffsanalyse, 213-239, Wissenschaftsverlag, Mannheim: (1987), reprinted in **ICFCA'2013**.
2. Duquenne V., Latticial structures in Data Analysis, *Th. Comp. Sci. 217* (1999) 407-436.
3. Guigues J.-L., Duquenne.V.: Familles minimales d'implications informatives résultant d'un tableau de données binaires. Mathématiques & Sciences Humaines 95, 5-18, (1986).
4. Duquenne, V.: Some Variations on Alan Day's Algorithm for Calculating Canonical Basis of Implications. In Diatta, J., Ecklund, P., Liquière, M. (eds.), **CLA'2007**, 17-25.
5. Ganter, B.: Algorithmen zur Formalen Begriffsanalyse. In Ganter, B., Wille, R., Wolf, K. (eds.), Beitrage zur Begriffsanalyse, , Wissenschaftsverlag, Mannheim: 241-255 (1987).
6. Duquenne V. and A. Cherfouh, On permutation lattices, *Math. Soc. Sci.* 27 (1993) 73-89.
7. V. Duquenne, Lattice Drawings and Morphisms, **ICFCA'2010**, *Lecture Notes in Artificial Intelligence 5986* (L. Kwuida and B. Sertkaya eds) 2010, 88-103.



Blocks are glued together along the double lines. It demonstrates substitution properties between attributes within blocks.

Fig. 1. Peasants x possessions. Lattice gluing decomposable, hence substitution properties... From : Models of possessions and Lattice Analysis, *Social Sci. Information* (1995).

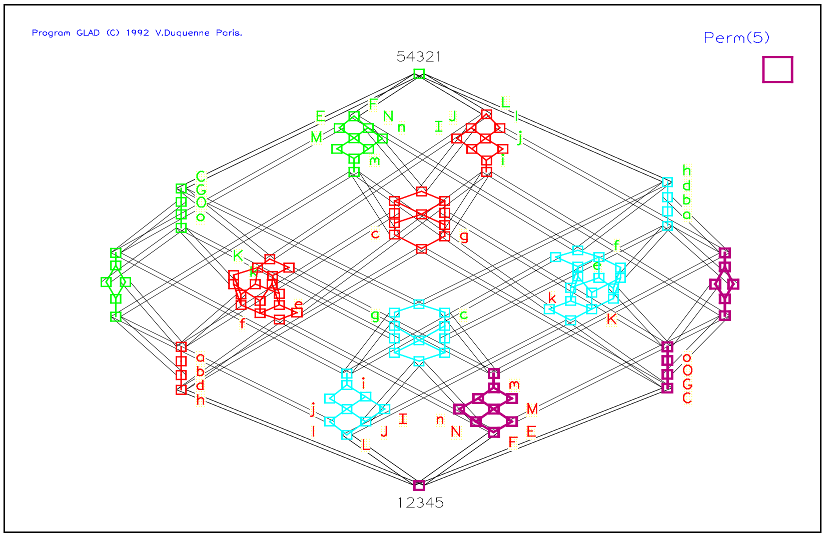


Fig. 2. Perm(5) quotiented by the meet of its maximal congruences: "having the same complements". From : On permutation lattices, *Mathematical Social Sciences* (1994).

A practical application of Relational Concept Analysis to class model factorization: lessons learned from a thematic information system

A. Osman Guédi[1,2,3], A. Miralles[2], M. Huchard[3], and C. Nebut[3]

[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

Abstract. During the design of class models for information systems, databases or programming, experts of the domain and designers discuss to identify and agree on the domain concepts. Formal Concept Analysis (FCA) and Relational Concept Analysis (RCA) have been proposed, for fostering the emergence of higher level domain concepts and relations, while factorizing descriptions and behaviors. The risk of these methods is overwhelming the designer with too many concepts to be analyzed. In this paper, we systematically study a practical application of RCA on several versions of a real class model for an information system in order to give precise figures about RCA and to identify which configurations are tractable.

Keywords: Class model, class model factorization, Formal Concept Analysis, Relational Concept Analysis

1 Introduction

Designing class models for information systems, databases or programs is a common activity, that usually involves domain experts and designers. Their task consists in capturing the domain concepts, and organizing them in a relevant specialization structure with adequate abstractions and avoiding redundant concepts. Formal Concept Analysis (FCA) and its variant Relational Concept Analysis (RCA) have been proposed to assist this elaboration phase, so as to introduce new abstractions emerging from the identified domain concepts, and to set up a factorization structure avoiding duplication. FCA classifies entities having characteristics[1], while RCA also takes into account the fact that entities are linked by relations. The concept lattices produced can be exploited so as to obtain the generalization structure for the class model.

Nevertheless, while this whole factorization structure of a class model is a mathematical object with strong theoretical properties, its practical use might

---

[1] The usual terms in the FCA domain are "objects" and "attributes"; we prefer not using them here because they conflict with the vocabulary of class models.

suffer from limitations due to the large size of the obtained lattices. In such a case, domain experts might be overwhelmed by the produced information making it difficult (or even impossible) to use it to improve the class model.

In this paper we want to assess, in a real case study, the size of the factorization results, in order to have a solid foundation for proposing practical recommendations, tools or approaches. We work with a kind of "worst case" of RCA application, by using all the modeling elements and not limiting our investigation to some elements (like classes and attributes, or classes and operations). We show, via various selected graphics, how RCA behaves. Our experiments indicate which configurations are tractable, admitting that some tools present results in a fine way, and which configurations lead to quite unusable results.

The rest of the paper is structured as follows. Section 2 briefly explains how FCA and RCA can contribute to a class model design. Section 3 settles the environment for our experiments and introduces our case study. Section 4 presents and discusses the obtained results. Section 5 presents related work, and section 6 concludes.

2 Concept lattices in class model refactoring

In this section, we explain how concept lattices implement and reveal the underlying factorization structure of a class model. We also show how this property can be exploited for class model refactoring, and in particular: generating new reusable abstractions that improve the class organization and understanding, especially for domain experts, limiting attribute and role duplication.

Formal Concept Analysis [5] is a mathematical framework that groups entities sharing characteristics: entities are described by characteristics (in a Formal Context), and FCA builds (formal) concepts from this description. In Relational Concept Analysis (RCA) [9], the data description consists of several Formal Contexts and relations. The main principle of RCA is to iterate on FCA application, and the concepts learnt during one iteration for one kind of entity are propagated through the relations to the other kinds of entities. The concepts are provided with a partial order which is a lattice. In the obtained concept lattices, we distinguish *merged concepts* and *new concepts*. A *merged concept* is a concept that has more than one entity in its simplified extent. This means that the entities of the simplified extent share the same description. A *new concept* is a concept that has an empty simplified extent. This means that no entity has exactly the simplified intent of the concept as set of characteristics: Entities of the whole extent own the characteristics of the simplified intent in addition to other characteristics.

To apply FCA to class models, we encode the elements of a class model into formal contexts. For example, we provide a context describing the classes by their attributes. The FCA approach then reveals part of the factorization structure and supports part of the refactoring process by using a straightforward description of UML elements. For example, we can discover *new concepts* for classes interpreted as new super-classes, factorizing two attributes.

Nevertheless this approach does not fully exploit the deep structure of the class model. Let us take the example of Figure 1(a). The attribute `name` is duplicated in classes `B1` and `B2`, and FCA can generate the model of Figure 1(b) that introduces a new class (here manually named `NamedElement`) that factorizes this attribute. However, FCA does not compute the factorization that can be found in Figure 1(c), in which a class called `SuperA` factorizes the two associations from `A1` to `B1` and from `A2` to `B2`, being given that now `B1` and `B2` have an ancestor `NamedElement`.

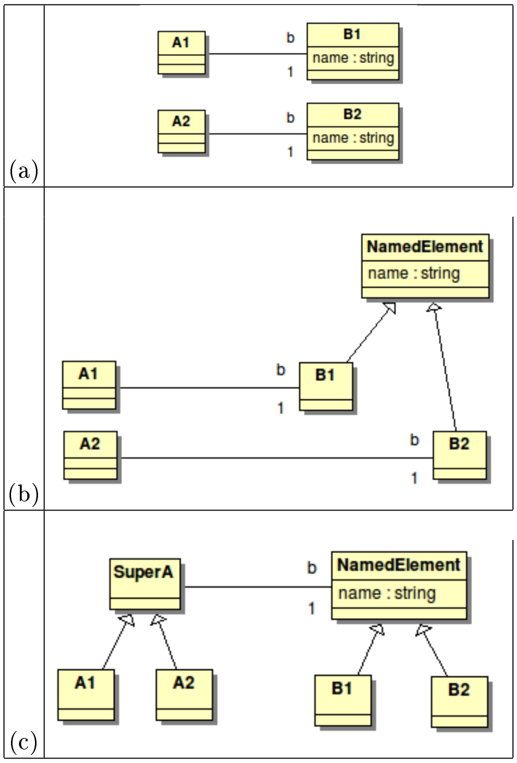


Fig. 1. Example of factorization in class models

Extracting abstractions using this deep structure can be done with RCA, which builds the entire factorization structure, including information on the elements (classes, attributes, associations) and their relations. RCA uses the fact that classes are linked through associations. In the first iteration step, RCA computes the factorization in Figure 1(b), and then propagates the *new concept* `NamedElement` through the association between classes. Then the factorization of Figure 1(c) is computed during the next iteration steps. The process stops there since a fixpoint is found (no new abstractions can be found).

The obtained structure contains no duplication, and improves the organization of the model. However, when applied on large data, RCA may result in the introduction of many *new concepts*, that may be too abstract, and/or too many to be analyzed. That is why in the next sections, we investigate on a case study the behavior of RCA for a large class model corresponding to an actual information system. Our objective is to determine if RCA remains suitable for large class models, and how to configure RCA to obtain exploitable results.

3 The Pesticides class model and experimental setup

Our case study is a class model which is part of a project from the Irstea institute, called Environmental Information System for Pesticides (EIS-Pesticides). It aims at designing an information system centralizing knowledge and information produced by two teams: a Transfer team in charge of studying the pesticide transfers from plots to rivers and a Practice team which mainly works on the agricultural practices of the farmers. The domain analysis has been carried on during series of meetings with one team or both teams. Fifteen versions of this class model have been created during this analysis. Figure 2 shows the number of model elements over the versions.

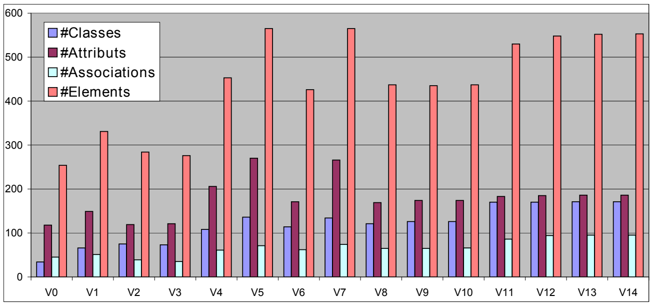


Fig. 2. The number of model elements over the various versions

Our tool is based on the Modeling Tool Objecteering[2] and the framework eRCA[3]. eRCA has been extended for computing metrics. In this paper we focus on a configuration (part of the meta-model) including the following entities described in formal contexts: classes, associations, operations (very few in the *Pesticides* model), roles and attributes. Their characteristics are their names. The relations describe: which class owns which attribute, which class owns which operation, which class owns which role, which association owns which role and which type (class) has a role. When applying RCA to this configuration, we obtain 5 concept lattices, one for each formal context. We also consider four

---

[2] http://www.objecteering.com/

[3] http://code.google.com/p/erca/

parameterizations for this configuration depending on whether we take into account navigability and undefined elements. If a navigability is indicated on an association, meaning that objects from the source know objects from the target (not the reverse), taking into account navigability (denoted by `Nav`) results in the following encoding: the source class owns the corresponding role, but the target class does not own any role corresponding to that association. Not taking into account navigability (denoted by `noNav`) means that the source class and the target class own their respective role in the association. In the modeling tool, unnamed roles are named "undefined". We can choose to include this "undefined" name in the contexts (denoted by `Undef`) or not (denoted by `noUndef`).

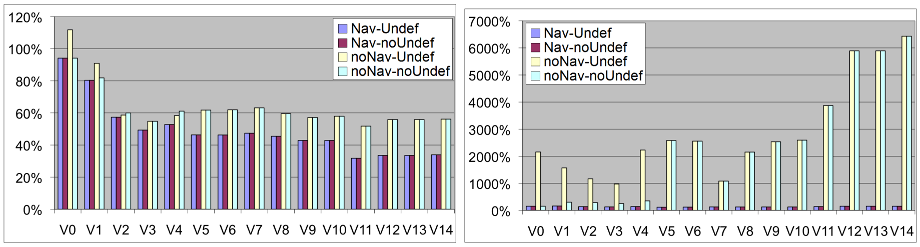
4 Results

In this section, we report the main results that we obtain. We consider two special iteration steps: step 1 (close to FCA application) and step 6 (where paths of length 6 in the model are followed, meaning that abstractions on classes created at step 1 have been exploited to create other class abstractions through roles and associations). At step 1 for example, common name attributes are used to find new superclasses. At step 4, new superclasses can be found as shown in Figure 1(c), and 2 steps later, new super-associations can be found from the class concepts found at step 4. We examine, for classes and associations, which are the main elements of the model, metrics on *new* class concepts and *new* associations concepts (Section 4.1), then on *merge* class and association concepts (Section 4.2). Execution time is presented in Section 4.3, and we conclude this part by giving indications about the number of steps when the process reaches the fixpoint.

4.1 New abstractions

We focus first on the *new concepts* that appear in the class lattice and in the association lattice. They will be interpreted as new superclasses or as new generalizations of associations. In a collaborative work, these concepts are presented to the experts who use some of them to improve the higher levels of the class model with domain concepts not explicit until then. This is why their number is important; if too many new abstractions are presented to the experts, these experts might be overwhelmed by the quantity, preventing a relevant and efficient use of the method.

Figure 3 (left-hand side) shows the *new concepts* in the class lattice (thus the new superclasses) at step 1, when paths of size 1 have been traversed. For example, this means that if some classes have attributes (or roles) of the same name in common, those attributes (or roles) will certainly be grouped in a *new* class concept. This new class concept can be presented to the expert to control if this corresponds to a new relevant class abstraction (or superclass). We notice that `Nav` parameterizations produce less *new concepts* than `noNav` ones. This is due to the fact that `noNav` parameterizations induce much more circuits in the analyzed data, increasing the number of RCA steps and the number of generated concepts. The number of *new concepts* decreases as the analysis process progresses.



STEP 1 STEP 6

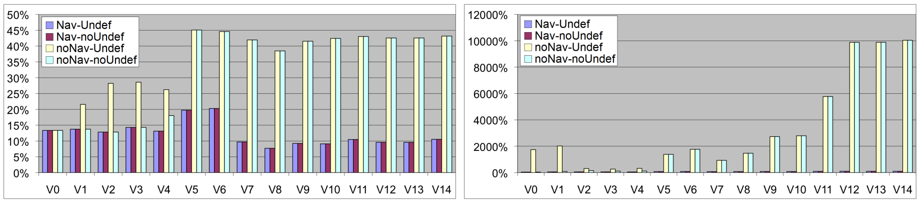
Fig. 3. New class abstractions created at step 1 and 6 v.s. the number of initial classes

In the best case (of percentage of new superclasses), 32% of new potential superclasses will be presented to the experts, for the model V11 which contains 170 classes, giving 54 new potential superclasses. In the worst case, we have 112% of new potential superclasses, for V0 model, which has 34 classes, thus only 38 new potential superclasses are found. At this stage, we do not see a serious difference between the four parameterizations.

Results obtained at step 6 are much more difficult to deal with. Figure 3 (right-hand-side) shows that the two parameterizations `noNav` (generating more cycles in data) give results that will need serious filtering to separate relevant *new concepts* from the large set of *new concepts*. `Nav` parameterizations will produce less than one and a half the initial number of classes, while `noNav` parameterizations can produce up to 10998 class concepts, really requiring either additional post-treatments or avoiding to generate all the concepts.

Figure 4 (left-hand side) shows the *new concepts* in the association lattice at step 1. They represent associations that are at a higher level of abstraction. Experts can examine them, to see if they can replace a set of low-level associations. In `Nav` parameterizations, at most 15 higher level associations are presented to experts; in `noNav` parameterizations, the number grows until 32, remaining very reasonable to analyze.

Figure 4 (right-hand side) shows the *new concepts* in the association lattice at step 6. It highlights the fact that, at this step, the number of these concepts may explode, and it is especially high in the last versions of the class model, in which we initially have many associations. The number of new association concepts, in `Nav` parameterizations, is less than a hundred, and it still remains reasonable (even if it is higher than in step 1), but in `noNav` parameterizations it dramatically grows and may reach about 9500 concepts.



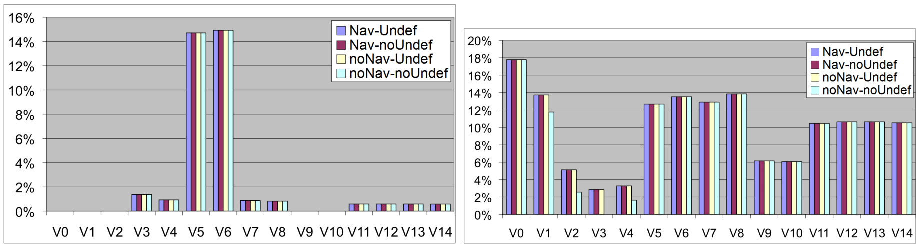
Step 1 Step 6

Fig. 4. New association abstractions created at step1 and 6 v.s. the number of initial associations

4.2 Merged concepts

Merged concepts are concepts which introduce several entities (*e.g* classes or associations) in their extent. Such entities share exactly the same description in the model. For example, a merge class concept can group classes that have exactly the same name attributes. This common description is first detected at step 1, then it does not change because the following steps refine the description by adding new relational characteristics and concepts; entities remain introduced in the same concepts. For classes and associations, the merged concept number is the same for the four analysis configurations. For experts, analyzing a *merged concept* consists in reviewing the simplified extent and examining if the entities (class or association) have been exhaustively described or effectively correspond to a same domain concept.

Figure 5 (left-hand side) presents metrics for merge class concepts. V5 and V6 have a higher percentage of *merged concepts* because during analysis, a package has been duplicated at step 5 for refactoring purpose. The duplicated classes have been removed at step V7. In the other cases, there are not so much merge class concepts to be presented to the experts, between 0% and 2%, giving a maximum of two classes. This often corresponds to classes with incomplete description, that the experts should develop into more details. The low number of such cases makes the task of experts easy.



Ratio # merge class concepts on # initial classes

Ratio # merge association concepts on # initial associations

Fig. 5. Merge class concept and merge association concepts vs. initial elements

Figure 5 (right-hand side) presents metrics for merge association concepts. The percentage of merge association concepts is higher than the percentage of merge class concepts. This is explained by the fact that associations are only described by roles, that occasionally share the same names (identical to some class names). It varies between about 2% and 18%, meaning that at most 10 merge association concepts are presented to the experts for evaluation, making a little bit more complicated the analysis task compared to the case of classes, but it remains very reasonable.

4.3 Execution time, used RAM and total number of steps

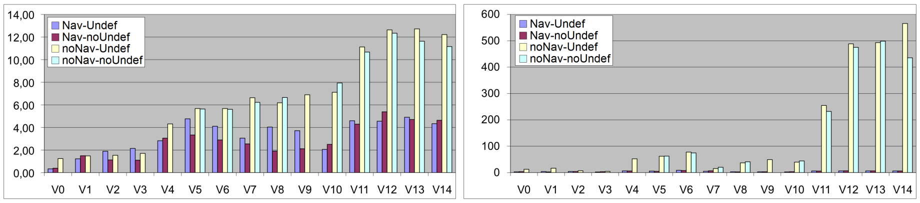
Experimentations have been performed on a cluster composed of 9 nodes, each one having 8 processors Intel (R) Xeon (R) CPU E5335 @ 2.00GHz with 8 Go of RAM. The operating system was Linux (64 bits) and the programs are written in Java.

Figure 6 shows the execution time in seconds, at step 1 and at step 6. At step 1, the execution time for the two `Nav` parameterizations are below 6 seconds, while for the two `noNav` parameterizations, for some versions (especially when there are more associations, like in the last versions) it may reach about 13 seconds. At step 6, the execution time for the two `Nav` parameterizations are below 8 seconds. But for the `noNav` parameterizations, we notice longer executions, up to 10 minutes. However, such a task does not require an instantaneous answer, and has not to be carried out too many times. Even if it occurs during an expert meeting, it can be admitted to spend a few minutes for constructing the concept lattices.

Table 1. Figures on used memory (in MegaBytes)

Step	Parameters	min	max	average
Step 1	Nav-Undef	39	453	237
	Nav-noUndef	17	471	205
	noNav-Undef	41	969	480
	noNav-noUndef	24	969	532
Step 6	Nav-Undef	44	471	213
	Nav-noUndef	6	403	140
	noNav-Undef	33	1846	656
	noNav-noUndef	33	1147	520

Table 1 shows the RAM used during execution, here again, `noNav` parameterizations are the worst, reaching about 2 GigaBytes of used memory. In the case of `Nav` parameterizations, it is interesting to observe that there is not a significant difference between step 1 and step 6.



STEP 1 STEP 6

Fig. 6. Execution time at step 1 and 6 (in seconds)

Figure 7 shows, for the `Nav-noUndef` parameterization the total number of steps needed to reach the fix-point, and the size of a longest path with no repeated arcs (such a path can be a cycle). We observe that the step number (from 6 to 16) is always below the size of the longest simple path which gives in our context a practical upper bound to the number of steps. This means that if we dispose in the future of relevant filtering strategies, we can envisage studying *new concepts* appearing after step 6.

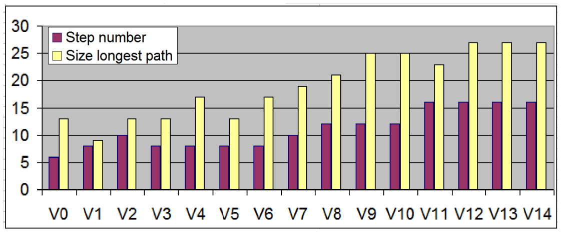


Fig. 7. Step number (first) and longest simple path size (second) in C2, `Nav-noUndef` parameterization

4.4 Discussion

During this study, we observed that analyzing *merge* class concepts and *merge* association concepts was a feasible task in all parameterizations. The analysis of *new* class concepts and *new* associations concepts is more difficult. `Nav` parameterizations produce exploitable results with a maximum of about 50 class concepts (resp. about 30 new association concepts) to be examined at step 1. At step 6, experts may have to face from one to three hundreds of new class concepts (resp. about one hundred of new association concepts). Execution time and used memory are not problematic issues and we get a practical upper bound to the number of steps in this kind of data, which is given by the size of a longest simple path.

These observations may be the starting point of an efficient steering method for a collaborative class model construction. The objective of such a method would be to insert between each model release, an RCA-based analysis, in order to accelerate the discovery of new abstractions, and the completion of elements (highlighting of the *merged concepts*). Both the concepts and the structure of the lattice are useful for the experts to determine which relevant modifications should be applied. The strength of the lattice representation is that it provides a structure adapted to the task of choosing the right new abstractions, since concepts can be presented following the specialization order, depending of what is the demand of experts.

Known effects of some parameterizations can serve to favor different steering strategies. The longest simple path size gives a bound on the step number, giving an heuristic to decide when to stop the RCA process, with an idea about how far we are from the convergence. `Nav` parameterizations can be easily controlled by looking, at each step, the appearing concepts (and marking the non-relevant ones to avoid finding them again at next steps). If information is expected from `NoNav` parameterizations, experts have to be very careful because many concepts will be created. A particular sub-order of the concept lattice (the AOC-poset), induced by the concepts that introduce an entity or a characteristic, might offer an important reduction of the produced concept numbers, without loosing main information about factorization.

Another track is limiting input data, for example by removing attributes that have limited semantic value and to group concepts declaring few attributes.

There are of course some limits to the generalization of our conclusions. The class model is mainly a data model (very few operations), destined to build a database schema and we study various versions of a same class model. Nevertheless, the *Pesticides* model is a classical model, representative of the models that are built in the environmental field.

5 Related work

The use of FCA in the domain of class model refactoring has a long and rich history in the literature. As far as we know, it has been introduced in the seminal paper of Godin et al. [6] for extracting abstract interfaces from a Smalltalk class hierarchy and extensions of the work have been published in [7]. Other approaches have been proposed, that take into account more information extracted from source code like super calls and method signatures in [2].

In [1], authors report an application of RCA to several medium-size class models of France Télécom (now Orange Labs). The RCA configuration was composed of classes, methods, attributes, and associations. Classes have no description; attributes are described by name, multiplicity and initial value; methods are described by name and method body; associations are described by names, role name and information like multiplicity and navigability. Associations are connected to their origin and destination classes, classes are connected to the attributes and operations they own, attributes are connected to classes that are their type, etc. The class models contain a few dozens of classes and the *new concepts* to be examined by experts varies from a few concepts to several hundreds.

In [11] detailed figures are given. In the largest project (57 classes), the number of *new concepts* was 110 for the classes, 9 for the associations and 59 for the properties. In this paper, we have several class models that are greater in terms of number of classes and we reify the role notion, rather than encoding it in the association description, with the risk of having more produced concepts. We discard technical description (like multiplicity which has no strong semantics). We also analyze the *merged concepts*, another interesting product of RCA.

In [8], RCA has been applied on an excerpt of the class model of the Jetsmiles software of JETSGO society[4]. The class model excerpt is composed of only 6 classes, connected by 9 associations, and about 30 attributes. Attributes and roles are mixed, and classes are connected by a relational context to attributes+roles, while attributes+roles are connected by another relational context to their type (when it is a class). The UML elements are described by many technical features: multiplicity, visibility, being "abstract", initial value, etc. A post-treatment analyzes the built concepts in order to keep the most relevant ones. The class concept lattice contains about 35 concepts while the attribute+role concept lattice has about 25 concepts. In [8], the size of the class model is very small. We suspect that using the configuration with many technical elements would not be scalable in the case of the *Pesticides* model.

RCA has been experimented on two Ecore models, two Java programs and five UML models in [4]. The used configuration is composed of the classes and the properties (including attributes and roles) described by their names and their connections. To report some representative results of this experiment, in Apache Common Collections, which is composed of 250 classes, RCA finds 34 new class concepts and less than 80 new property concepts; in UML2 metamodel, which is composed of 246 classes and 615 properties, RCA extracts 1534 new class concepts and 998 new property concepts. In this experiment, associations were not encoded, contrary to what we do. Nevertheless an explosion of the concept number yet appears. In our case, we introduce associations in the configuration, and we show, that with some precautions as annotating by navigability and naming the roles, refactoring with data including the associations may remain feasible.

A more recent study [3] compared three strategies of FCA/RCA application to part of the open-source Java Salome-TMF software which comprises 37 classes and 142 attributes[5]. In the RCA strategy (ARC-NAME), a formal context includes the classes (no description), a formal context describes the attributes by their names and the hyperonyms of their names, and relations connect classes and their attributes (and reversely). ARC-NAME produces 33 new class concepts, and 3 merge class concepts, 21 new attribute concepts and 13 merge attribute concepts. Java softwares do not have associations and it is difficult to generalize these results to the general case of class models. Compared to this work, here we do not use linguistic information, this will be done in a future work, nevertheless the terms in the model are not technical identifiers but rather do-

---

[4] http://www.jetsgo.net/

[5] http://wiki.ow2.org/salome-tmf/

main terms carefully selected by the expert group, thus there are less problems in using them directly.

Here we use the same class models as in [10] where RCA based metrics were proposed to assist a designer during the evolution of the class model in indicating him the evolution of the level of description and the level of abstraction.

6 Conclusion and perspectives

In this paper, we describe the most advanced study of the application of RCA on class models, so as to obtain a relevant factorization structure. We apply RCA on several versions of the model of the same information system (from 40 to 170 classes), and we study the impact of several parameters in the application. The objective was to observe RCA on real-sized class models, so as to draw conclusions, mainly on its scalability. The experiment shows that taking into account the navigability, it is still possible to analyze the newly introduced abstractions. Consequently, RCA can be considered to scale to real-size models, if it is adequately parameterized. However, the produced results remain quite large to analyze, and new strategies can be settled to face the number of concepts to analyze.

References

1. Dao, M., Huchard, M., Hacene, M.R., Roume, C., Valtchev, P.: Improving Generalization Level in UML Models Iterative Cross Generalization in Practice. In: ICCS 2004. pp. 346–360 (2004)
2. Dicky, H., Dony, C., Huchard, M., Libourel, T.: On automatic class insertion with overloading. In: OOPSLA 96. pp. 251–267 (1996)
3. Falleri, J.R.: Contributions à l'IDM : reconstruction et alignement de modèles de classes. Ph.D. thesis, Université Montpellier 2 (2009)
4. Falleri, J.R., Huchard, M., Nebut, C.: A generic approach for class model normalization. In: ASE 2008. pp. 431–434 (2008)
5. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundation. Springer-Verlag Berlin (1999)
6. Godin, R., Mili, H.: Building and maintaining analysis-level class hierarchies using Galois lattices. In: OOPSLA 93. pp. 394–410 (1993)
7. Godin, R., Mili, H., Mineau, G., Missaoui, R., Arfi, A., Chau, T.: Design of Class Hierarchies Based on Concept (Galois) Lattices. Theory And Practice of Object Systems 4(2) (1998)
8. Hacene, M.R.: Relational concept analysis, application to software re-engineering. Ph.D. thesis, Universite du Quebec A Montreal (2005)
9. Hacene, M.R., Huchard, M., Napoli, A., Valtchev, P.: Relational concept analysis: mining concept lattices from multi-relational data. Ann. Math. Artif. Intell. 67(1), 81–108 (2013)
10. Osman Guédi, A., Miralles, A., Amar, B., Nebut, C., Huchard, M., Libourel, T.: How relational concept analysis can help to observe the evolution of business class models. In: CLA 2012. pp. 139–150 (2012)
11. Roume, C.: Analyse et restructuration de hiérarchies de classes. Ph.D. thesis, Université Montpellier 2 (2004)

Galois Sub-Hierarchies Used for Use Case Modeling

Ants Torim[1]

Tallinn University of Technology, Ehitajate tee 5, 19086 Tallinn, Estonia
`ants.torim@ttu.ee`

Abstract. Use case diagrams are the core diagrams of the Unified Modeling Language (UML), de facto standard for software modeling. They are used to visualize relations between the users (Actors) and the functionality of the software system (Use Cases). Galois sub hierarchy (GSH) is a sub-order of the concept lattice that contains only concepts with object or attribute labels. This paper investigates the viability of GSH for visualizing the information contained within use case diagrams. While it is possible that a GSH diagram is more complex than a use case diagram for certain formal contexts a study of 87 student projects found no such case. On average, use case diagrams had 3.7 times more graphical elements than corresponding GSH diagrams, demonstrating the viability of GSH as a more compact alternative to the use case diagram.

1 Introduction

Software engineering has a long tradition of graphical modeling, there are many different diagram types like flowcharts, BPMN, ERD and even languages like UML containing over dozen diagram types. Most of these diagrams have elements connected with directed or non-directed connections. Each element is represented as a node and each connection as a line between these nodes. While this approach is easy to understand and apply, methods of Formal Concept Analysis (FCA) like Galois sub hierarchies (GSH) can represent the same information in a more concise way, significantly reducing the number of graphical elements. This is achieved because a single node in GSH diagram can represent several elements (it can have many labels) and a line can represent many connections. This conforms to the "reduce redundant data-ink" principle from E. Tufte's classic work on visual information displays [19]. GSH diagram makes it easy to see which elements have same connections and which element has a subset of other elements connections inviting interesting comparisons.

GSH diagrams are most natural to use when there are two types of elements and connections between these (a bipartite graph). This applies to the UML use case diagram that describes actors, use cases and connections between them. A study presented here compares the GSH approach with the UML use case diagram. There is also a brief overview about describing the connections between use cases and data tables with diagrams, information present in CRUD matrix, another traditional software engineering artifact.

2 Galois Sub-Hierarchies

Our method of visual representation is based on Galois sub- hierarchy (GSH) diagrams from the field of Formal Concept Analysis (FCA). This article uses some FCA terminology (formal concept, concept lattice, extent, intent) without explanation and definitions, these can be found from many foundational articles of this field [21], [9], [22], [23] . GSH is a subset of the concept lattice that contains only labeled concepts. More formally, concept (A, B) from the formal context (G, M, I) belongs to the GSH if and only if for some object $g \in G$, $(A, B) = (\{g\}'', \{g\}')$, or dually, for some attribute $m \in M$, $(A, B) = (\{m\}', \{m\}'')$. GSH as a tool for software engineering was introduced by Godin and Mili [10] for the purpose of class hierarchy re-engineering.

This work differs from the standard FCA practice as the main area of interest is not finding the concepts but visualizing the connections between the elements of G and M in a concise way. Semantics of G and M can vary: objects and attributes, use cases and actors, use cases and data tables. Users of GSH diagrams would need to be acquainted with the following properties to see the connections between G and M:

1. GSH diagrams show nodes (concepts), connections between them and labels from the sets G and M attached to the nodes. Each element from G and M has exactly one corresponding label.
2. $g \in G$ is connected to $m \in M$ iff there is an upward path from label g to label m or they are labels of the same node.
3. If $g_1, g_2 \in G$ and there is an upward path from g_1 to g_2 then the set of elements g_2 is connected to, g_2', is a subset of g_1'.
4. Dually, if $m_1, m_2 \in M$ and there is a downward path from m_1 to m_2 then the set of elements m_2 is connected to, m_2', is a subset of m_1'.
5. If $g_1, g_2 \in G$ and g_1 and g_2 are labels of the same node then $g_2' = g_1'$.
6. Dually, if $m_1, m_2 \in M$ and m_1 and m_2 are labels of the same node then $m_2' = m_1'$.

Figure 1 presents a simple formal context where $G = \{1, 2, 3, 4, 5, 6\}$ and $M = \{a, b, c, d, e, f\}$ and its corresponding GSH diagram.

	a	b	c	d	e	f
1		x			x	
2	x					
3					x	
4				x		x
5		x				
6			x	x		x

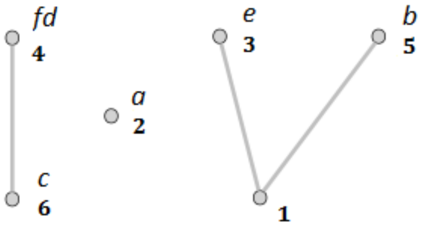


Fig. 1. A formal context with the corresponding GSH diagram.

There are several tools for concept lattice generation: Concept Explorer [24], ToscanaJ [4], GaLicia [20]. Of these, only GaLicia supports Galois sub-hierarchies, but its labeling scheme is not convenient for our purposes as its labels contain full concept intents and extents, therefore same element can appear many times in different labels. Two freely available tools were developed as bachelor theses, supervised by the author. One is *GSH builder* by Kristo Aun [3], another is *GSH* by Maarja Raud [15]. Both generate GSH diagrams that show the labels, not extents or intents.

3 Use Cases and Actors

Use Case modeling is a common tool for specifying functional requirements. An actor is something with behavior (person, computer system) who interacts with our system. A use case is a collection of related success and failure scenarios that describe an actor using a system to support a goal [13]. A detailed description of the use case is given in a text document while the use case diagram shows a general overview: actors and their relationships to use cases. Use case diagram can also show include and extend relations between use cases. A use case diagram is a diagram type within Unified Modeling Language. There are many books written about the topic including [16] and [13].

Following example (Figure 2) is redrawn from Craig Larmans partial use case diagram describing NextGen sales system: a computerized application used to record sales and handle payments in a retail store [13](pp. 29, 71). This is a basic use case diagram showing use cases, actors and connections between them.

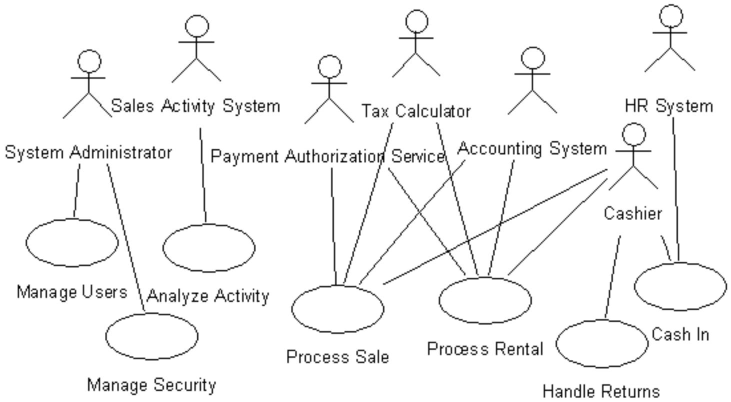


Fig. 2. Use case diagram. Actors, like System Administrator, are shown as stick figures. Use cases, like Manage Users, are shown as ovals. Actors participation within a use case is shown as a line. Spatial arrangement conveys no information here, unlike in diagrams of FCA.

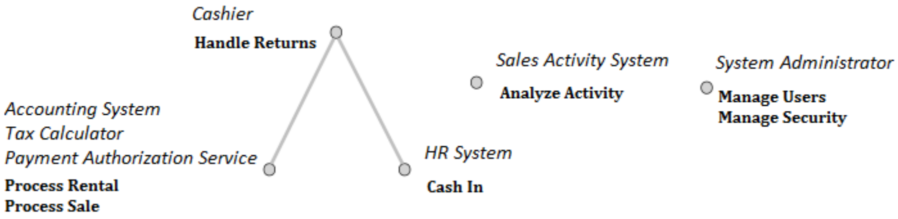


Fig. 3. GSH diagram showing connections between use cases and actors.

Figure 3 shows a GSH diagram, equivalent to the use case diagram from Figure 2. It is more concise with only 5 nodes and 2 lines, compared to 14 nodes and 14 lines from Figure 2. A comparison of diagram types based on counting the number of visual elements may seem simplistic but it is in accordance with the principle stated by E. Tufte in his influential work on information displays [19]: " erase redundant data-ink, within reason". Possible reasons for redundancy being: "giving a context and order to complexity, facilitating comparisons over various parts of data, perhaps crafting an aesthetic balance." It is much easier to see from GSH diagrams the actors that are related to same use cases, for example Accounting System, Tax Calculator and Payment Authorization System. GSH diagram makes also visible subset relationships between the use case sets that actors participates in, for example Cashier can do anything that a HR System can do. Therefore GSH diagram both reduces the data ink and compares favorably with the use case diagram in giving a context and order to complexity and facilitating comparisons over various parts of data.

Use case diagrams can contain relations between the use cases or between the actors. Relating use cases is described by C. Larman [13] as "an optional step to possibly improve their comprehension or reduce duplication" GSH diagram showing relations between actors and use cases can not contain this information. Figure 4 presents an example about generalization and include relationships. Generalization is shown as a relation with a big arrowhead from less general subtype to more general supertype. Subtype inherits relations that its supertypes have. Actor *Moderator* is a subtype of an actor *User* and thus inherits its connection to *Post a comment* use case. Generalization relation between use cases is defined dually. Generalization relations can be used to reduce the number of connections within the use case diagrams. While defining formal contexts we add inherited relations to the subtypes.

Include and extend relations between use cases describe sub-functionality: more complex use case includes the behavior of a smaller use case. They allow to introduce different levels of abstraction: A. Cockburn [6] defines three common levels of abstraction: *summary*, *user goal* (default level) and *sub-function* level.

He also mentions *very high summary* (used rarely) and *too low* (should not be used) abstraction levels. S. Ambler recommends to avoid more than two levels of use case associations [1]. Use cases *Post a comment* and *Delete inappropriate comment* include a common *Log in* sub-functionality. Use case *Manage comments* includes use cases *Post a comment* and *Delete inappropriate comment*. Levels of abstraction different from the default *user goal* level are shown through UML stereotypes.

Extend and include arrows correspond to the direction of reference within the use case documentation. In the case of an include relation, use case containing the sub-functionality has a reference to it, in the case of an extend relation, the sub-functionality has a reference to the use case containing it. Extend relation is treated here as an include relation going to the opposite direction.

A method used here to deal with the include and extend relations is to focus on a single level of abstraction (preferably *user goal* or *summary*). Use cases at higher or lower levels of abstractions are removed and their relations to actors are added to the use cases they have include/extend relations with. This can introduce superfluous relations: it is impossible to deduce from the use case diagram if a particular actor from the higher level use case participates in certain sub-functionality or not, use case text has to be examined for that.

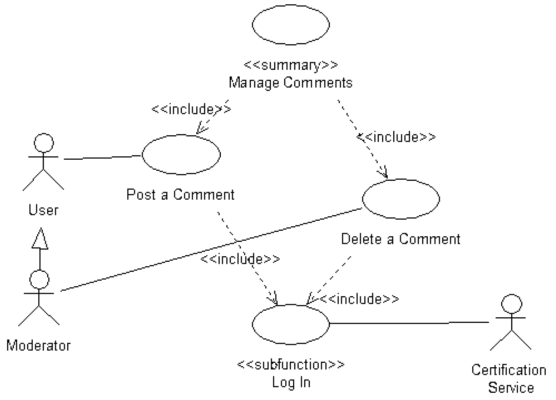


Fig. 4. Use case diagram with generalization and include relations.

Figure 5 shows how previous diagram (Figure 4) has been flattened as described to the *user goal* abstraction level through the removal of generalization and include relations and is now in the form that can be used for GSH generation.

GSH diagrams scale well when the number of use cases increases. Figure 6 is based on the example project *Chair in University* from the course "Introduction to information systems" in Tallinn University of Technology. Original documen-

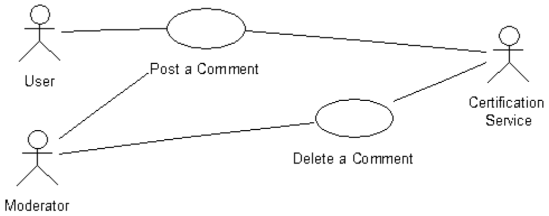


Fig. 5. Previous use case diagram flattened to the user goal level of abstraction.

tation had entire use case model split into four use case diagrams containing 25 use cases, 3 actors and 30 connections between actors and use cases. These 4 diagrams are not reproduced here due to limited space. Equivalent GSH diagram contains 5 nodes and 4 lines. That is a significant improvement in conciseness.

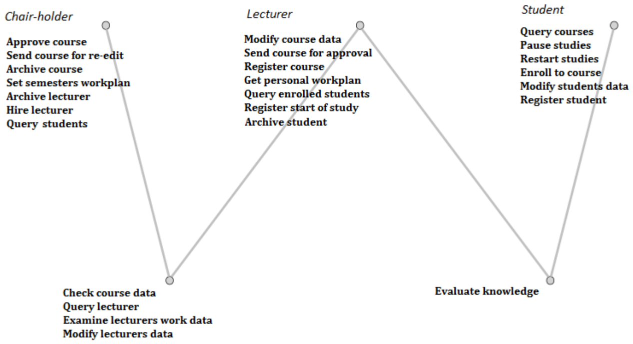


Fig. 6. Labelled line diagram for Chair in University information system.

In the previous examples GSH diagrams have all been simpler (less nodes, less connections) than use case diagrams. It is easy to see that GSH diagram can have no more nodes than the corresponding use case diagram as each GSH node must have at least one label and labels don't repeat. However, for certain contexts, GSH diagram can have more connections than the corresponding use case diagram, as shown in the following example.

Figure 7 shows a formal context with 16 connections and a corresponding GSH diagram with 18 connections. That raises a question if GSH diagrams are really simpler than use case diagrams for practical applications. Following study tries to answer this.

	A	B	C	D
1	x	x		
2	x		x	
3	x			x
4		x	x	
5		x		x
6			x	x
7	x	x	x	x

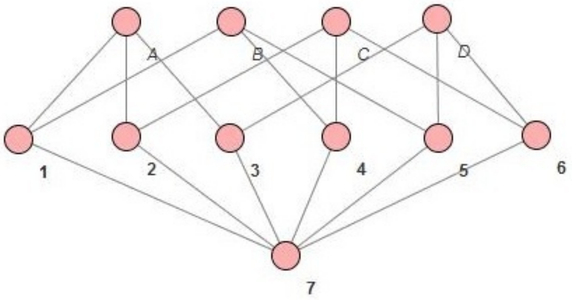


Fig. 7. A formal context with a corresponding GSH diagram that has more lines than the number of original connections.

4 Study

A study of 87 student projects, that were presented to the author for 2012 "Introduction to information systems" course, was completed to compare the use case and GSH diagrams. Student projects contained the analysis documentation for a freely chosen information system, including use case diagrams. Some of these projects were done in groups and were larger and there was also a variation of effort and quality. For all these projects a corresponding GSH diagram was generated, based on its use case diagram.

Some diagrams contained generalization relations between actors. In this case sub-actors inherited all the relations from super-actors in the corresponding formal context. Some diagrams contained ≪include≫relationships between use cases. For these cases, only use cases at the *user goal* level were kept, use cases included in these and their connections with actors were merged into the *user goal* level use cases as described in the previous section. Use case and connection counts are for diagrams after removing the use cases not at the *user goal* level of abstraction but before the removal of generalization relations. Generalization relation is counted as one line.

Table 1. Results of the study. UC: number of use cases, A: number of actors, L: number of lines in the use case diagram, GSH_C: number of concepts in GSH, GSH_L: number of lines in GSH.

	UC	A	$UC + A$	L	$UC + A + L$	GSH_C	GSH_L	GSH_{C+L}
Minimum	3	2	6	4	10	2	0	2
Average	10.77	3.55	14.32	14.86	29.18	4.64	3.2	7.84
Maximum	27	9	32	40	68	13	14	24

Ratio between the average number of elements of the use case diagrams and the GSH diagrams is $(UC + A + L)/GSH_{C+L} = 3.72$.

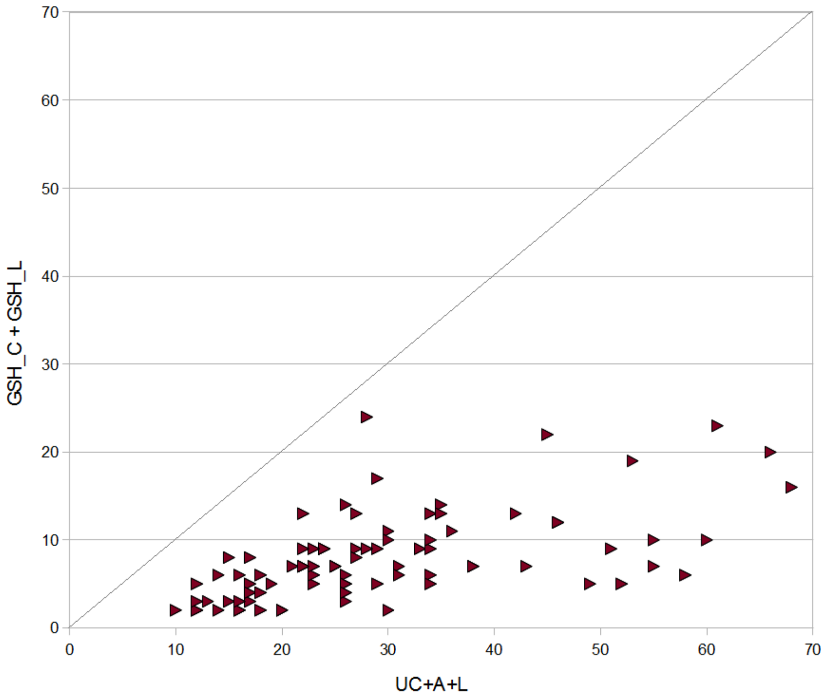


Fig. 8. Scatter plot showing the number of visual elements (use case and GSH diagrams) for the 87 student projects.

Figure 8 shows the scatter plot of student projects showing the number of visual elements on use case and GSH diagrams. It is easy to see that in all cases GSH diagram was simpler than use case diagram, as all the data points lie below the diagonal $(UC + A + L) = GSH_{C+L}$ line. This confirms that, at least for the information systems with 30 or less use cases, GSH diagrams are much more concise than the use case diagrams.

Use case diagrams have their own advantages. They are easier to sketch and modify by hand and they are easier to decompose into several diagrams. That seems to suggest complementary roles for use case and GSH diagrams, use case diagrams for quick sketching and GSH diagrams for a well-formated and concise view of the system.

5 CRUD matrix

Use cases are not only connected to the actors who require such a functionality but they operate on data tables. GSH diagrams are useful for modeling these connections too. CRUD matrix is a well known artifact of software engineering that describes relations between data tables and use cases. It is described in several popular books about systems design [7] and databases [14]. Use of CRUD matrix as a basis for GSH diagrams was described by author in [18]. It is shortly summarized here to show the usefulness of GSH diagrams for different software engineering activities. There are 4 basic actions performed on data tables by use cases: (C)reate, (R)ead, (U)pdate, (D)elete. In some variations use cases are replaced with actors or business processes.

Table 2 contains a CRUD matrix for a simplified library system. Columns correspond to data tables and rows to use cases. Letters c, r, u, d inside the cells of the matrix correspond to 4 basic actions. For example, use case Add New Task reads data from the table Employee and creates (adds) new data to the table Task.

Table 2. CRUD matrix for a simplified library system. Reused from previous article [18].

	Employee	Reader	Task	Loan	EmployeePosition	Book
Manage readers		crud				
Manage employees	crud				cd	
Manage books						crud
Add loan				c		r
Add new task	r		c			
Return loaned book				u		r

We can think of a CRUD matrix as defining dependencies between use case layer and data layer. To describe only dependencies we introduce a new binary matrix where all entries with no actions in the original CRUD matrix are replaced with 0 and all entries with at least one action are replaced with 1. We refer to such a matrix as a usage matrix. Table 3 is a usage matrix for Table 2.

It is obvious that usage matrix, and therefore GSH diagram, can be generated automatically from CRUD matrix. That kind of tool could provide visual representation of CRUD matrix without extra effort from the tool user.

Figure 9 is a GSH diagram based on the usage matrix from Table 3. From the GSH diagram it is much easier to see the elements with same dependencies, like use cases *Add loan* and *Return loaned book* and disconnected subsystems, like use case *Manage Readers* with data table *Reader*. GSH diagrams are also helpful for detecting hidden similarities/isomorphisms between different contexts. It is

Table 3. Usage matrix for a simplified library system. Based on Table 2.

	Employee	Reader	Task	Loan	EmployeePosition	Book
Manage readers		x				
Manage employees	x				x	
Manage books						x
Add loan				x		x
Add new task	x		x			
Return loaned book				x		x

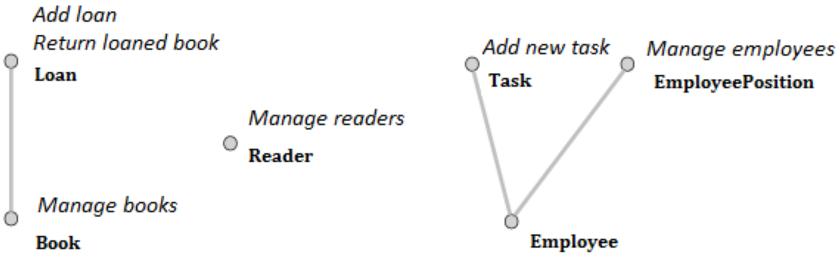


Fig. 9. Labelled line diagram based on Table 3.

much easier to see that GSH diagrams from Figures 1 and 9 are isomorphic than that matrices from Figure 1 and Table 3 are isomorphic.

6 Related work

The use of methods of FCA for software engineering is not a novel idea. A thorough survey of FCA support for software engineering activities is given in [17]. Most of such research is about extracting potential class hierarchies from different contexts.

Dolques et al [8] propose a FCA-based method for simplifying use case diagrams through the introduction of new generalizations. The result of this method is a refactored use case diagram. Their method preserves the information of include and extend relations. Reduction of diagram elements seems to be smaller than with GSH based method reviewed here though these results are hard to compare exactly as they present them in the terms of edge density (ratio of existing edges to all possible edges). It is possible that the edge density goes down while the actual number of edges increases when new generalized use cases and actors are introduced.

Wolfgang Hesse and Thomas Tilley [12] use a concept lattice connecting use cases and ”things” as a tool for identifying candidate classes for object oriented design. There has been much research into using FCA and GSH for class hierarchy design [10], [11]. Algorithms for GSH generation are compared in the article by Arévalo et al [2], a newer algorithm Hermes is presented by Berry et al [5].

7 Conclusions

GSH diagrams can be a concise replacement for UML use case diagrams. In our study they had 3.7 times less visual elements. GSH diagrams are likely to be useful wherever there are connections between two types of elements: actors and use cases, use cases and data tables, use cases and classes, business processes and use cases and so on.

One area for further research is the software engineering activity of grouping elements into subsystems. Similar use cases can be grouped into functional subsystems, similar data tables can be grouped into registers. GSH and concept lattice diagrams can help here by organizing elements by similar connections. Use cases that depend on the same data tables are likely to be similar. Grouping elements with similar connections into same subsystems would also help to minimize the connections at subsystem level. GSH diagrams can also be used to visualize the subsystem level connections, thus promising to be a quite universal tool for software engineering.

References

1. Ambler, S., W. (2005): The Elements of UML 2.0 Style. Cambridge University Press.
2. Arévalo, G., Berry, A., Huchard, M., Perrot, G., Sigayret, A. (2007): Performances of Galois Sub-Hierarchy-Building Algorithms. Formal Concept Analysis, LNCS, vol. 4390, 166-180.
3. Aun, K.: Galois sub-hierarchy builder. [WWW] http://sourceforge.net/projects/gshbuilder/ (15.09.2013).
4. Becker, P., Hereth, J., Stumme, G. (2002): ToscanaJ - an Open Source Tool for Qualitative Data Analysis. Advances in Formal Concept Analysis for Knowledge Discovery in Databases. Proc. Workshop FCAKDD of the 15th European Conference on Artificial Intelligence, 1-2.
5. Berry, A., Huchard, M., Napoli, A., Sigayret, A. (2012): Hermes: an Efficient Algorithm for Building Galois Sub-hierarchies. CLA 2012: 21-32.
6. Cockburn, A. (2000): Writing Effective Use Cases. Boston: Addison-Wesley.
7. Dennis, A., Haley Wixom, B., Roth, R. (2008): Systems Analysis and Design, 4th ed. Wiley.
8. Dolques, X., Huchard, M., Nebut, C., Reitz, P. (2012): Fixing Generalization Defects in UML Use Case Diagrams. Fundam. Inform. 115(4): 327-356.
9. Ganter, B.,Wille, R. (1998): Formal Concept Analysis, Mathematical Foundations. Berlin: Springer.
10. Godin, R., Mili, H. (1993): Building and Maintaining Analysis-Level Class Hierarchies using Galois Lattices. Proceedings of OOPSLA 28, 394-410.

11. Godin, R., Valtchev, P. (2005): Formal Concept Analysis-Based Class Hierarchy Design in Object-Oriented Software Development. Formal Concept Analysis 2005, LNCS, vol. 3626, 304-323.
12. Hesse, W., Tilley, T. (2005): Formal Concept Analysis Used for Software Analysis and Modelling. Formal Concept Analysis 2005, LNCS, vol. 3626, 259-282.
13. Larman, C. (2002): Applying UML and Patterns, 2nd ed. Upper Saddle River, NJ: Prentice Hall.
14. Oppel, A. J. (2004): Databases Demystified. New York: McGraw-Hill.
15. Raud, M.: GSH. [WWW] http://staff.ttu.ee/ torim/fca.html (15.09.2013).
16. Rumbaugh, J., Jacobson, I., Booch, G. (1999): The Unified Modelling Language Reference Manual. Boston: Addison Wesley.
17. Tilley, T., Cole, R., Becker, P., Eklund, P. (2005): A Survey of Formal Concept Analysis Support for Software Engineering Activities. Formal Concept Analysis 2005, LNCS, vol. 3626, 250-271.
18. Torim, A. (2011): A Visual Model of the CRUD Matrix. Proceedings of the 21th European-Japanese Conference on Information Modelling and Knowledge Bases. (Ed.) Jaak Henno, Yasuhi Kiyoki, Takehiro Tokuda, Naofumi Yoshida. Tallinn: TTU Press, 114 - 122.
19. Tufte, E. R. (2001): The Visual Display of Quantitative Information, 2nd ed. Cheshire, Connecticut: Graphics Press.
20. Valtchev, P., Grosser, D., Roume, C., Hacéne, R. (2003): Galicia: an Open Platform for Lattices. Using Conceptual Structures: Contrib. to the 11th ICCS, 241-254.
21. Wille, R.: Restructuring lattice theory (1982): an approach based on hierarchies of concepts. Ordered Sets, pp. 445-470.
22. Wille, R., Stumme, G., Ganter, B. (2005): Formal Concept Analysis: Foundations and Applications, Berlin: Springer.
23. Wille, R. (2005): Formal Concept Analysis as Mathematical Theory of Concepts and Concept Hierarchies. Formal Concept Analysis 2005, LNCS, vol. 3626, 47-70.
24. Yevtushenko, S. A. (2000): System of Data Analysis "Concept Explorer" (In Russian). Proceedings of the 7th national conference on Artificial Intelligence KII, 127-134.

Computing Similarity Dependencies with Pattern Structures

Jaume Baixeries[1], Mehdi Kaytoue[2], and Amedeo Napoli[3]

[1] Universitat Politècnica de Catalunya. 08032, Barcelona. Catalonia.
[2] Université de Lyon. CNRS, INSA-Lyon, LIRIS. UMR5205, F-69621, France.
[3] LORIA (CNRS - Inria Nancy Grand Est - Université de Lorraine), B.P. 239, F-54506, Vandœuvre-lès-Nancy.
`jbaixer@lsi.upc.edu,mehdi.kaytoue@insa-lyon.fr,amedeo.napoli@loria.fr`

Abstract. Functional dependencies provide valuable knowledge on the relations between the attributes of a data table. To extend their use, generalizations have been proposed, among which purity and approximate dependencies. After discussing those generalizations, we provide an alternative definition, the similarity dependencies, to handle a similarity relation between data-values, hence un-crisping the basic definition of functional dependencies. This work is rooted in formal concept analysis, and we show that similarity dependencies can be easily characterized and computed with pattern structures.

1 Introduction

In the relational database model, functional dependencies (FDs) are among the most popular types of dependencies [19] since they indicate a functional relation between sets of attributes: the values of a set of attributes are determined by the values of another set of attributes. To handle errors and uncertainty in real-world data, alternatives exist. *Approximate Dependencies* [12] are FDs that hold in a part –which is user defined– of the database. *Purity Dependencies* [15] express the relationship on the relative impurity induced by two partitions of the table (generated by two sets of attributes). If the impurity is zero, we have a FD.

These generalizations do not necessarily capture the semantics of some patterns that may hold in a dataset. This motivates the definition of "Similarity Dependencies", which can be seen as a generalization of Functional Dependencies, but un-crispring the basic definition of FDs: similar values of an attribute determine similar values of another attribute. Similarity has been considered for FDs under several terms, e.g. fuzzy FDs [3], matching dependencies [16], constraint generating dependencies [2]. Moreover, it is still an active topic of research in the database community [4,8,16,17].

The main objective of the present article is to give a characterization of similarity dependencies within FCA [10], thanks to the formalism of pattern structures [9]. Indeed, characterizing and computing FDs is strongly related to lattice theory and FCA. For example, the lattice characterization of a set of FDs is studied in [5,6,7], while a characterization within a formal context in

FCA is proposed in [10]. The latter is based on a *binarization*, which is the transformation of the original set of data into a binary context. To overcome the burden usually induced by such a transformation, pattern structures [9] have emerged as a valuable alternative to avoid arbitrary transformations and complexity problems [1].

Accordingly, our purpose here is threefold. Firstly, we propose a definition of *Similarity Dependencies*, and secondly a formalization based on pattern structures in FCA, avoiding a transformation of data into a binary table. It follows that classical algorithms of FCA can be –almost directly– applied to compute similarity dependencies. This work is based on [1] where FDs are characterized thanks to pattern structures, and on [13] where similarity is introduced in pattern structures as a tolerance relation (reflexive, symmetric, but not transitive). Finally, we also report preliminary experiments showing the capabilities of the approach.

The paper is organized as follows. In Section 2 we introduce the definition of Functional, Approximate and Purity Dependencies. In Section 3 we propose a definition and a characterization of Similarity Dependencies with pattern structures. Finally, Section 4 reports preliminary experimental results showing the capabilities of our approach.

2 Functional, Approximate and Purity Dependencies

2.1 Notation

We deal with datasets which are sets of tuples. Let $\mathcal{U}$ be a set of attributes and Dom be a set of values (a domain). For the sake of simplicity, we assume that Dom is a numerical set. A tuple t is a function $t : \mathcal{U} \mapsto Dom$ and then a table T is a set of tuples. Usually a table is presented as a matrix, as in the table of Example 1, where the set of tuples (or objects) is $T = \{t_1, t_2, t_3, t_4\}$ and $\mathcal{U} = \{a, b, c, d\}$ is the set of attributes.

The functional notation allows to associate an attribute with its value. We define the functional notation of a tuple for a set of attributes X as follows, assuming that there exists a total ordering on $\mathcal{U}$. Given a tuple $t \in T$ and $X = \{x_1, x_2, \ldots, x_n\} \subseteq \mathcal{U}$, we have:

$$t(X) = \langle t(x_1), t(x_2), \ldots, t(x_n) \rangle$$

In Example 1, we have $t_2(\{a, c\}) = \langle t_2(a), t_2(c) \rangle = \langle 4, 4 \rangle$. In this paper, the set notation is usually omitted and we write ab instead of $\{a, b\}$.

Example 1. This is an example of a table $T = \{t_1, t_2, t_3, t_4\}$, based on the set of attributes $\mathcal{U} = \{a, b, c, d\}$.

id	a	b	c	d
t_1	1	3	4	1
t_2	4	3	4	3
t_3	1	8	4	1
t_4	4	3	7	3

We are also dealing with the set of partitions of a set. Let S be any arbitrary finite set, then, $\mathrm{Part}(S)$ is the set of all possible partitions that can be formed with S. The set of partitions of a set is a lattice [11]. We recall that partitions can also be considered as equivalence classes induced by an equivalence relation.

Now, we define the set of the "maximal subsets" of a set.

Definition 1. *Given a finite base set S and $X = \{X_1, X_2, \ldots, X_n\}$ a set of subsets of S, a subset X_i is maximal in X if there does not exist any other subset X_j in X such that $X_i \subset X_j$.*

Then X_{Max} is the set of the maximal subsets of X.

For example, let $S = \{a, b, c\}$ and $X = \{\{a, b\}, \{b, c\}, \{a\}, \{b\}\}$. Then X is a subset of $\wp(S)$ the powerset of S, but not all elements of X are maximal subsets. Indeed, $X_{Max} = \{\{a, b\}, \{b, c\}\}$.

Moreover, we define the function max_S which applies to a set of sets such as X and returns the set of maximal subsets of X, i.e. X_{Max}.

Definition 2. *Given a finite set S and a subset $X = \{X_1, X_2, \ldots, X_n\}$ of $\wp(S)$, the function max_S returns the set X_{Max} of maximal subsets of X:*

$$max_S(X) = X_{Max} = \{X_i \in X \mid \nexists X_j \in X : X_i \subset X_j\}$$

2.2 Functional Dependencies

We now introduce functional dependencies (FDs).

Definition 3 ([19]). *Let T be a set of tuples (or a data table), and $X, Y \subseteq \mathcal{U}$. A* functional dependency (FD) *$X \to Y$ holds in T if:*

$$\forall t, t' \in T : t(X) = t'(X) \Rightarrow t(Y) = t'(Y)$$

For example, the functional dependencies $a \to d$ and $d \to a$ hold in the table of Example 1, whereas the functional dependency $a \to c$ does not hold since $t_2(a) = t_4(a)$ but $t_2(c) \neq t_4(c)$.

There is an alternative way of considering Functional Dependencies using partitions of the set of tuples T. Taking a set of attributes $X \subseteq \mathcal{U}$, we define the partition of tuples induced by this set as follows.

Definition 4. *Let $X \subseteq \mathcal{U}$ be a set of attributes in a table T. Two tuples t_i and t_j in T are equivalent w.r.t. X when:*

$$t_i \sim t_j \iff t_i(X) = t_j(X)$$

Then, the partition of T induced by X is a set of equivalence classes:

$$\mathit{\Pi}_X(T) = \{c_1, c_2, \ldots, c_m\}$$

For example, if we consider the table in Example 1, we have $\Pi_a(T) = \{\{t_1, t_3\}, \{t_2, t_4\}\}$.

Given X, $\Pi_X(T)$ is a partition or alternatively an equivalence relation. Then we have:

1. $\bigcup \Pi_X(T) = T$, for all $X \subseteq \mathcal{U}$.
2. $c_i \cap c_j = \emptyset$ for all $c_i, c_j \in \Pi_X(T)$, $i \neq j$.

The classes in a partition induced by X are disjoint and they cover all the tuples in T. The set of all partitions of a set T is $\text{Part}(T)$. We can also notice that the set of partitions of any set $\text{Part}(T)$ induces an ordering relation $\leq$:

$$\forall P_i, P_j \in \text{Part}(T) : P_i \leq P_j \iff \forall c \in P_i : \exists c' \in P_j : c \subseteq c'$$

For example: $\{\{t_1\}, \{t_2\}, \{t_3, t_4\}\} \leq \{\{t_1\}, \{t_2, t_3, t_4\}\}$. According to the partitions induced by a set of attributes, we have an alternative way of defining the necessary and sufficient conditions for a functional dependency to hold:

Proposition 1 ([12]). *A functional dependency $X \rightarrow Y$ holds in T if and only if $\Pi_Y(T) \leq \Pi_X(T)$.*

Again, taking the table in Example 1, we have that $a \rightarrow d$ holds and that $\Pi_d \leq \Pi_a$ since $\Pi_a(T) = \{\{t_1, t_3\}, \{t_2, t_4\}\}$ and $\Pi_d(T) = \{\{t_1, t_3\}, \{t_2, t_4\}\}$ (actually $d \rightarrow a$ holds too).

2.3 Purity and Approximate Dependencies

Approximate Dependencies [12]. In a table, there may be some tuples that prevent a functional dependency from holding. Those tuples can be seen as exceptions (or errors) for that dependency. Removing such tuples allows the dependency to exist: then a threshold can be set to define a set of "approximate dependencies" holding in a table. For example, a threshold of 10% means that all functional dependencies holding after removing up to 10% of the tuples of a table are valid approximate dependencies. The set of tuples to be removed for validating a functional dependency does not need to be the same for each approximate dependency. Considering in Example 2 the dependency $Month \rightarrow Av.Temp$, we can check that 6 tuples should be removed before verifying the dependency: we keep only one tuple for `Month 1` and one tuple for `Month 5` (actually just as if we remove "duplicates"). Then, if the threshold is equal to or larger than 75%, $Month \rightarrow Av.Temp$ is a valid Approximate Dependency.

Example 2. This table is an excerpt of the *Average Daily Temperature Archive* [4] from The University of Dayton, that shows the month average temperatures for different cities.

id	Month	Year	Av. Temp.	City
t_1	1	1995	36.4	Milan
t_2	1	1996	33.8	Milan
t_3	5	1996	63.1	Rome
t_4	5	1997	59.6	Rome
t_5	1	1998	41.4	Dallas
t_6	1	1999	46.8	Dallas
t_7	5	1996	84.5	Houston
t_8	5	1998	80.2	Houston

Purity Dependencies [15] are a generalization of the relationship between partitions induced by the left-hand side and right-hand side of a functional dependency. These dependencies are based on the relative impurity measure of two partitions. In order to compute this impurity measure, we need a concave and subadditive function defined on the interval $[0, 1]$ (for example, the binary entropy function). The intuition about this measure is that it computes *how much those partitions disagree*, i.e. how far two partitions π and σ are from fulfilling the relation $\pi \leq \sigma$. If the impurity measure is zero (or close to zero), then $\pi \leq \sigma$.

For example, the impurity measure (details on this measure are given in [14]) of partition $\{\{1, 2, 3\}, \{4, 5\}\}$ w.r.t. partition $\{\{1, 2\}, \{3, 4, 5\}\}$ is 5.6, whereas the impurity measure of partition $\{\{1, 3\}, \{2, 5\}, \{4\}\}$ w.r.t. partition $\{\{1, 2\}, \{3, 4, 5\}\}$ is 8.2. In the first pair of partitions, only tuple 3 is misplaced, i.e. moving 3 from one partition to another leads to the the same partitions, whereas in the second example, the number of misplaced elements is larger (2, 3, and 4 should be moved).

An important feature of this measure is that if a partition is finer than another, then, their relative impurity measure is exactly 0. This implies that a purity dependency $X \rightarrow Y$ holds if and only if the relative impurity of $\Pi_X(T)$ w.r.t. $\Pi_Y(T)$ is below a user-defined threshold. Therefore, if $\Pi_Y(T) \leq \Pi_X(T)$, a functional dependency is a valid purity dependency, regardless of the threshold.

For example, we consider all the possible dependencies having the attribute *Average Temperature* in their right-hand side. The purpose of this choice is to find out which attributes determine the values of the average temperature (Av. Temp.) in Example 2. Considering Approximate Dependencies, we introduce the two metrics *# Tuples* and *Percentage*: *# Tuples* denotes the minimal number of tuples that must be removed from the dataset for allowing the dependency to hold, and *Percentage* denotes the percentage that *# Tuples* represents for the whole dataset. For example, the Approximate Dependency $Month \rightarrow Av.Temp$ holds when we remove at least 6 (well-chosen) tuples, which represent 75% of the whole dataset.

Example 3. Dependencies with *Average Temperature* in their right-hand and the metrics related to Approximate and Purity Dependencies.

Dependency	#Tuples	Percentage	Purity
`Month -> Av. Temp`	6	75%	12.98
`Month, Year -> Av. Temp`	1	12.5%	4.0
`Month, City -> Av. Temp`	4	50%	4.0
`Year -> Av. Temp`	3	37.5%	8.26
`Year, City -> Av. Temp`	0	0%	0.0
`City -> Av. Temp`	4	50%	4.0

As for the purity measure, we use the measure of relative entropy of two partitions described in [14]. If we examine the dependency $Month \rightarrow Av.Temp$, we should the relative entropy of the partitions induced by *Month* and *Av. Temp.*, which are, respectively:

$\Pi_{Month} = \{\{t_1, t_2, t_5, t_6\}, \{t_3, t_4, t_7, t_8\}\}$

$\Pi_{Av.Temp.} = \{\{t_1\}, \{t_2\}, \{t_3\}, \{t_4\}, \{t_5\}, \{t_6\}, \{t_7\}, \{t_8\}\}$

Then, the relative entropy of Π_{Month} and $\Pi_{Av.Temp.}$ is 12.98, i.e. the largest of the conditional entropies that are computed. Actually the number of tuples that need to be reallocated for $\Pi_{Av.Temp.} \leq \Pi_{Month}$ is significantly large. It is also significant that the number of tuples that need to be removed for the dependency $Year, City \rightarrow Av.Temp$ to hold is zero and that the relative entropy of $\Pi_{Year,City}$ and $\Pi_{Av.Temp.}$ is zero as well. The Functional Dependency $Year, City \rightarrow Av.Temp$ holds because there is no pair of tuples t_i, t_j such that $t_i(Year, City) = t_j(Av.Temp.)$, i.e. there is no need to remove any tuple to verify this dependency. In addition the relative entropy of $\Pi_{Year,City}$ and $\Pi_{Av.Temp.}$ is zero, because the partitions induced by both sides, $\Pi_{Year,City}$ and $\Pi_{Av.Temp}$ are exactly the same: $\{\{t_1\}, \{t_2\}, \{t_3\}, \{t_4\}, \{t_5\}, \{t_6\}, \{t_7\}, \{t_8\}\}$. Therefore, the relation $\Pi_{Year,City} \leq \Pi_{Av.Temp}$ holds, i.e. the relative entropy is zero and this dependency trivially holds.

Yet, the intuition about this dataset is that the “Average Temperature” depends, to some extent, on the location and the month, i.e. given a city and a month, we should be able to predict the average temperature. But this intuitive relationship is somehow difficult to deduce with Approximate and Purity Dependencies. For example, the metrics for the dependency $Month, City \rightarrow Av.Temp$ indicate that 4 tuples must be removed (50% of the dataset) for checking this dependency, or alternatively, the relative entropy of the partitions $\Pi_{Month,City}$ and $\Pi_{Av.Temp}$ is 4.0. Considering the number of tuples, removing 50% of the whole dataset is a lot, especially if the intuition tells that this dependency should hold. Considering the entropy rate, the smallest entropy rate is zero and the largest computed rate is 12.98. Thus, it seems difficult to deduce the right threshold in each case.

Instead of considering measures that deal with the sets of tuples as a whole, dependencies could be directly related with the notion of “similarity”: if two tuples have similar values for the attributes *Month* and *City*, then they should have a similar value for the attribute *Av. Temp.* This can be interpreted as follows: if two cities are close enough and the corresponding months are also close enough, then the average temperature in the cities should be close enough or “similar” as well. In such a context, *“having similar values”* depends on the type of the attributes. For temperatures it mean that the absolute difference of the values is less than a given threshold. For months, it could mean that they are adjacent. For cities, it could mean that their locations are close enough.

Such a kind of dependency would provide more control and a more intuitive explanation of the relations existing between attributes.

3 Similarity Dependencies

First, we define a *tolerance* relation in a set S:

Definition 5. *$\theta \subseteq S \times S$ is a tolerance relation if:*

1. *$\forall s_i \in S : s_i \theta s_i$ (reflexivity)*
2. *$\forall s_i, s_j \in S : s_i \theta s_j \iff s_j \theta s_i$ (symmetry)*

A tolerance relation is not necessarily transitive and induces *blocks of tolerance*:

Definition 6. *Given a set S, a subset $K \subseteq S$, and a tolerance relation $\theta \subseteq S \times S$, K is a* block of tolerance *of θ if:*

1. *$\forall x, y \in K : x\theta y$ (pairwise correspondence)*
2. *$\forall z \notin K, \exists u \in K : \neg(z\theta u)$ (maximality)*

All elements in a tolerance block are in pairwise correspondence, and the block is maximal with respect to the relation θ. The set of all tolerance blocks induced by a tolerance relation θ on the set S is denoted by S/θ (by analogy with the notation of equivalence classes). S/θ is a set of maximal subsets of S and as such, $S/\theta \in \wp(\wp(S))$. Thus we have:

Property 1. $\forall K_i, K_j \in S/\theta : K_i \nsubseteq K_j$ and $K_j \nsubseteq K_i$ for all $i \neq j$

Then, we define a partial ordering on the set of all possible tolerance relations in a set S as follows:

Definition 7. *Let θ_1 and θ_2 two tolerance relations in the set S. We say that $\theta_1 \leq \theta_2$ if and only if $\forall K_i \in S/\theta_1 : \exists K_j \in S/\theta_2 : K_i \subseteq K_j$*

This relation is a partial ordering and induces a lattice where the meet and join operations of two tolerance relations θ_1 and θ_2, or, equivalently, on the sets of blocks of tolerance of θ_1 and θ_2 are:

Definition 8. *Let θ_1 and θ_2 two tolerance relations in the set S.*
$\theta_1 \wedge \theta_2 = \theta_1 \cap \theta_2 = \max_S(\{k_i \cap k_j \mid k_i \in S/\theta_1, k_j \in S/\theta_2\})$
$\theta_1 \vee \theta_2 = \theta_1 \cup \theta_2 = \max_S(S/\theta_1 \cup S/\theta_2)$

The meet $\theta_1 \wedge \theta_2$ is the set of pairwise intersections of all blocks in S/θ_1 and S/θ_2, and then removing intersections that are not maximal. The join is simpler as it consists in simply joining the blocks of tolerance in S/θ_1 and S/θ_2 and then removing the unions that are not maximal.

An example of a tolerance relation is the *similarity* that can be defined within a set of integer values as follows. Given two integer values v_1, v_2 and a threshold ϵ (user-defined): $v_1\theta v_2 \iff |v_1 - v_2| \leq \epsilon$. For example, when $S = \{1,2,3,4,5\}$ and $\epsilon = 2$, then $S/\theta = \{\{1,2,3\},\{2,3,4\},\{3,4,5\}\}$. S/θ is not a partition as transitivity does not apply.

We now come back to the set of tuples T and the set of attributes M. For each attribute $m \in M$, we define a tolerance relation on the values of that attribute: θ_m. The set of tolerance blocks induced by the tolerance relation of the attribute m is T/θ_m. All the tuples in a tolerance block $K \in T/\theta_m$ are *similar* one to the other according to their values w;r.t. the attribute m.

Example 4. Let us define a tolerance relation on an attribute $m \in \{a,b,c,d\}$ as follows: $t_i\theta_m t_j \iff |t_i(m) - t_j(m)| \leq \epsilon$.

Now, assuming that $\epsilon = 1$ in example 1, we have:

$T/\theta_a = \{\{t_1,t_3\},\{t_2,t_4\}\}, T/\theta_b = \{\{t_1,t_2,t_4\},\{t_3\}\}, T/\theta_c = \{\{t_1,t_2,t_3\},\{t_4\}\}$ and $S/\theta_d = \{\{t_1,t_3\}, \{t_2,t_4\}\}$.

We can also extend this definition to sets of attributes. Given $X \subseteq \mathcal{U}$, the similarity relation θ_X is defined as follows:

$$(t_i, t_j) \in \theta_X \iff \forall m \in X : (t_i, t_j) \in \theta_m$$

Two tuples are similar w.r.t. a set of attributes X if and only if they are similar w.r.t. *each* attributes in X. We now can define a *similarity dependency*:

Definition 9. *Let $X, Y \subseteq \mathcal{U}$: $X \rightarrow Y$ is a similarity dependency iff:* $\forall t_i, t_j \in T : t_i \theta_X t_j \Rightarrow t_i \theta_Y t_j$

In the case of a functional dependency, $X \rightarrow Y$ holds if and only if, for each pair of tuples having the *same value* w.r.t. the attributes in X, then, they have the *same value* w.r.t. the attributes in Y.

In the case of a similarity dependency, $X \rightarrow Y$ holds if and only if, for each pair of tuples having *similar values* w.r.t. the attributes in X, then, they have *similar values* w.r.t. the attributes in Y.

Example 5. We revisit the table in Example 4 and we define the tolerance relation: $t_i \theta_m t_j \iff |t_i(m) - t_j(m)| \leq 2$. Then the following similarity dependencies hold: $a \rightarrow d, ab \rightarrow d, abc \rightarrow d, ac \rightarrow d, b \rightarrow d, bc \rightarrow d, c \rightarrow d$.

It is interesting to notice that $b \rightarrow d$ is a similarity dependency but not a functional dependency, as $t_1(b) = t_2(b)$ and $t_1(d) \neq t_2(d)$. Because of the same pair of tuples, the similarity dependency $bcd \rightarrow a$ does not hold, as $t_1 \theta_{bcd} t_2$ but we do not have $t_1 \theta_a t_2$, since $|t_1(a) - t_2(a)| \nleq 2$.

By contrast, the functional dependency $bcd \rightarrow a$ holds because there is no pair of tuples t_i, t_j such that $t_i(bcd) = t_j(bcd)$.

Example 6. Going back to example 2, let us compute the Similarity Dependencies that hold and that have the attribute *Av. Temp.* in their right-hand side).

Dependency	Holds
`Month -> Av. Temp`	N
`Month, Year -> Av. Temp`	N
`Month, City -> Av. Temp`	Y
`Year -> Av. Temp`	N
`Year, City -> Av. Temp`	N
`City -> Av. Temp`	N

The only similarity dependency that holds is $Month, City \rightarrow Av.Temp$, using the following similarity measures for each attribute: $x\ \theta_{Month}\ y \iff |x - y| \leq 0$, $x\ \theta_{Year}\ y \iff |x - y| \leq 0$, $x\ \theta_{City}\ y \iff distance(x, y) \leq 500$ and $x\ \theta_{Av.Temp}\ y \iff |x - y| \leq 10$.

The similarity imposes that the month and year must be the same, whereas the distance between cities should be less than 500 kilometers and the difference between average temperatures should be less than 10 degrees (all these values are of course arbitrary).

In particular, considering the tuples t_1, t_2: $t_1 \theta_{Month,City} t_2$ since $t_1(Month) = t_2(Month) = \langle\, 1 \,\rangle$ and $t_1(City) = t_2(City) = \langle\, Milan \,\rangle$. From the other side, we have that $t_1 \theta_{Av.Temp.} t_2$ since $|36.4 - 33.8| \leq 10$.

3.1 Computing Similarity Dependencies with Pattern Structures

A pattern structure allows one to apply FCA directly on non-binary data [9]. 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 thanks to a subsumption relation $\sqsubseteq$: $\forall c, d \in D$, $c \sqsubseteq d \Longleftrightarrow c \sqcap d = c$. A pattern structure $(G, (D, \sqcap), \delta)$ is based on two derivation operators $(\cdot)^{\square}$. For $A \subseteq G$ and $d \in (D, \sqcap)$:

$$A^{\square} = \bigsqcap_{g \in A} \delta(g) \qquad\qquad d^{\square} = \{g \in G | d \sqsubseteq \delta(g)\}.$$

These operators form a Galois connection between $(\wp(G), \subseteq)$ 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 $(A_1, d_1) \leq (A_2, d_2) \Leftrightarrow A_1 \subseteq A_2 \ \ (\Leftrightarrow d_2 \sqsubseteq d_1)$, the set of all concepts forms a complete lattice called pattern concept lattice.

Thanks to the formalism of pattern structures, similarity dependencies can be characterized (and computed) in an elegant manner. Firstly, the description of an attribute $m \in M$ is given by $\delta(m) = G/\theta_m$ which is given by the set of tolerance blocks w.r.t. θ_m and $G = T$. As tolerance relations can be ordered as presented and discussed in Definitions 7 and 8, then descriptions can be ordered within a lattice.

Then, a dataset can be represented as a pattern structure $(M, (D, \sqcap), \delta)$ where M is the set of original attributes, and $(D, \sqcap)$ is the set of sets of blocks of tolerance over G provided with the meet operation defined in Definition 8.

An example of concept formation is given as follows. Starting from the set $\{a, c\} \subseteq M$ and assuming that $t_i \theta_m t_j \iff |t_i(m) - t_j(m)| \leq 2$ for all attributes:

$$\begin{aligned}
\{a, c\}^{\square} = \delta(a) \sqcap \delta(c) &= \{\{t_1, t_3\}, \{t_2, t_4\}\} \sqcap \{\{t_1, t_2, t_3\}, \{t_4\}\} \\
&= \{\{t_1, t_3\}, \{t_2\}, \{t_4\}\} \\
\{\{t_1, t_3\}, \{t_2\}, \{t_4\}\}^{\square} &= \{m \in M | \{\{t_1, t_3\}, \{t_2\}, \{t_4\}\} \sqsubseteq \delta(m)\} = \{a, c\}
\end{aligned}$$

This pattern concept lattice allows us to characterize all similarity dependencies holding in M:

Proposition 2. *A similarity dependency $X \to Y$ holds in a table T if and only if: $\{X\}^{\square} = \{XY\}^{\square}$ in the pattern structure $(M, (D, \sqcap), \delta)$.*

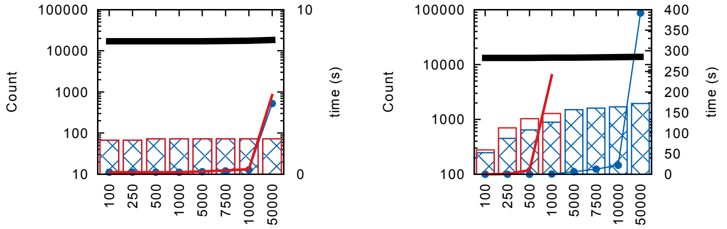
Proof. First of all, we notice that $(t, t') \in \{X\}^{\square}$ if and only if $t(X)\theta_X t'(X)$, since $(t, t') \in \{X\}^{\square}$ if and only if $\forall x \in X : t(x)\theta_x t'(x)$, if and only if $t(X)\theta_X t'(X)$. We also notice that $\{X, Y\}^{\square} \subseteq \{X\}^{\square}$.
($\Rightarrow$) We prove that if $X \to Y$ holds in T, then, $\{X\}^{\square} = \{X, Y\}^{\square}$, that is, $\{X\}^{\square} \subseteq \{X, Y\}^{\square}$. We take an arbitrary pair $(t, t') \in \{X\}^{\square}$, that is: $t(X)\theta_X t'(X)$.

Since $X \rightarrow Y$ holds, it implies that $t(XY)\theta_{XY}t'(XY)$, and this implies that $(t,t') \in \{X,Y\}^{\Box}$.
($\Leftarrow$) We take an arbitrary pair $t,t' \in T$ such that $t(X)\theta_X t'(X)$. Therefore, we have that $(t,t') \in \{X\}^{\Box}$, and by hypothesis, $(t,t') \in \{XY\}^{\Box}$, that is: $t(XY)\theta_{XY}t'(XY)$. Since this is for all pairs $t,t' \in T$ such that $t(X) = t'(X)$, this implies that $X \rightarrow Y$ holds in T.

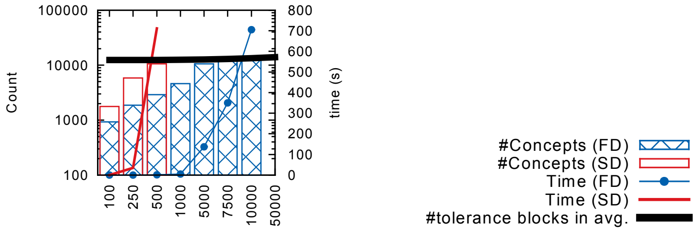
4 Experiments

Dataset description. Electronic sport denotes the extreme practice of video games where so-called cyber-athletes compete in world-wide tournaments. As for any sport, such professionals are surrounded by sponsors and practice within professional teams. These professional games are even broadcast by commentators over specialized TV channels [18]. StarCraft II (Blizzard Entertainment) is the most competitive video game. Since e-sport is a digital entertainment, one can easily find game statistics and recording in great numbers on the Web. We list more than 209,000 games between two opponents and their associated statistics (attributes). For each game, we derive two tuples (one for each of the players involved). Each player in a game (tuple) is described by 31 attributes such as his faction, the result, and several indicators of his game play.
Experimental settings. The final dataset has about 400,000 tuples described by 31 attributes with different domain types (Boolean, qualitative, and numerical). For attributes with Boolean or non-ordered qualitative domains, the similarity parameters are set to 0 as for classical FDs, since we do not have similarity constraints between their values. For the others, parameters are set by an expert of the domain, helped with the distribution of that attribute values. Thanks to the genericity of pattern structures, we slightly modified the very simplistic Java version of CloseByOne used in [1] for extracting classical FDs. The only modification lies in building the descriptions, i.e. producing tolerance blocks instead of partitions over the set of objects. We experimented on 1.8GHz and 4GB of RAM machines.
Preliminary results. We build first several different sub-datasets with randomly chosen set of tuples and different subsets of attributes. We report execution times for extracting pattern concepts (and their count) to characterize functional dependencies and similarity dependencies for three datasets in Figure 1. We also report the average number of tolerance classes of each attribute (that allows to build their description). In Figure 1 (a), the dataset is composed of qualitative and not comparable attributes only. Thus, we set the similarity parameters to 0 and observe that extracting FDs and SDs requires the same amount of time and returns the same concepts (since setting the similarity parameter to 0 leads to partitions). Naturally, the number of tolerance blocks is high, corresponding to the cardinality of attribute domains. We added more attributes, among which 5 numerical, and introduce the similarity parameters, see Figure 1 (b) and (c). The number of tolerance blocks is still high since it is an average for all attributes, and attribute domains are very dense. We notice that

there are more concepts to characterize SDs than FDs. This is due to our choice of similarity parameters. Finally, we face memory issues when computing pattern concepts for SDs, when the algorithms terminates for FDs. This is due to our pattern implementations, i.e. how a pattern is represented in the memory. We used *striped partitions*, i.e. not store any part of size 1, which can strongly reduce the pattern size in memory. For FDs, this experimentally happens more often than for SDs, due to the relaxation of the equality constraint. We need to investigate other pattern implementations.



(a) 8 attributes (b) 12 attributes



(c) 17 attributes (d) Legend

Fig. 1. Experimental results (X-axis represents the number of objects/tuples)

5 Conclusion

We discussed how Functional, Approximate and Purity Dependencies may not capture some relationships among attributes that intuitively exist in a dataset. We presented alternatively Similarity Dependencies, to express relationships between attributes based on a similarity measure that depends on the semantics of each attribute. We showed that similarity dependencies are easily characterized in FCA with pattern structures and we gave a preliminary experimental study.

Future work is in concern with a deeper investigation of the best in-memory pattern representations for fast and scalable computation, the introduction of a minimal support as well as a qualitative evaluation of similarity dependencies.

Acknowledgments. 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).

References

1. J. Baixeries, M. Kaytoue, and A. Napoli. Computing functional dependencies with pattern structures. In L. Szathmary and U. Priss, editors, *CLA*, volume 972 of *CEUR Workshop Proceedings*, pages 175–186. CEUR-WS.org, 2012.
2. M. Baudinet, J. Chomicki, and P. Wolper. Constraint-generating dependencies. *J. Comput. Syst. Sci.*, 59(1):94–115, 1999.
3. R. Belohlávek and V. Vychodil. Data tables with similarity relations: Functional dependencies, complete rules and non-redundant bases. In M.-L. Lee, K.-L. Tan, and V. Wuwongse, editors, *DASFAA*, volume 3882 of *Lecture Notes in Computer Science*, pages 644–658. Springer, 2006.
4. L. Bertossi, S. Kolahi, and L. V. S. Lakshmanan. Data cleaning and query answering with matching dependencies and matching functions. In *Proceedings of the 14th International Conference on Database Theory*, ICDT '11, pages 268–279, New York, NY, USA, 2011. ACM.
5. N. Caspard and B. Monjardet. The lattices of closure systems, closure operators, and implicational systems on a finite set: A survey. *Discrete Applied Mathematics*, 127(2):241–269, 2003.
6. A. Day. The lattice theory of fonctionnal dependencies and normal decompositions. *International Journal of Algebra and Computation*, 02(04):409–431, 1992.
7. J. Demetrovics, G. Hencsey, L. Libkin, and I. B. Muchnik. Normal form relation schemes: A new characterization. *Acta Cybern.*, 10(3):141–153, 1992.
8. W. Fan, H. Gao, X. Jia, J. Li, and S. Ma. Dynamic constraints for record matching. *The VLDB Journal*, 20(4):495–520, Aug. 2011.
9. B. Ganter and S. O. Kuznetsov. Pattern structures and their projections. In H. S. Delugach and G. Stumme, editors, *Conceptual Structures: Broadening the Base, Proceedings of the 9th International Conference on Conceptual Structures (ICCS 2001)*, LNCS 2120, pages 129–142. Springer, 2001.
10. B. Ganter and R. Wille. *Formal Concept Analysis.* Springer, Berlin, 1999.
11. G. Graetzer, B. Davey, R. Freese, B. Ganter, M. Greferath, P. Jipsen, H. Priestley, H. Rose, E. Schmidt, S. Schmidt, F. Wehrung, and R. Wille. *General Lattice Theory.* Freeman, San Francisco, CA, 1971.
12. Y. Huhtala, J. Kärkkäinen, P. Porkka, and H. Toivonen. Tane: An efficient algorithm for discovering functional and approximate dependencies. *Computer Journal*, 42(2):100–111, 1999.
13. M. Kaytoue, Z. Assaghir, A. Napoli, and S. O. Kuznetsov. Embedding tolerance relations in formal concept analysis: an application in information fusion. In J. Huang, N. Koudas, G. J. F. Jones, X. Wu, K. Collins-Thompson, and A. An, editors, *CIKM*, pages 1689–1692. ACM, 2010.
14. D. Simovici and S. Jaroszewicz. An axiomatization of partition entropy. *Information Theory, IEEE Transactions on*, 48(7):2138–2142, 2002.
15. D. A. Simovici, D. Cristofor, and L. Cristofor. Impurity measures in databases. *Acta Inf.*, 38(5):307–324, 2002.
16. S. Song and L. Chen. Efficient discovery of similarity constraints for matching dependencies. *Data & Knowledge Engineering*, (0):–, 2013. (in press).
17. S. Song, L. Chen, and P. S. Yu. Comparable dependencies over heterogeneous data. *The VLDB Journal*, 22(2):253–274, Apr. 2013.
18. T. L. Taylor. *Raising the Stakes: E-Sports and the Professionalization of Computer Gaming.* MIT Press, 2012.
19. J. Ullman. *Principles of Database Systems and Knowledge-Based Systems, volumes 1–2.* Computer Science Press, Rockville (MD), USA, 1989.

Enumerating Pseudo-Intents in a Partial Order

Alexandre Bazin and Jean-Gabriel Ganascia

Université Pierre et Marie Curie, Laboratoire d'Informatique de Paris 6
Paris, France
`Alexandre.Bazin@lip6.fr`
`Jean-Gabriel@Ganascia.name`

Abstract. The enumeration of all the pseudo-intents of a formal context is usually based on a linear order on attribute sets, the lectic order. We propose an algorithm that uses the lattice structure of the set of intents and pseudo-intents to compute the Duquenne-Guigues basis. We argue that this method allows for efficient optimizations that reduce the required number of logical closures. We then show how it can be easily modified to also compute the Luxenburger basis.

1 Introduction

Various fields, such as artificial intelligence, data mining and databases, are concerned with the problem of finding implications between sets of attributes describing objects. Formal concept analysis offers a sound mathematical framework to help develop algorithms that compute implications. As it has been shown in [3], the set of implications of minimum cardinality needed to obtain all the interesting information can be found by enumerating attribute sets known as pseudo-intents. The best-known algorithm for the computation of pseudo-intents is Next Closure [4], in which pseudo-intents are enumerated following a total order on attribute sets called lectic order. The problem of enumerating pseudo-intents in lectic order has been studied in [2] and found to be CONP-complete. The same work has been unable to find a lower bound to the complexity of this problem once the restriction on the order is removed. This indicates that the order plays an important role and that using a weaker order could potentially yield interesting results. This prompted us to study an algorithm based on the same principles as Next Closure but with a partial order on the attribute sets instead of a linear order. We are inspired by the multiple algorithms for the related problem of enumerating formal concepts that make use of the lattice structure of the problem. There already are approaches that do not make use of the lectic order for pseudo-intents, such as the attribute incremental algorithm that has been introduced in [8], or the divide-and-conquer approach in [9] but we will not compare it to our own in this work as their nature differs.

After an overview of the notations in formal concept analysis and a brief recall of Next Closure, we will present our algorithm. In a second part, we will show that it can be modified to compute not only the Duquenne-Guigues basis but also the Luxenburger basis. The functioning of the algorithm will then be illustrated by means of a small example.

2 Notations

In formal concept analysis, data is represented by a triple $\mathcal{C} = (\mathcal{O}, \mathcal{A}, \mathcal{R})$, called *formal context*, where $\mathcal{O}$ is a set of objects, $\mathcal{A}$ a set of binary attributes and $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$ a relation assigning a set of attributes to each object. An object o is said to be *described* by an attribute set A iff $(o, a) \in \mathcal{R}$ for every $a \in A$.

Two derivation operators are commonly employed and defined as follows :

$$.' : 2^{\mathcal{A}} \rightarrow 2^{\mathcal{O}}$$

$$A' = \{o \in \mathcal{O} \mid \forall a \in A, (o, a) \in \mathcal{R}\} \tag{1}$$

$$.' : 2^{\mathcal{O}} \rightarrow 2^{\mathcal{A}}$$

$$O' = \{a \in \mathcal{A} \mid \forall o \in O, (o, a) \in \mathcal{R}\} \tag{2}$$

The compositions of these two operators are closure operators and we will use the notation $.''$ for both. A set A is said to be *closed* if $A = A''$. A closed attribute set is called an *intent* and a closed object set an *extent.* A pair (E, I) where $E \subseteq \mathcal{O}$ and $I \subseteq \mathcal{A}$ is called a *formal concept* of the context if $I = E'$ and $E = I'$. A formal concept (A, B) is said *more general* than a concept (C, D) if $C \subset A$ or $B \subset D$. The set of all concepts of a given context ordered in this way is a complete lattice called *concept lattice* of the context.

An *implication* is a relation between two attribute sets A and B, denoted by $A \rightarrow B$. An implication $A \rightarrow B$ is said to *hold* in a context if every object described by A is also described by B. We write $A \equiv B$ if $A \rightarrow B$ and $B \rightarrow A$. The implication $A \rightarrow A''$ always holds in the context.

An attribute set A is a *pseudo-intent* if it is not an intent and $P'' \subset A$ for all pseudo-intents $P \subset A$. The set $\mathcal{B} = \{A \rightarrow A'' \mid A \text{ is a pseudo-intent}\}$, called the *canonical basis*, is a set of implications of minimum cardinality such that every implication that holds in the context can be inferred from it through the application of the Armstrong rules. That is, it is the minimum number of implications containing all information on the relations between attribute sets in the context.

Computing the canonical basis of a context thus requires the enumeration of its pseudo-intents. This presents some challenges as the number of pseudo-intents can be exponential in the size of the context (the number of attributes), as shown in [5]. Moreover, whether pseudo-intents can be enumerated without intents is still an open problem and the number of intents can itself be exponential in the number of pseudo-intents.

2.1 Next Closure

The most known algorithm for computing the canonical basis is Ganter's Next Closure [4]. For an arbitrary linear order $<$ on $\mathcal{A}$, a linear order, called *lectic order*, $\leq_{lec}$ on attribute sets is defined as follows:

$$A \leq_{lec} B \Leftrightarrow \min(A \Delta B) \in B$$

Δ is the symmetric difference. The implicational closure of an attribute set A with respect to a set of implications $\mathcal{L}$, usually noted $\mathcal{L}(A)$, is the smallest superset of A such that, for every $X \subseteq \mathcal{L}(A)$, we have $X \rightarrow Y \in \mathcal{L} \Rightarrow Y \subseteq \mathcal{L}(A)$. The implicational closure is a closure operator. The implicational pseudo-closure of an attribute set A, noted $\mathcal{L}^-(A)$, is the smallest superset of A such that, for every $X \subset \mathcal{L}(A)$, we have $X \rightarrow Y \in \mathcal{L} \Rightarrow Y \subseteq \mathcal{L}(A)$.

It is shown that both intents and pseudo-intents are closed under $\mathcal{B}^-$. Next Closure, in its original form, computes closed sets for a given closure operator. Applied to the computation of pseudo-intents, it goes through implicationally pseudo-closed subsets of $\mathcal{A}$ in lectic order and checks whether the result is an intent or a pseudo-intent. The lectic order being linear, every set is computed once and only once. For an attribute set A and an attribute i, the operator $\oplus$ is defined as follows :

$$A \oplus i = \mathcal{B}^-(\{a \in A \mid a \leq i\} \cup \{i\})$$

The closed set that immediately follows A in the lectic order is $A \oplus i$ with i maximal such that $\min((A \oplus i) \setminus A)$. For each set closed under $\mathcal{B}^-$, Next Closure tests whether it is an intent or a pseudo-intent and then constructs the next set with the operator $\oplus$ until it reaches $\mathcal{A}$. Numerous optimizations can be added to reduce the number of implicational closures. For example, as proposed in [8], if $Next(A) = A \oplus i = B$ and $\min(B'' \setminus A) < \max(A)$ then we can continue as if $A \oplus i$ had been rejected by Next Closure. It is the only one we will consider here as it is specific to Next Closure and has no incidence on our algorithm.

3 Algorithm

3.1 Principles

We consider the lattice $\Phi = (\{A = \mathcal{B}^-(A) \mid A \subseteq \mathcal{A}\}, \subseteq)$ of attribute sets closed under $\mathcal{B}^-$ ordered by the inclusion relation. Enumerating pseudo-intents is enumerating all the elements of Φ and testing whether they are intents or pseudo-intents. As with other enumeration problems, we want to make sure every element is found once and only once. The other great FCA problem, computing the formal concepts of a context, can be viewed as the enumeration of the elements of $(\{\mathcal{B}(A) \mid A \subseteq \mathcal{A}\}, \subseteq)$. It has been extensively studied (see [6] for a comparison of different algorithms) and various approaches have been proposed for both generating elements from previous ones and checking whether an attribute set has already been found (or even preventing the generation of redundant sets).

Surprisingly, it seems only Next Closure is commonly used for the problem of pseudo-intents, non-batch algorithms excluded. Indeed, it is very efficient in that the lectic order on attribute sets allows for the generation of the next set without stocking more than the current set and the total order ensures we do not generate them more than once. However, we feel that the fact that sets are not always generated by one of their subsets is a weakness for some applications, such as data mining. We propose to modify Next Closure with techniques from some algorithms for formal concepts that make use of the lattice structure.

The lectic order has the interesting property that a set B is greater than all its subsets. If $B \in \Phi$ is supposed to be generated by a single set, it should be one that is easily identifiable such as its lectically greatest subset $A \in \Phi$. For any two A and B in Φ, we shall use $A \rightsquigarrow B$ to denote the fact that A is the lectically greatest subset of B closed under $\mathcal{B}^-$ or, equivalently, that B is generated from A. We say that A is the predecessor of B and B is a successor of A.

Proposition 1. *The relation $\rightsquigarrow$ defines a spanning tree of the neighbouring graph of Φ.*

Proof. Any non-empty set B has at least one strict subset in Φ. We call A its lectically greatest subset in Φ. The lectic order respects the inclusion relation so there is no C such that $A \subset C \subset B$ and A is a lower cover of B. Since every non-empty $B \in \Phi$ has a single lower cover A such that $A \rightsquigarrow B$, the successor relation defines a spanning tree of the neighbouring graph of Φ. $\square$

Proposition 2. *For any two A, $B \in \Phi$, the attribute set B is an upper cover of A if and only if $B = \mathcal{B}^-(A \cup \{i\})$ for every $i \in B \setminus A$.*

Proof. If B is an upper cover of A, then there is no C such that $A \subset \mathcal{B}^-(C) \subset B$. We have $\mathcal{B}^-(A \cup \{i\}) \subseteq B$ when $i \in B \setminus A$, so $B = \mathcal{B}^-(A \cup \{i\})$ for all $i \in B \setminus A$.

If $B = \mathcal{B}^-(A \cup \{i\})$ for every $i \in B \setminus A$, given that $A \cup \{i\}$ is the smallest subset of B that contains both A and i, then there is no superset of A that is a strict subset of B. $\square$

Attribute sets are generated according to the tree, starting from $\emptyset$. The recursive nature of the definition of a pseudo-intent prevents us from computing a set before the computation of all its subsets. The lectic order solves this problem efficiently but our tree contains only the knowledge of the lectically greatest subset. In order to make sure that every subset has been found first, we propose to consider attribute sets in order of increasing cardinality. This way, when an intent or pseudo-intent of cardinality n is considered, we are sure that all pseudo-intents of cardinality $n - 1$ have been found.

Proposition 3. *If $i < \min(A)$ and $A \subset A \oplus i$, then $A \oplus i$ has a subset in Φ that is lectically greater than A.*

Proof. If $i < \min(A)$, then $\mathcal{B}^-(\{a \in A \mid a \leq i\} \cup \{i\}) = \mathcal{B}^-(\{i\})$. If $A \subset A \oplus i$, then $\{i\}$ is a pseudo-intent that is a subset of $A \oplus i$ and is lectically greater than A. $\square$

Algorithm 1 Successors(A)

```
$Successors = \emptyset$
for every attribute $i$ greater than $\min(A)$ do
    $B = A \oplus i$
    if $A \subset B$ then
        if $i = \min(B \setminus A)$ and $\forall j \in B \setminus A$, $A \oplus j = B$ then
            $Successors = Successors \cup \{B\}$
        end if
    end if
end for
return $Successors$
```

Proposition 4. *For any two attribute sets A and B such that $A \subset B$, $A \rightsquigarrow B \Leftrightarrow \forall i \in B \setminus A, A \oplus i = B$.*

Proof. If $\forall i \in B \setminus A, A \oplus i = B$, then B has no subset in Φ lectically greater than A. As such, $\forall i \in B \setminus A, A \oplus i = B \Rightarrow A \rightsquigarrow B$.

If $A \rightsquigarrow B$ and $\exists i \in B \setminus A$ such that $A \oplus i \subset B$, then B has a subset in Φ lectically greater than A, which contradicts the hypothesis. As such, $A \rightsquigarrow B \Rightarrow \forall i \in B \setminus A, A \oplus i = B$. □

So, in order to generate all the successors of A, it suffices to compute $A \oplus i$ for every i greater than $\min(A)$. If a resulting attribute set is an upper cover of A it is a successor.

Testing whether a new attribute set $B = A \oplus i$ is a successor of A is easy as we know $A \oplus j$ for every $j \in B \setminus A$. Algorithm 1 uses this to compute the successors of an attribute set.

If an attribute set B is a pseudo-intent, it is a $\wedge$-irreducible element of Φ. That is, it has a single upper cover. If $B = \mathcal{B}^-(A)$, the set B'' is the lectically next set after B if $\max(A) \leq \min(A'' \setminus A)$. That way, if an attribute set is a pseudo-intent, we do not have to explicitly compute its successors.

Algorithm 2 uses Algorithm 1 to go through the intents and pseudo-intents of the context and compute its canonical basis. It starts with $\emptyset$ and computes the closure of all the attribute sets of the same cardinality simultaneously before generating their successors. However, when an intent A is considered and its successors generated, only the pseudo-intents of cardinality lesser than $|A| + 1$ have been found so the $A \oplus i$ computed at this point might be different from the final $A \oplus i$. For example, if $|A \oplus i| - |a| = 2$, an implication $B \rightarrow C$ with $|B| = |A| + 1$ and $B \subset A \oplus i$ will change the result. For this reason, we must split Algorithm 1 into two parts. First, the sets $A \oplus i$ are computed for all i once every intent of cardinality $|A|$ have been found. Then, when $A \oplus i$ is considered as a Candidate, we must check whether it is still logically closed and, if it is, test whether it is a successor with Proposition 4.

Algorithm 2 Computing the Duquenne-Guigues Basis

```
$Card = 0$
$Candidates = \{\emptyset\}$
$\mathcal{B} = \emptyset$
while $Candidates \neq \emptyset$ do
    $Intents = \emptyset$
    for every attribute set $A$ of cardinality $Card$ in $Candidates$ do
        if $A = \mathcal{B}^{-}(A)$ and it is a successor of its generator then
            $B = A''$
            if $A \neq B$ then
                $\mathcal{B} = \mathcal{B} \cup \{A \rightarrow B\}$
                if $B$ lectically follows $A$ then
                    $Candidates = Candidates \cup \{B\}$
                end if
            else
                $Intents = Intents \cup \{A\}$
            end if
        else
            $A = \mathcal{B}^{-}(A)$
        end if
    end for
    for every set $C$ in $Intents$ do
        $Candidates = Candidates \cup \{C \oplus i \mid i \in \mathcal{A}$ and $i = \min((C \oplus i) \setminus C)\}$
    end for
    $Card = Card + 1$
end while
return $\mathcal{B}$
```

Proposition 5. *Algorithm 2 terminates and produces the canonical basis of the context.*

Proof. An attribute set can only be used to generate attribute sets of greater cardinality. The set of attributes is finite, so the algorithm terminates when it reaches $\mathcal{A}$.

Every element of Φ, except $\emptyset$, has a lower neighbour that is a predecessor in the spanning tree. If we generate the attribute sets along the tree, the lattice Φ being complete, every one of its elements is considered at least once. If an attribute set A is not equal to $\mathcal{B}^{-}(A)$ it is not a pseudo-intent so all pseudo-intents are found. □

During Algorithm 2, Algorithm 1 is applied to every intent to generate attribute sets. Their closure is then computed. As such, Algorithm 2 is in $\mathcal{O}(|\Phi| \times (X + Y))$ where X is the complexity of computing the closure of a set and $Y = |A| \times L$ is the complexity of generating successors that depends on the complexity L of the saturation algorithm used. A study of algorithms for computing the implicational closure and their use in our problem can be found in [1]. Note that, since every potential pseudo-intent P is constructed from an intent I with $I \subset P$, computing P'' can be done on the subcontext containing the objects of I'.

3.2 Improvements

Additional reduction of the number of implicational closures required to compute the base can be achieved by making use of the property that every attribute set is constructed by adding an attribute to one of its subsets.

Proposition 6. *If* $B = A \oplus i$ *is a successor of* A *with* $i = \min(B \setminus A)$*, then* $B \oplus j = A \oplus j$ *for every attribute* $j < i$*.*

Proof. If $j < i$, then $\mathcal{B}^-(\{b \in B \mid b < j\} \cup \{j\}) = \mathcal{B}^-(\{a \in A \mid a < j\} \cup \{j\})$ because $i = \min(B \setminus A)$. Thus, we have $B \oplus j = A \oplus j$. □

This lets us use the knowledge acquired on a set A to reduce the number of necessary logical closures on its supersets. Indeed, for any $B = A \oplus i$ with $i = \min(B \setminus A)$, the set $B \oplus j$ is already known for every attribute lesser than i. This means that, in order to compute the successors of B, we need only to use the $\oplus$ operator with attributes greater than i.

We can also use the fact that there are pseudo-intents P such that $P'' = \mathcal{A}$, which means that no object of the context is described by P. Indeed, for an intent A such that $A \rightsquigarrow A \oplus i$, there are two possibilities for i. If $i \in \bigcup_{o \in A'} \{o\}'$, meaning there is at least an object described by $A \cup \{i\}$, the set $A \oplus i$ is either an intent or a pseudo-intent. If $i \in \mathcal{A} \setminus \bigcup_{o \in A'} \{o\}'$, meaning no object is described by $A \cup \{i\}$, the closure of $A \oplus i$ is $\mathcal{A}$ and, for any superset B of A, the set $B \oplus i$ is either equal to $A \oplus i$ or $\mathcal{A}$. This means that computing $B \oplus i$ is unnecessary for every successors B of A in the spanning tree.

4 Computing a Base for Partial Implications

4.1 Luxenburger Basis

A partial implication between two attribute sets A and B, otherwise called association rule, is a relation of the form $A \rightarrow_{s,c} B$ where s is called the *support* and c the *confidence*. It means that $s\%$ of the objects of the context are described by A and that $c\%$ of them are also described by B. Implications, as defined in Section 2, are partial implications with a confidence of 100%. An attribute set A having the same support as A'', the confidence and support of a partial implication $A \rightarrow_{s,c} B$ can be derived from $A'' \rightarrow_{s,c} B''$. As such, to obtain a basis for partial implications, one needs only to know the intents of the formal context.

It has been shown in [7] that a minimal basis for partial implications, called Luxenburger basis, is obtained by considering a set $\{A \rightarrow_{s,c} B \mid A = A'' \text{ and } B = B''\}$ that forms a spanning tree of the neighbouring graph of the intent lattice such that $\mathcal{A}$ is the conclusion of a single partial implication.

4.2 Modification of the Algorithm

In Algorithm 2, every attribute set is generated from a single intent with the exception of some essential intents that lectically follow a pseudo-intent. We would like those intents to be constructed from their lectically greatest subset that is an intent instead of just their lectically greatest subset. Let us suppose that we have an intent A and a pseudo-intent B such that $A \rightsquigarrow B$ and $B \rightsquigarrow B''$. The lectically greatest intent that is a subset of B'' is either A or a superset of A so it can be constructed by adding attributes of $B'' \setminus A$ to A.

Algorithm 3 computes C for a given A, B and B''.

Algorithm 3 Lectically Greatest Sub-Intent

```
$C = A$
for every attribute $i$ in $B'' \setminus A$ in increasing order do
    if $\mathcal{B}(C \cup \{i\}) \neq B''$ then
        $C = C \cup \{i\}$
    end if
end for
return $C$
```

Proposition 7. *Algorithm 3 terminates and computes the lectically greatest intent that is a subset of B''*

Proof. There is a finite number of attributes and the loop goes through them in a total order so the algorithm terminates. The attribute set it returns, C, is either A or closed under $\mathcal{B}(.)$, so it is an intent. It is the implicational closure of a subset of B'' and it is not equal to B'' so we have $C \subset B''$. Let us suppose that there is an intent $D \subset B''$ lectically greater than C with $i = \min(C \Delta D)$. The attribute i is such that $\mathcal{B}(X \cup \{i\}) \subset B''$ for any $A \subseteq X \subseteq D$ so the algorithm must have added it to C and we have $C = D$.

Thus, the algorithm returns C, the greatest intent that is a subset of B''. □

If, in Algorithm 2, we maintain the spanning tree we used to generate attribute sets, it is easy to apply Algorithm 3 to every intent that lectically follows a pseudo-intent. If we change the predecessor of those intents to the attribute set computed by Algorithm 3 we obtain a spanning tree of Φ in which the predecessor of every intent is an intent. As $\mathcal{A}$ also has a unique predecessor, it gives us the Luxenburger basis of the context along with the Duquenne-Guigues basis.

5 Example

We apply Algorithm 2 on the following context with $a < b < c < d < e$:

	a	b	c	d	e
o1	×	×			
o2		×		×	×
o3		×	×	×	
o4			×		×
o5				×	×

Fig. 1. Context 1

The algorithm starts with $\emptyset$. It is closed so we generate its successors.

$$\left|\begin{array}{l} \emptyset\oplus e=\mathbf{e}\\ \emptyset\oplus d=\mathbf{d}\\ \emptyset\oplus c=\mathbf{c}\\ \emptyset\oplus b=\mathbf{b}\\ \emptyset\oplus a=\mathbf{a} \end{array}\right|$$

The set of $Candidates$ is then $\{a,b,c,d,e\}$. Among them, only a is a pseudo-intent with $a''=ab$ so we add $a\rightarrow ab$ to the basis. Moreover, ab lectically follows a so we add ab to $Candidates$. Its lectically greatest subset that is an intent is b. We then generate the successors of b, c, d and e. There are no attributes greater than $\min(e)=e$ so e has no successors.

$$\left|\begin{array}{l|l|l} b\oplus e=\mathbf{be} & c\oplus e=\mathbf{ce} & d\oplus e=\mathbf{de}\\ b\oplus d=\mathbf{bd} & c\oplus d=\mathbf{cd} & \\ b\oplus c=\mathbf{bc} & & \end{array}\right|$$

The set of $Candidates$ is then $\{ab,bc,bd,be,cd,ce,de\}$. Among them, bc, be and cd are pseudo-intents so we add $bc\rightarrow bcd$, $be\rightarrow bde$ and $cd\rightarrow bcd$ to $\mathcal{B}$. The set bcd lectically follows bc so we add bcd to $Candidates$. Its lectically greatest subset that is an intent is bd. We then generate the successors of ab, bd, ce and de. The set de has no successors for the same reasons as the set e in the previous step and we already know that $c\oplus d=cd$ so $ce\oplus d$ is known and ce has no successors.

$$\left|\begin{array}{l|l} ab\oplus e=abde & bd\oplus e=\mathbf{bde}\\ ab\oplus d=\mathbf{abd} & \\ ab\oplus c=abcd & \end{array}\right|$$

The set of $Candidates$ is then $\{abd,bcd,bde\}$. Among them, abd is a pseudo intent so we add $abd\rightarrow abcde$ to $\mathcal{B}$. The set $abcde$ lectically follows abd so we add $abcde$ to $Candidates$. Its lectically greatest subset that is an intent is ab. We then generate the successors of bcd and bde. We already know $bd\oplus c$ so we also know $bde\oplus c$. As such, computing $bde\oplus c$ is not needed and no other attribute is available so bde has no successors.

$$\left|bcd \oplus e = \mathbf{bcde}\right|$$

The set of *Candidates* is then $\{bcde, abcde\}$. Only *bcde* has a cardinality of 4 and it is a pseudo-intent so we add $bcde \rightarrow abcde$ to $\mathcal{B}$. There are no new intents so we continue with the elements of *Candidates* of cardinality 5. The set *abcde* is an intent and has no successors since it is equal to $\mathcal{A}$.

The algorithm stops having produced the following implications :

- $a \rightarrow ab$
- $bc \rightarrow bcd$
- $be \rightarrow bde$
- $cd \rightarrow bcd$
- $abd \rightarrow abcde$
- $bcde \rightarrow abcde$

This is the Duquenne-Guigues basis of the context. In addition, the following spanning tree of the intent lattice has been constructed :

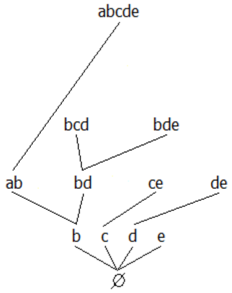


Fig. 2. Spanning Tree of the Lattice of Intents of Context 1

It corresponds to the following set of partial implications :

- $\emptyset \rightarrow_{1,0.6} b$
- $\emptyset \rightarrow_{1,0.4} c$
- $\emptyset \rightarrow_{1,0.6} d$
- $\emptyset \rightarrow_{1,0.6} e$
- $b \rightarrow_{0.6,0.33} ab$
- $b \rightarrow_{0.6,0.66} bd$
- $c \rightarrow_{0.4,0.5} ce$
- $d \rightarrow_{0.6,0.66} de$
- $ab \rightarrow_{0.2,0} abcde$
- $bd \rightarrow_{0.4,0.5} bcd$
- $bd \rightarrow_{0.4,0.6} bde$

To compute it, Algorithm 3 has been used a total of 3 times.

To compute the Duquenne-Guigues basis of this context, our algorithm has performed 16 logical closures. Since the version of Next Closure using the optimizations presented in Section 2.1 would have performed only 15, it is more efficient in this case. However, on the context presented in Figure 3, our algorithm needs only 16 logical closures while Next Closure performs 19 of them. This is due to the fact that attributes are not considered if they have been used to generate an implication $P \rightarrow \mathcal{A}$ and this context has a number of pairs of attribute that never appear together. This leads us to believe that our algorithm may be more efficient in those kinds of cases.

	a	b	c	d	e
o1	×				
o2		×			
o3	×	×			
o4					×
o5				×	
o6				×	×
o7			×		
o8			×		×
o9			×	×	
o10			×	×	×

Fig. 3. Context 2

6 Conclusion

We proposed an algorithm that computes the Duquenne-Guigues basis of a formal context. It makes use of the lattice structure of the set of both intents and pseudo-intents in a fashion similar to that of algorithms for computing the concept lattice. The construction of attribute sets is inspired by Next Closure, the most common batch algorithm for computing pseudo-intents, in that it uses the lectic order to ensure the uniqueness of generated sets while avoiding going through what has already been computed. We showed that the algorithm can easily be modified, without much loss complexity-wise, to produce both the Duquenne-Guigues and the Luxenburger bases. Those two bases together are sought after in the domain of association rules mining where it is crucial to obtain a minimal number of rules. They are usually derived from the set of frequent concepts that has to be computed first and, to the best of our knowledge, no algorithm is able to compute them both directly and at the same time without a significant increase in complexity.

Some improvements can be made on the generation of the attribute sets. Most notably, the inclusion test in Algorithm 1 could be done during the computation of the implicational closure and multiple closures could be realized simultaneously. The fact that attribute sets are effectively generated following a partial order instead of a total order could also permit some degree of parallelization. Such optimizations will be the subject of further research on our part.

References

1. Konstantin Bazhanov and Sergei A. Obiedkov. Comparing performance of algorithms for generating the Duquenne-Guigues basis. In *CLA*, pages 43–57, 2011.
2. Felix Distel and Barış Sertkaya. On the complexity of enumerating pseudo-intents. *Discrete Applied Mathematics*, 159(6):450 – 466, 2011.
3. V. Duquenne and J.-L. Guigues. Famille minimale d'implications informatives résultant d'un tableau de données binaires. *Mathématiques et Sciences Humaines*, 24(95):5–18, 1986.
4. Bernhard Ganter. Two basic algorithms in concept analysis. In *Proceedings of the 8th international conference on Formal Concept Analysis*, ICFCA'10, pages 312–340, Berlin, Heidelberg, 2010. Springer-Verlag.
5. Sergei O. Kuznetsov. On the intractability of computing the Duquenne-Guigues base. *Journal of Universal Computer Science*, 10(8):927–933, 2004.
6. Sergei O. Kuznetsov and Sergei Obiedkov. Comparing performance of algorithms for generating concept lattices. *Journal of Experimental and Theoretical Artificial Intelligence*, 14:189–216, 2002.
7. Michael Luxenburger. Implications partielles dans un contexte. *Mathématiques et Sciences Humaines*, 113:35–55, 1991.
8. 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.
9. Petko Valtchev and Vincent Duquenne. On the merge of factor canonical bases. In *Proceedings of the 6th international conference on Formal concept analysis*, ICFCA'08, pages 182–198, Berlin, Heidelberg, 2008. Springer-Verlag.

Decomposition of Intervals in the Space of Anti-Monotonic Functions

Patrick De Causmaecker, Stefan De Wannemacker

CODeS & iMinds-ITEC-KU Leuven
Department of Computerscience
KULAK, Katholieke Universiteit Leuven
`Patrick.DeCausmaecker@kuleuven-kulak.be`

Abstract. With the term 'anti-monotonic function', we designate specific boolean functions on subsets of a finite set of positive integers which we call the universe. Through the well-known bijective relationship between the set of monotonic functions and the set of anti-monotonic functions, the study of the anti-monotonic functions is equivalent to the study of monotonic functions. The true-set of an anti-monotonic function is an antichain. If the universe is denoted by N, the set of anti-monotonic functions is denoted by $AMF(N)$. This set can be partially ordered in a natural way. This paper studies enumeration in the resulting lattice of anti-monotonic functions. We define intervals of anti-monotonic functions according to this order and present four properties of such intervals, Finally we give a formula for the size of a general interval and a recursion formula for the n-th number of Dedekind.

Keywords: Dedekind numbers, anti-monotonic functions, antichains, complete distributive lattices

1 Intervals of Anti-Monotonic Functions

A Boolean valued function on the subsets of a set N, in short a Boolean function, is said to be anti-monotonic iff it is true for at most one of any two sets A and B for which $A \subsetneq B \subseteq N$ holds. A Boolean function is said to be monotonic iff the fact that it is true for a set B implies that it is true for any subset of B. There is a natural bijection between the set of monotonic functions and the set of anti-monotonic functions through the maximal true sets (e.g. [3]). Note furthermore that there is a bijection between anti-monotonic functions and antichains: the true sets of an anti-monotonic function form an antichain, i.e. a set of subsets of N such that no member set is included in another member set. In this paper we will use the antichain to denote an anti-monotonic function. We will denote the set of anti-monotonic functions on the set $N \subset \mathbb{N}$ by $AMF(N)$. We will use Greek letters to denote elements of $AMF(N)$ and Roman capitals to denote subsets of N, e.g. for $A \nsubseteq B, B \nsubseteq A$, $\alpha = \{A, B\} \in AMF(N)$. Unless otherwise stated, $N = \{1, \ldots n\}$ will be the set of the first n positive integers, and in this

© paper author(s), 2013. Published in Manuel Ojeda-Aciego, Jan Outrata (Eds.): CLA 2013, pp. 57–67, ISBN 978–2–7466–6566–8, Laboratory L3i, University of La Rochelle, 2013. Copying permitted only for private and academic purposes.

case we will occasionally use the notation $AMF(n) \equiv AMF(N)$. The size of $AMF(N)$ or $AFM(n)$ is the n-th number of Dedekind [8]. This size is known for values of n up to $n = 8$ [4]. Asymptotic expansions have been developed building on the size of the middle layer [6, 5].

Example 1. For $N = \{1, 2\}$, the Boolean function $2^N \to \mathbb{B}$, $\{\emptyset \to false, \{\emptyset\} \to false, \{1\} \to true, \{2\} \to true, \{1, 2\} \to false\}$ is anti-monotonic. The antichain of true sets is given by $\{\{1\}, \{2\}\}$. This antichain will be used to denote the anti-monotonic function.

For finite N, anti-monotonic functions form a finite distributive lattice with the join, meet and partial order given by

$$\alpha \vee \beta \;=\; max(\alpha \cup \beta) \tag{1}$$

$$\alpha \wedge \beta \;=\; max\{A \cap B | A \in \alpha, B \in \beta\} \tag{2}$$

$$\alpha \leq \beta \;\Leftrightarrow\; \forall A \in \alpha : \exists B \in \beta : A \subseteq B \tag{3}$$

$$(\Leftrightarrow \alpha \vee \beta = \beta \Leftrightarrow \alpha \wedge \beta = \alpha),$$

where for a general set S of subsets of N, $max(S)$ is the set containing only the largest sets in S according to $\subseteq$. A comprehensive textbook on Boolean functions is [2]. A recent study on counting non-equivalent monotone Boolean functions is found in [1]. Our antichains correspond to the notion of minimal sets playing an important role in the latter paper. A first analysis of intervals and decomposition is in [9]. We will make extensive use of the intervals in this lattice. For two antichains $\alpha, \beta \in AMF(N)$, the closed interval with bounds α and β is given by

$$[\alpha, \beta] = \{\chi \in AMF(N) | \alpha \leq \chi \leq \beta\}. \tag{4}$$

Analogous definitions hold for (half)open intervals. Note that these intervals are empty in case $\alpha \not\leq \beta$, in particular in case of non comparable α and β.

2 Disconnected Intervals

Given two intervals $[\rho_1, \rho_2]$ and $[\rho'_1, \rho'_2]$ in $AMF(N)$, we have

$$\{\chi \vee \chi' | \chi \in [\rho_1, \rho_2], \chi' \in [\rho'_1, \rho'_2]\} = [\rho_1 \vee \rho'_1, \rho_2 \vee \rho'_2]. \tag{5}$$

We will use the notation

$$[\rho_1, \rho_2] \vee [\rho'_1, \rho'_2] \equiv [\rho_1 \vee \rho'_1, \rho_2 \vee \rho'_2]. \tag{6}$$

A fundamental property in the decomposition of intervals is related to the concept of (dis)-connectedness. Two intervals are said to be disconnected if the decomposition in equation 6 is unique. Two intervals are connected if they are not disconnected. If two intervals are disconnected, we will call the join of these intervals *direct*:

Definition 1. *The interval* $[\rho_1 \vee \rho'_1, \rho_2 \vee \rho'_2]$ *is the direct join of two intervals* $[\rho_1, \rho_2]$ *and* $[\rho'_1, \rho'_2]$ *in* $AMF(N)$ *if the intervals are disconnected. The direct join is denoted by* $[\rho_1, \rho_2] \ovee [\rho'_1, \rho'_2]$.

Example 2. For $N = \{1,2,3\}$, we have $[\{\{1\}\}, \{\{1,3\}\}] \vee [\{\{2\}\}, \{\{2,3\}\}] = [\{\{1\},\{2\}\}, \{\{1,3\},\{2,3\}\}]$. The element $\{\{1\},\{2\},\{3\}\} = \{\{1\}\} \vee \{\{2\},\{3\}\} = \{\{1\},\{3\}\} \vee \{\{2\}\}$ shows that the two intervals on the lefthand side are connected. In the case of $[\{\{1\},\{3\}\}, \{\{1,3\}\}] \vee [\{\{2\},\{3\}\}, \{\{2,3\}\}] = [\{\{1\},\{2\},\{3\}\}, \{\{1,3\},\{2,3\}\}]$, we see that the underlying intervals are disconnected and $[\{\{1\},\{2\},\{3\}\}, \{\{1,3\},\{2,3\}\}] = [\{\{1\},\{3\}\}, \{\{1,3\}\}] \ovee [\{\{2\},\{3\}\}, \{\{2,3\}\}]$.

3 Decomposition Theorem

The decomposition of intervals is based on the following Theorem 1, which is actually valid in a general distributive lattice.

Lemma 1. *For two anti-monotonic functions* $\alpha, \beta \in AMF(N)$ *we have*

$$[\alpha \wedge \beta, \alpha \vee \beta] = [\alpha \wedge \beta, \alpha] \ovee [\alpha \wedge \beta, \beta]. \tag{7}$$

Proof. It is clear that for $\chi_\alpha \in [\alpha \wedge \beta, \alpha]$ *and* $\chi_\beta \in [\alpha \wedge \beta, \beta]$, *we have* $\chi_\alpha \vee \chi_\beta \in [\alpha \wedge \beta, \alpha \vee \beta]$. *Moreover, since* $\chi_\beta \wedge \alpha \leq \beta \wedge \alpha$, *we have* $\chi_\alpha = (\chi_\alpha \vee \chi_\beta) \wedge \alpha$ *and similarly* $\chi_\beta = (\chi_\alpha \vee \chi_\beta) \wedge \beta$. *For* $\chi \in [\alpha \wedge \beta, \alpha \vee \beta]$ *we have* $\chi \wedge \alpha \in [\alpha \wedge \beta, \alpha], \chi \wedge \beta \in [\alpha \wedge \beta, \beta]$ *and*

$$(\chi \wedge \alpha \vee \chi \wedge \beta) = \chi \wedge (\alpha \vee \beta) = \chi. \tag{8}$$

More generally we have

Theorem 1. *For three anti-monotonic functions* $\alpha, \beta, \rho \in AMF(N)$ *such that* $\rho \in [\alpha \wedge \beta, \alpha \vee \beta]$ *we have*

$$[\rho, \alpha \vee \beta] = [\rho, \alpha \vee \rho] \ovee [\rho, \beta \vee \rho]. \tag{9}$$

Proof. Note that $(\alpha \vee \rho) \wedge (\beta \vee \rho) = (\alpha \wedge \beta) \vee \rho = \rho$ *so that the interval* $[\rho, \alpha \vee \beta] = [\rho, (\alpha \vee \rho) \vee (\beta \vee \rho)]$ *satisfies the conditions of Lemma 1. Any* $\chi \in [\rho, \alpha \vee \beta]$ *has the unique decomposition* $\chi = (\chi \wedge (\alpha \vee \rho)) \vee (\chi \wedge (\beta \vee \rho))$.

Theorem 1 can be strengthened as

Theorem 2. *For three anti-monotonic functions* $\alpha, \beta, \rho \in AMF(N)$ *such that* $\rho \in [\alpha \wedge \beta, \alpha \vee \beta]$ *we have*

$$[\rho, \alpha \vee \beta] = [\rho \wedge \alpha, \alpha] \ovee [\rho \wedge \beta, \beta]. \tag{10}$$

Proof. Any $\chi \in [\rho, \alpha \vee \beta]$ *satisfies* $\chi = (\chi \wedge \alpha) \vee (\chi \wedge \beta)$, *with* $\chi \wedge \alpha \in [\rho \wedge \alpha, \alpha], \chi \wedge \beta \in [\rho \wedge \beta, \beta]$. *Any* $\chi = \chi_\alpha \vee \chi_\beta$ *with* $\chi_\alpha \in [\rho \wedge \alpha, \alpha], \chi_\beta \in [\rho \wedge \beta, \beta]$ *is in* $[\rho, \alpha \vee \beta]$. *Furthermore* $\chi \wedge \alpha = (\chi_\alpha \wedge \alpha) \vee (\chi_\beta \wedge \alpha)$ *where* $\chi_\alpha \wedge \alpha = \chi_\alpha$ *(since* $\chi_\alpha \in [\rho \wedge \alpha, \alpha]$*) and* $\chi_\beta \wedge \alpha \leq \beta \wedge \alpha$ *(since* $\chi_\beta \in [\rho \wedge \beta, \beta]$*) so that* $\chi_\beta \wedge \alpha \leq \rho \wedge \alpha \leq \chi_\alpha$, *and we conclude* $\chi_\alpha = \chi \wedge \alpha$. *Equivalently, we obtain* $\chi_\beta = \chi \wedge \beta$ *proving the uniqueness of decomposition.*

Corollary 1. *For any two anti-monotonic functions α, ρ, the intervals $[\rho, \rho \vee \alpha]$ and $[\rho \wedge \alpha, \alpha]$ are isomorphic lattices.*

Proof. Since $\rho \wedge \alpha \leq \rho$, we can apply Theorem 2 to find $[\rho, \rho \vee \alpha] = [\rho, \rho] \ovee [\rho \wedge \alpha, \alpha]$. This implies that $[\rho \wedge \alpha, \alpha] \rightarrow [\rho, \rho \vee \alpha] : \chi \rightarrow \rho \vee \chi$ defines an isomorphism with inverse $[\rho, \rho \vee \alpha] \rightarrow [\rho \wedge \alpha, \alpha] : \chi \rightarrow \alpha \wedge \chi$.

Corollary 2. *For two anti-monotonic functions $\rho_1, \rho_2 = \vee_{i \in I} \alpha_i$ with $\forall i, j \in I : \alpha_i \wedge \alpha_j \leq \rho_1$, we have*

$$\begin{aligned} [\rho_1, \rho_2] &= \ovee_{i \in I} [\rho_1, \rho_1 \vee \alpha_i] && (11) \\ &= \ovee_{i \in I} [\rho_1 \wedge \alpha_i, \alpha_i]. && (12) \end{aligned}$$

Proof. The proof follows from a simple iteration over the indices $i \in I$, applying Theorems 1 and 2 for each component α_i, $i \in I$.

In the following, we will use the notation $o_{\rho,\gamma}$ for any two anti-monotonic functions $\rho \geq \gamma$ to denote the largest χ for which $\chi \wedge \rho = \gamma$. A general partition of an interval is given by Theorem 3.

Theorem 3. *For anti-monotonic functions $\rho_1 \leq \rho \leq \rho_2$*

$$[\rho_1, \rho_2] = \cup_{\gamma \in [\rho_1, \rho]} [\gamma, o_{\rho,\gamma} \wedge \rho_2]. \tag{13}$$

The intervals $[\gamma, o_{\rho,\gamma} \wedge \rho_2]$ for $\gamma \leq \rho$ are disjoint and nonempty.

Proof. For each $\gamma \in [\rho_1, \rho]$ consider the set $S_\gamma = \{\chi \in [\rho_1, \rho_2] | \chi \wedge \rho = \gamma\}$. These sets are disjoint. Since for each $\chi \in [\rho_1, \rho_2]$, we have $\chi \wedge \rho \in [\rho_1, \rho]$, the union of these sets is the whole interval. γ is a lower bound on S_γ. Since $(\chi_1 \wedge \rho = \gamma$ and $\chi_2 \wedge \rho = \gamma) \Rightarrow (\chi_1 \vee \chi_2) \wedge \rho = \gamma$, the set has exactly one maximal element. We denote this element in the case of $\rho_2 = \{N\}$ by $o_{\rho,\gamma}$. Since, in addition, $\chi_1 \wedge \rho = \gamma, \chi_2 \wedge \rho = \gamma \Rightarrow (\chi_1 \wedge \chi_2) \wedge \rho = \gamma$, the set of all solutions to the equation in the lattice is closed under $\wedge$ and $\vee$ and hence equals the full interval $[\gamma, o_{\rho,\gamma}]$. For general ρ_2, S_γ is the intersection with $[\rho_1, \rho_2]$ which is given by $[\gamma, o_{\rho,\gamma} \wedge \rho_2]$.

The function $o_{\rho,\gamma}$ defined for any $\rho \geq \gamma \in AMF(N)$ is the top of the interval $[\gamma, o_{\rho,\gamma}] = \{\chi | \chi \wedge \rho = \gamma\}$. It is given by

$$o_{\rho,\gamma} = \widetilde{\tilde{\gamma} \backslash \tilde{\rho}} \tag{14}$$

where ˜ denotes the dual in the lattice $AMF(N)$.

Note 1. Note that the theorems so far, including the proofs, did not refer explicitly to anti-monotonic functions. In fact, they only relied on the properties of the operators $\wedge$ and $\vee$ and are seen to be valid in any complete distributive lattice. The following sections specifically refer to the definition of an anti-monotonic function as a function on subsets of a superset. Although we believe, the following properties, especially Theorem 4, can be generalized as well, we for now restrict the discussion to the space $AMF(N)$.

In what follows, disconnectedness of intervals turns out to be related to a corresponding property of the top of the interval.

Definition 2. *Given two anti-monotonic functions* $\rho, \alpha \in AMF(N)$ *with* $\rho \leq \alpha$. *Two sets* $A, B \in \alpha$ *are said to be connected with respect to* ρ *if and only if* $\{A \cap B\} \not\leq \rho$. *Connectedness of such sets is denoted by* $C_\rho(A, B)$. $C_\rho(.,.)$ *defines a graph with the sets of* α *as vertices. The vertices of each connected component of this graph correspond to a subset of* α *and thus to an anti-monotonic function. We will refer to these anti-monotonic functions as the connected components of* α *with respect to* ρ *and denote the set of such components by* $C_{\rho,\alpha}$.

We now have

Corollary 3. *For anti-monotonic functions* $\rho_1, \rho_2 \in AMF(N)$ *with* $\rho_1 \leq \rho_2$

$$[\rho_1, \rho_2] = \circledvee_{\chi \in C_{\rho_1,\rho_2}} [\rho_1 \wedge \chi, \chi]. \tag{15}$$

Proof. The proof follows immediately from Corollary 2.

Corollary 3 leads to Algorithm 1 for the total decomposition of an interval. Examples 3, 4 and 5 illustrate how the algorithm works.

Example 3. Consider the interval $[\{\emptyset\}, \{\{1,2\},\{3\}\}]$. Since we have $\{\{1,2\} \cap \{3\}\} = \{\emptyset\} \leq \{\emptyset\}$ so that $C_{\{\emptyset\},\{\{1,2\},\{3\}\}} = \{\{\{1,2\}\},\{\{3\}\}\}$, and
$[\{\emptyset\}, \{\{1,2\},\{3\}\}] = [\{\emptyset\}, \{\{1,2\}\}] \circledvee [\{\emptyset\}, \{\{3\}\}]$.

Example 4. Consider the interval $[\{\{4\}\}, \{\{1,2,4\},\{3,4\}\}]$. Since we have $\{\{1,2,4\} \cap \{3,4\}\} = \{\{4\}\} \leq \{\{4\}\}$ so that $C_{\{\{4\}\},\{\{1,2,4\},\{3,4\}\}} = \{\{\{1,2,4\}\},\{\{3,4\}\}\}$, and
$[\{\{4\}\}, \{\{1,2,4\},\{3,4\}\}] = [\{\{4\}\}, \{\{1,2,4\}\}] \circledvee [\{\{4\}\}, \{\{3,4\}\}]$.

Example 5. Consider the interval $[\{\{4\},\{6\}\}, \{\{1,2,4\},\{3,4\},\{3,5,6\}\}]$. Since we have
$\{\{1,2,4\} \cap \{3,4\}\} = \{\{4\}\} \leq \{\{4\},\{6\}\}$, $\{\{1,2,4\} \cap \{3,5,6\}\} = \{\emptyset\} \leq \{\{4\},\{6\}\}$,
$\{\{3,4\} \cap \{3,5,6\}\} = \{\{3\}\} \not\leq \{\{4\},\{6\}\}$,
so that $C_{\{\{4\},\{6\}\},\{\{1,2,4\},\{3,4\},\{3,5,6\}\}} = \{\{\{1,2,4\}\},\{\{3,4\},\{3,5,6\}\}\}$, and
$[\{\{4\},\{6\}\}, \{\{1,2,4\},\{3,4\},\{3,5,6\}\}] = [\{\{4\}\}, \{\{1,2,4\}\}] \circledvee [\{\{4\},\{6\}\}, \{\{3,4\},\{3,5,6\}\}]$.

Algorithm 1 Decompose the interval $[\rho_1, \rho_2]$

```
Require: $\rho_1 \leq \rho_2$
Ensure: Result is a set of intervals, $\{I_1, I_2, \ldots, I_k\}$, such that $[\rho_1, \rho_2] = \circledvee_{i=1\ldots k} I_i$
  function DECOMPOSEINTERVAL($\rho_1, \rho_2$)
      Compute the set $C_{\rho_1,\rho_2}$ of connected components according to Definition 2.
      return $\{[\rho_1 \wedge \gamma, \gamma] | \gamma \in C_{\rho_1,\rho_2}\}$
  end function
```

In what follows, we will use the following notations

Definition 3. *Let $\rho_1, \rho_2 \in AMF(N)$, $\rho_1 \leq \rho_2$. We use the notion of level λ_l in the interval $[\rho_1, \rho_2]$ to denote maximal anti-monotonic functions consisting of elements of a specific size l, and we introduce the $(.)^+$ and $(.)^-$ operators to transform functions from one level to a neighboring level, as follows:*

$$\lambda_l = \{A \subseteq N | \rho_1 \vee \{A\} \in]\rho_1, \rho_2], |A| = l\} (\forall l \geq 0), \tag{16}$$

$$\alpha^+ = \{X \in \lambda_{l+1} | \forall x \in X : X \backslash \{x\} \in \lambda_l \Rightarrow X \backslash \{x\} \in \alpha\}, (\forall \alpha \subseteq \lambda_l, \forall l \geq 0), \tag{17}$$

$$\alpha^- = \{X \in \lambda_{l-1} | \exists A \in \alpha : X \subseteq A\}, (\forall \alpha \subseteq \lambda_l, \forall l > 0). \tag{18}$$

Note that in Definition 3, in (17) $\alpha^+ \subseteq \lambda_{l+1}$ and in (18) $\alpha^- \subseteq \lambda_{l-1}$,

4 Decomposition of a General Interval into Computationally Easy Intervals

We will now use decomposition to compute the size of any interval. Theorem 4 builds on the following Lemma.

Lemma 2. *Let $\rho_1, \rho_2 \in AMF(N)$, $\rho_1 \leq \rho_2$ and $\chi \in [\rho_1, \rho_2]$. Then χ has the following unique decomposition:*

$$\begin{aligned} \chi = \rho_1 \vee \chi_0 \vee \chi_1 \vee \chi_2 \vee \ldots, \\ \textit{where } \forall l \geq 0 \ : \ \chi_l \subseteq \lambda_l, \\ \chi_{l+1}^- \leq \chi_l, \\ \chi_{l+1} \leq \chi_l^+. \end{aligned} \tag{19}$$

Proof. We start from the decomposition in sets of specific levels:

$$\chi = (\chi \cap \rho_1) \vee (\chi \cap \lambda_0) \vee (\chi \cap \lambda_1) \vee (\chi \cap \lambda_2) \vee \ldots (\chi \cap \lambda_{s-1}) \vee (\chi \cap \lambda_s) \tag{20}$$

where s is the size of the largest set in χ. We now set $\chi_l = \emptyset$ for $l > s$. Further, let $\chi_s = \chi \cap \lambda_s$ and note that the decomposition does not change if we add χ_s^- to the sets of level $s - 1$.

$$\chi = (\chi \cap \rho_1) \vee (\chi \cap \lambda_0) \vee (\chi \cap \lambda_1) \vee (\chi \cap \lambda_2) \vee \ldots (\chi \cap \lambda_{s-1} \vee \chi_s^-) \vee \chi_s. \tag{21}$$

This suggests the recursive definition

$$\forall l \in \{0, \ldots, s-1\} : \chi_l = (\chi \cap \lambda_l) \vee \chi_{l+1}^- \tag{22}$$

leading to

$$\chi = (\chi \cap \rho_1) \vee \chi_0 \vee \chi_1 \vee \chi_2 \vee \ldots \chi_{s-1} \vee \chi_s \tag{23}$$

and since $\chi \geq \rho_1$:

$$\chi = \rho_1 \vee \chi_0 \vee \chi_1 \vee \chi_2 \vee \ldots \chi_{s-1} \vee \chi_s. \tag{24}$$

The inequalities in (19) now follow immediately from (22) and Definition 3. Given the decomposition (19), it follows immediately that $\chi_s = \chi \cap \lambda_s$. $\chi_{s-1} \geq \chi_s^-$

implies $\chi_s^- \subseteq \chi_{s-1}$. *Furthermore, necessarily* $\chi \cap \lambda_{s-1} \subseteq \chi_{s-1}$ *so that we find* $\chi_{s-1} \geq \chi_s^- \vee (\chi \cap \lambda_{s-1})$. *Since any set in* χ_{s-1} *not dominated by a set in* χ_s *is necessarily in* χ, *we have* $\chi_{s-1} \leq \chi_s^- \vee (\chi \cap \lambda_{s-1})$ *and equality follows. Recursive application of this reasoning proves uniqueness.*

Theorem 4. *For* $\rho_1, \rho_2 \in AMF(N)$ *with* $\rho_1 \leq \rho_2$, *we have*

$$|[\rho_1, \rho_2]| = \sum_{\alpha_1 \subseteq \lambda_1} \sum_{\alpha_3 \subseteq \alpha_1^{++}} \sum_{\alpha_5 \subseteq \alpha_3^{++}} \dots 2^{|\lambda_0| - |\alpha_1^-| + |\alpha_1^+| - |\alpha_3^-| + |\alpha_3^+| - |\alpha_5^-| \dots} \tag{25}$$

$$= \sum_{\alpha_0 \subseteq \lambda_0} \sum_{\alpha_2 \subseteq \alpha_0^{++}} \sum_{\alpha_4 \subseteq \alpha_2^{++}} \dots 2^{|\alpha_0^+| - |\alpha_2^-| + |\alpha_2^+| - |\alpha_4^-| \dots}. \tag{26}$$

Proof. *Note that the number of non trivial summations in (25) and (26) is always finite: there is a maximal level for any finite interval, above this level* α^{++} *will be empty and the contribution in the power of* 2 *will be zero. Given the decomposition in Lemma 2, and a list of specific levels* $l_1 < l_2 < \dots < l_k$ *where* $\sigma_{l_i} \subseteq \lambda_{l_i}$ *are given such that*

$$\forall l_i, l_{i+1} : \sigma_{l_{i+1}}^{-d_i} \leq \sigma_{l_i}$$

$$\mathit{and}\ \forall l_{i-1}, l_i : \sigma_{l_i} \leq \sigma_{l_{i-1}}^{+d_{i-1}}$$

where $d_i = l_{i+1} - l_i$ *and* $\alpha^{+/-d} = (\dots((\alpha^{+/-})^{+/-})\dots)^{+/-}$ *(d operators* $(.)^{+/-}$*), one can ask for the set of* χ *decomposing according to (19) such that* $\forall i : \chi_{l_i} = \sigma_{l_i}$. *This set has a lower bound* $\chi_b = \rho_1 \vee \sigma_{l_0}^{-l_0} \vee \sigma_{l_0}^{-(l_0-1)} \dots \vee \sigma_{l_0} \vee \sigma_{l_1}^{-d_0} \vee \sigma_{l_1}^{-(d_0-1)} \dots$ *and an upper bound* $\chi_t = \rho_1 \vee \lambda_0 \vee \lambda_1 \dots \vee \lambda_{l_0-1} \vee \sigma_{l_0} \vee \sigma_{l_0}^{+} \vee \sigma_{l_0}^{+2} \dots \vee \sigma_{l_0}^{+(l_1-l_0-1)} \vee \sigma_{l_1} \vee \sigma_{l_1}^{+} \dots$. *In fact, all elements in the interval* $[\chi_b, \chi_t]$ *satisfy this requirement. In the case of all odd, respectively all even, levels given, summing the sizes of all such intervals over all possible specifications* σ_{2l+1}, *respectively* σ_{2l}, *results in the expansions of the Theorem.*

Theorem 4 allows to compute intervals in $AMF(N)$ for $|N| = 7$, and computes all intervals for $|N| = 6$ in milliseconds.

Example 6. As a simple application of Theorem 4, consider intervals of the form $I_N = [\{\emptyset\}, \binom{N}{2}]$ where, for convenience, we use the notation $\binom{N}{k}$ to denote the set of subsets of size k of a set $N = \{1, 2, ..., n\}$. I_N is seen to have only two nonempty levels. Indeed, $\lambda_0 = \{\}$, λ_1 is the set of all singletons of elements of N and $\lambda_2 = \binom{N}{2}$. Since $\lambda_0 = \{\}$, for each $\alpha_1 \subseteq \lambda_1$ we have $\alpha_1^- = \{\}$, while $\alpha_1^+ = \binom{span(\alpha_1)}{2}$ [1]. We find

$$|I_N| = \sum_{\alpha_1 \subseteq \lambda_1} 2^{|\lambda_0| - |\alpha_1^-| + |\alpha_1^+|} = \sum_{S \subseteq N} 2^{\binom{|S|}{2}}$$

$$= \sum_{i=0}^{n} \binom{n}{i} 2^{\binom{i}{2}}, \tag{27}$$

---

[1] Here the span of an anti-monotonic function is the set of elements occurring in true sets of the function, i.e. $span(\alpha) = \bigcup_{X \in \alpha} X$

which is the well known formula for the number of labeled graphs with at most n nodes (Sloane series A006896 [7]). This identity becomes obvious when we apply the alternative expression:

$$
\begin{aligned}
|I_N| &= \sum_{\alpha_2 \subseteq \lambda_2} 2^{|\lambda_1| - |\alpha_2^-|} && (28)\\
&= \sum_{graphs\ g\ on\ n\ vertices} 2^{n - |vertices\ in\ g|} \\
&= \sum_{i=0}^{n} |graphs\ covering\ \{1,2,...,i\}| \binom{n}{i} 2^{n-i}. && (29)
\end{aligned}
$$

5 A Recursion Formula for the Size of the Complete Space

The previous sections were concerned with the structure of arbitrary intervals. In this section, we present a formula for the size of $AMF(n+k), k \geq 0$ summing over the space $AMF(n)$. The formula is used to generate an efficient algorithm to compute the size of $AMF(n+2)$ from $AMF(n)$. We start from the following observation, using the operator × defined as [2]

$$\forall \chi \in AMF(N), S \subset \mathbb{N}, S \cap N = \emptyset : \chi \times \{S\} = \{X \cup S | X \in \chi\}. \qquad (30)$$

Example 7. For $\chi = \{\{1\}, \{2,3\}, \{3,4\}\}$ and $S = \{5,6,7\}$ we have according to this definition $\chi \times \{S\} = \{\{1,5,6,7\}, \{2,3,5,6,7\}, \{3,4,5,6,7\}\}$.

We can now prove

Lemma 3. *Given $n, k > 0$, $N = \{1, \ldots, n\}$ and $K_n = \{n+1, \ldots, n+k\}$, for each $\chi \in AMF(N \cup K)$ there exists exactly one sequence $\{\chi_{\{S\}} | S \subseteq K_n\}$ of functions in $AMF(N)$ such that*

$$
\begin{aligned}
\chi &= \bigvee_{S \subseteq K_n} \chi_S \times \{S\}, && (31)\\
\forall S \subseteq K_n &: \chi_S \in AMF(N), && (32)\\
\forall S, S' \subseteq K_n &: S \subseteq S' \Rightarrow \chi_S \geq \chi_{S'}. && (33)
\end{aligned}
$$

Proof. For each $S \subseteq K_n$ define $\chi_S = \{X \backslash K_n | X \in \chi, S \subseteq X\}$

Corollary 4. *For finite $N, K \subseteq \mathbb{N}, N \cap K = \emptyset$ as in Lemma 3, the size of $AMF(N \cup K)$ is equal to the number of homomorphisms $(2^K, \subseteq) \rightarrow (AMF(N), \geq)$.*

---

[2] This is a restricted definition of the × operator discussed extensively in [9]. This paper introduces an effective enumeration technique which can be used in to sum over the space $AMF(N)$.

Now consider the restricted homomorphisms $(2^K\backslash\{\emptyset,K\},\subseteq) \rightarrow (AMF(N),\geq)$, i.e fix χ_S for any $S \notin \{\emptyset,N\}$. Any such restricted homomorphism can be completed by components $\chi_0 \geq \bigvee_{k\in K}\chi_{\{k\}}$ and $\chi_N \leq \bigwedge_{k\in K}\chi_{N\backslash\{k\}}$. We define coefficients P_{N,K,ρ_0,ρ_N} as follows

Definition 4. *For finite $N,K \subseteq \mathbb{N}, N\cap K=\emptyset$, and for $\rho_0,\rho_N \in AMF(N), \rho_0 \geq \rho_N$, P_{N,K,ρ_0,ρ_N} is the number of homomorphisms $f:(2^K\backslash\{\emptyset,K\},\subseteq) \rightarrow ([\rho_N,\rho_0],\geq)$ such that $\bigvee_{k\in K} f(\{k\}) = \rho_0$ and $\bigwedge_{k\in K} f(N\backslash\{k\}) = \rho_N$*

and we find

Theorem 5. *For finite $N,K \subseteq \mathbb{N}, N\cap K = \emptyset$,*

$$|AMF(N\cup K)| = \sum_{\rho_0\geq\rho_N\in AMF(N)} |[\emptyset,\rho_N]|P_{N,K,\rho_0,\rho_N}|[\rho_0,\{N\}]|. \tag{34}$$

Proof. *Any restricted homomorphism f can be extended by elements of the intervals $[\emptyset,\rho_N]$ and $[\rho_0,\{N\}]$, and any extension results in a different function in $AMF(N\cup K)$.*

The P-coefficients are in general hard to compute. In the special case of $|K|=2$ however, the following property leads to a simple algorithm.

Property 1. For finite $N,K\subseteq\mathbb{N}, N\cap K=\emptyset, |K|=2$, we have

$$P_{N,K,\rho_0,\rho_N} = 2^{|C_{\rho_N\backslash\rho_0,\rho_0\backslash\rho_N}|}. \tag{35}$$

Proof. Let $K=\{k_1,k_2\}$. The coefficient is the number of solutions to the simultaneous equations

$$\chi_{\{k_1\}} \vee \chi_{\{k_2\}} = \rho_0, \tag{36}$$

$$\chi_{\{k_1\}} \wedge \chi_{\{k_2\}} = \rho_N. \tag{37}$$

Let $A,B \subseteq \rho_0\backslash\rho_N$ such that $C_{\rho_N}(A,B)$, i.e. $\{A\cap B\} \not\leq (\rho_N\backslash\rho_0)$. Then A and B must be in at least one of $\chi_{\{k_1\}}$ or $\chi_{\{k_2\}}$ due to (36) and in at most one due to (37). On the other hand, any set A in $\rho_0\backslash\rho_N$ must be in either $\chi_{\{k_1\}}$ or $\chi_{\{k_2\}}$ and can not be in both.

We thus obtain the formula

$$|AMF(n+2)| = \sum_{\rho_0\geq\rho_n\in AMF(n)} |[\emptyset,\rho_n]||[\rho_0,\{N\}]|2^{C_{\rho_n\backslash\rho_0,\rho_0\backslash\rho_n}}. \tag{38}$$

A Java implementation of Algorithm 2 , summing over non equivalent functions only for ρ_N in $AMF(6)$, allowed to compute $AMF(8)$ in 40 hours on a Macbook Pro. Note that the order of summation can be chosen such as to minimize the number of evaluations of the interval sizes $|[\rho_0,\{N\}]|$. Indeed, although the sizes of these intervals could be computed through the mapping SIZE of the representative of the class of the dual of ρ_0, these transformations still are computationally intensive.

Algorithm 2 Recursion formula using P-coefficients to compute $|AMF(n+2)|$ by enumeration of $AMF(n)$

```
Require: $n \in \mathbb{N}, n \geq 0$.
Ensure: Result is Dedekind Number $n+2$, this is the size of the space $AMF(n+2)$.
  function DedekindNumber(n+2)
      Compute the set $R$ of nonequivalent representatives in $AMF(n)$
      of equivalence classes under permutation of $N$.
      Compute the cardinalities of each of the equivalence classes as Count : $R->\mathbb{N}$.
      For each $\rho \in R$, compute the size of $[\emptyset, \rho]$ as Size : $R->\mathbb{N}$.
      $sum \leftarrow 0$
      for $\rho_0 \in AMF(n)$ do
          $partialSum \leftarrow 0$
          for $\rho_N \in R, \rho_N \leq \rho_0$ do
              $partialSum \leftarrow partialSum + \text{Count}(\rho_N) * \text{Size}(\rho_N) * 2^{|C_{\rho_N \setminus \rho_0, \rho_0 \setminus \rho_N}|}$
          end for
          $sum \leftarrow sum + partialSum * |[\rho_0, \{N\}]|$
      end for
      return $sum$.
  end function
```

6 Conclusions and Future Research

In this paper, we analyzed intervals in the space of anti-monotonic functions. Some structural properties were derived which allowed decomposition. We used the properties of intervals to derive a formula allowing efficiently computing the size of any interval in spaces with values of n up to 7. Finally, we derived an expansion of the size of the $(n+k)$th space based on an enumeration of the space n. The terms in this expansion are products of sizes of intervals multiplied by coefficients which we termed 'P-coefficients of order k'. P-coefficients of order 2 turn out to be efficiently computable, and the resulting formula, combined with a reduction to nonequivalent anti-monotonic functions, allowed computing the 8th number of Dedekind on a very standard laptop in less than two days.

The results in sections $1-3$ were obtained using the operators $\wedge$ and $\vee$ only and are thus valid for any distributive lattice. The proofs of the results in sections $4, 5$ explicitly relied on properties of sets. It is not hard to see that there are more general equivalents of these formulae. We plan to extend the analysis of intervals and derive the more general equivalents in a forthcoming paper.

The success in computing $|AMF(8)|$ on a basic laptop naturally leads to the question how far a more sophisticated hardware could take us towards computing $|AMF(9)|$. We are presently undertaking such an attempt, but new idea's will be needed to succeed. Computing $|AMF(9)|$ using P-coefficients of order 2 according to Algorithm 2 involves enumerating $AMF(7) \times$ *nonequivalent* $AMF(7)$ which is exactly $2414682040998 \times 490013148 = 1183225948328495041704$ terms. Each term would require computing a second order P-coefficient and multiplying two interval sizes of intervals in $AMF(7)$. There would be 490013148 such interval sizes to compute.

More promising is the study of the P-coefficients of higher order. Algorithm 2 for $n = 6$ equipped with a fast evaluator for order 3 P-coefficients could produce $|AMF(9)|$. Study of these higher order P-coefficients hence is on our research agenda.

References

1. Tamon Stephen and Timothy Yusun, Counting inequivalent monotone Boolean functions, arXiv preprint arXiv:1209.4623, 2012.
2. Y. Crama and P.L. Hammer, Boolean functions: Theory, algorithms, and applications, Encyclopedia of Mathematics and Its Applications, Cambridge University Press, 2011.
3. Kahn, Jeff (2002), Entropy, independent sets and antichains: a new approach to Dedekind's problem, Proc. Amer. Math. Soc. 130 (2), pp. 371-378.
4. Wiedemann, Doug (1991), A computation of the eighth Dedekind number, Order 8 (1): 56
5. Korshunov, Aleksej Dmitrievich (1981), The number of monotone boolean functions (Russian), Problemy Kibernet. 38, pp. 5-108.
6. Kleitman, Daniel and Markowsky, George (1975), On Dedekind's problem: the number of isotone Boolean functions. II, Trans. Amer. Math. Soc. 213, pp. 373-390.
7. Sloane, N. J. A., The On-Line Encyclopedia of Integer Sequences. (OEIS), *http://www.research.att.com/ njas/sequences/.*
8. Dedekind, Richard (1897), Über Zerlegungen von Zahlen durch ihre größten gemeinsamen Teiler, Gesammelte Werke, 2, pp. 103-148.
9. P. De Causmaecker, S. De Wannemacker, Partitioning in the space of anti-monotonic functions. arXiv:1103.2877 [math.NT], 2011.

Attribute exploration with fuzzy attributes and background knowledge

Cynthia Vera Glodeanu

Technische Universität Dresden,
01062 Dresden, Germany
`Cynthia-Vera.Glodeanu@tu-dresden.de`

Abstract. Attribute exploration is a formal concept analytical tool for knowledge discovery by interactive determination of the implications holding between a given set of attributes. The corresponding algorithm queries the user in an efficient way about the implications between the attributes. The result of the exploration process is a representative set of examples for the entire theory and a set of implications from which all implications that hold between the considered attributes can be deduced. The method was successfully applied in different real-life applications for discrete data. In many instances, the user may know some implications before the exploration starts. These are considered as background knowledge and their usage shortens the exploration process. In this paper we show that the handling of background information can be generalised to the fuzzy setting.

Keywords: Knowledge discovery, Formal Concept Analysis, Fuzzy data

1 Introduction

Attribute exploration [1] allows the interactive determination of the implications holding between the attributes of a given context. However, there are situations when the object set of a context is too large (possibly infinite) or difficult to enumerate. With the examples (possibly none) of our knowledge we build the object set of the context step-by-step. The *stem base* of this context, that is a minimal set of non-redundant implications from which all the implications of the context follow, is built stepwise and we are asked whether the implications of the base are true. If an implication holds, then it is added to the stem base. If however, an implication does not hold, then we have to provide a counterexample. While performing an attribute exploration we have to be able to distinguish between true and false implications and to provide correct counterexamples for false implications. This is a crucial point since the algorithm is naive and will believe whatever we tell it. Once a decision was taken about the validity of an implication, the choice cannot be reversed. Therefore, the counterexamples may not contradict the so-far confirmed implications. The procedure ends when all implications of the current stem base hold in general. This way we obtain an object set which is representative for the entire theory, that may also be infinite.

The exploration process can be shortened by taking some *background knowledge* [2] into account that the user has at the beginning of the exploration.

As many data sets contain fuzzy data, it is a natural wish to generalise attribute exploration to the fuzzy setting. In [3] we have already shown how this can be done. It turned out that we have to make some restrictions on the implications (using the globalisation as the hedge) in order to be able to perform a successful attribute exploration. In this paper we generalise the fuzzy attribute exploration to the case with background knowledge. The main work starts in Section 3 where we show how non-redundant bases can be obtained while using background knowledge. The theory for the exploration is developed in Section 4 including an appropriate algorithm. In Section 5 we use a real-world data set to illustrative both exploration with and without background knowledge.

It should be mentioned that there is some overlap with results presented in the authors PhD thesis [4], see Chapter 5.

2 Preliminaries

2.1 Crisp Attribute Exploration

We assume basic familiarities with Formal Concept Analysis [1].

In the introductory section we have already explained the principle of attribute exploration. However, we have not yet presented its key to success. This is, why we do not have to reconsider already confirmed implications after adding new objects to the context.

Proposition 1. *([1]) Let $\mathbb{K}$ be a context and $P_1, P_2, \dots, P_n$ be the first n pseudo-intents of $\mathbb{K}$ with respect to the lectic order. If $\mathbb{K}$ is extended by an object g the object intent $g^{\uparrow}$ of which respects the implications $P_i \to P_i^{\downarrow\uparrow}$, $i \in \{1, \dots, n\}$, then $P_1, P_2, \dots, P_n$ are also the lectically first n pseudo-intents of the extended context.*

Attribute exploration was successfully applied in both theoretical and practical research domains. On the one hand it facilitated the discovery of implications between properties of mathematical structures, see for example [5–7]. On the other hand it was also used in real-life scenarios, for instance in chemistry [8], information systems [9], etc.

In case the user knows some implications between the attributes in advance, *attribute exploration with background knowledge* [10, 2] can be applied. Using background knowledge in the exploration will considerably reduce the time of the process as the user has to answer less questions and provide less counterexamples.

2.2 Formal Fuzzy Concept Analysis

Due to lack of space we omit the introduction of some basic notions from fuzzy logic, fuzzy sets and Fuzzy Formal Concept Analysis. We assume that the reader is familiar with these notions and refer to standard literature. For fuzzy theory we refer to [11, 12], in particular, notions like residuated lattices with hedges and **L**-sets can be found for instance in [13]. For Fuzzy Formal Concept Analysis see [14, 15] but also [16, 17]. The theory about Fuzzy Formal Concept Analysis with hedges can be found in [13]. In this section we only present some notions concerning lectic order, attribute implications and the computation of their non-redundant bases.

We start with the *fuzzy lectic order* [18] which is defined as follows: Let $L = \{l_0 < l_1 < \cdots < l_n = 1\}$ be the support set of some linearly ordered residuated lattice and $M = \{1, 2, \ldots, m\}$ the attribute set of an **L**-context (G, M, I). For $(x, i), (y, j) \in M \times L$, we write

$$(x, l_i) \leq (y, l_j) :\Longleftrightarrow (x < y) \text{ or } (x = y \text{ and } l_i \geq l_j).$$

We define $B \oplus (x, i) := ((B \cap \{1, 2, \ldots, x-1\}) \cup \{{}^{l_i}/x\})^{\downarrow\uparrow}$ for $B \in \mathbf{L}^M$ and $(x, i) \in M \times L$. Furthermore, for $B, C \in \mathbf{L}^M$, we define $B <_{(x,i)} C$ by

$$B \cap \{1, \ldots, x-1\} = C \cap \{1, \ldots, x-1\} \text{ and } B(x) < C(x) = l_i. \qquad (1)$$

We say that B is **lectically smaller** than C, written $B < C$, if $B <_{(x,i)} C$ for some (x, i) satisfying (1). As in the crisp case, we have that $B^+ := B \oplus (x, i)$ is the least intent which is greater than a given B with respect to $<$ and (x, i) is the greatest with $B <_{(x,i)} B \oplus (x, i)$ (for details we refer to [18]).

Fuzzy Implications and Non-redundant Bases. Fuzzy attribute implications were studied in a series of papers by Bělohlávek and Vychodil [19, 20].

We denote by $\mathrm{S}(A, B)$ the truth value of "the **L**-set A is a subset of the **L**-set B". Futher, $(-)^*$ denotes the hedge of a residuated lattice **L**, i.e., $(-)^* : L \to L$ is a map satisfying $a^* \leq a$, $(a \to b)^* \leq a^* \to b^*$, $a^{**} = a^*$ and $1^* = 1$ for every $a, b \in L$. 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 $a = 1$.

A **fuzzy attribute implication** (over the attribute set M) is an expression $A \Rightarrow B$, where $A, B \in \mathbf{L}^M$. The verbal meaning of $A \Rightarrow B$ is: "if it is (very) true that an object has all attributes from A, then it also has all attributes from B". The notions "being very true", "to have an attribute", and the logical connective "if-then" are determined by the chosen **L**. For an **L**-set $N \in \mathbf{L}^M$ of attributes, the degree $||A \Rightarrow B||_N \in L$ to which $A \Rightarrow B$ is valid in N is defined as

$$||A \Rightarrow B||_N := S(A, N)^* \to S(B, N).$$

If N is the **L**-set of all attributes of an object g, then $||A \Rightarrow B||_N$ is the truth degree to which $A \Rightarrow B$ holds for g. For $\mathcal{N} \subseteq \mathbf{L}^M$, the degree $||A \Rightarrow B||_{\mathcal{N}} \in L$ to which $A \Rightarrow B$ holds in $\mathcal{N}$ is defined by $||A \Rightarrow B||_{\mathcal{N}} := \bigwedge_{N \in \mathcal{N}} ||A \Rightarrow B||_N$. For an **L**-context (G, M, I), let $I_g \in \mathbf{L}^M$ $(g \in G)$ be an **L**-set of attributes such that $I_g(m) = I(g, m)$ for each $m \in M$. Clearly, I_g corresponds to the row labelled g in (G, M, I). We define the degree $||A \Rightarrow B||_{(G,M,I)} \in L$ to which $A \Rightarrow B$ holds in (each row of) $\mathbb{K} = (G, M, I)$ by $||A \Rightarrow B||_{\mathbb{K}} := ||A \Rightarrow B||_{\mathcal{N}}$, where

$\mathcal{N} := \{I_g \mid g \in G\}$. Denote by $\mathrm{Int}(G^*, M, I)$ the set of all intents of $\mathfrak{B}(G^*, M, I)$. The degree $||A \Rightarrow B||_{\mathfrak{B}(G^*,M,I)} \in L$ to which $A \Rightarrow B$ holds in (the intents of) $\mathfrak{B}(G^*, M, I)$ is defined by

$$||A \Rightarrow B||_{\mathfrak{B}(G^*,M,I)} := ||A \Rightarrow B||_{\mathrm{Int}(G^*,M,I)}. \tag{2}$$

Lemma 1. *([20]) For each fuzzy attribute implication* $A \Rightarrow B$ *it holds that* $||A \Rightarrow B||_{(G,M,I)} = ||A \Rightarrow B||_{\mathfrak{B}(G^*,M,I)} = S(B, A^{\downarrow\uparrow})$.

Due to the large number of implications in a formal context, one is interested in the stem base of the implications. Neither the existence nor the uniqueness of the stem base for a given **L**-context are guaranteed in general [20].

Let T be a set of fuzzy attribute implications. An **L**-set of attributes $N \in \mathbf{L}^M$ is called a **model** of T if $||A \Rightarrow B||_N = 1$ for each $A \Rightarrow B \in T$. The set of all models of T is denoted by $\mathrm{Mod}(T) := \{N \in \mathbf{L}^M \mid N \text{ is a model of } T\}$. The degree $||A \Rightarrow B||_T \in L$ to which $A \Rightarrow B$ **semantically follows** from T is defined by $||A \Rightarrow B||_T := ||A \Rightarrow B||_{\mathrm{Mod}(T)}$. T is called **complete** in (G, M, I) if $||A \Rightarrow B||_T = ||A \Rightarrow B||_{(G,M,I)}$ for each $A \Rightarrow B$. If T is complete and no proper subset of T is complete, then T is called a **non-redundant basis**.

Theorem 1. *([20]) T is complete iff* $\mathrm{Mod}(T) = \mathrm{Int}(G^*, M, I)$.

As in the crisp case the stem base of a given **L**-context can be obtained through the pseudo-intents. $\mathcal{P} \subseteq \mathbf{L}^M$ is called a **system of pseudo-intents** if for each $P \in \mathbf{L}^M$ we have:

$$P \in \mathcal{P} \iff (P \neq P^{\downarrow\uparrow} \text{ and } ||Q \Rightarrow Q^{\downarrow\uparrow}||_P = 1 \text{ for each } Q \in \mathcal{P} \text{ with } Q \neq P).$$

Theorem 2. *([20])* $T := \{P \Rightarrow P^{\downarrow\uparrow} \mid P \in \mathcal{P}\}$ *is complete and non-redundant. If* $(-)^*$ *is the globalisation, then T is unique and minimal.*

3 Adding background knowledge to the stem base

The user may know some implications between attributes in advance. We will call such kind of implications *background implications.* In this section we will focus on finding a minimal list of implications, which together with the background implications will describe the structure of the concept lattice.

The theory about background knowledge for the crisp case was developed in [2] and a more general form of it in [10]. The results from [2] for implication bases with background knowledge follow by some slight modifications of the results about implication bases without background knowledge presented in [1]. The same applies for the fuzzy variant of this method. Hence, if we choose the empty set as the background knowledge, we obtain the results from [19, 20].

We start by investigating the stem bases of **L**-contexts relative to a set of background implications. Afterwards we show how some notions and results for fuzzy implications and their stem bases change for our new setting.

The attribute sets of **L**-contexts and the residuated lattices **L** will be considered finite. Further, **L** is assumed to be linearly ordered.

Definition 1. *Let $\mathbb{K}$ be a finite* $\mathbf{L}$*-context and let $\mathcal{L}$ be a set of background attribute implications. A set $\mathcal{B}$ of fuzzy attribute implications of $\mathbb{K}$ is called* $\mathcal{L}$***-complete*** *if every implication of $\mathbb{K}$ is entailed by $\mathcal{L} \cup \mathcal{B}$. We call $\mathcal{B}$,* $\mathcal{L}$***-non-redundant*** *if no implication $A \Rightarrow B$ from $\mathcal{B}$ is entailed by $(\mathcal{B} \setminus \{A \Rightarrow B\}) \cup \mathcal{L}$. If $\mathcal{B}$ is both $\mathcal{L}$-complete and $\mathcal{L}$-non-redundant, it is called an* $\mathcal{L}$***-base****.*

Note that if we have $\mathcal{L} = \varnothing$ in the above definition, then all $\mathcal{L}$-notions are actually the notions introduced for sets of fuzzy implications. This remark holds also for the other notions introduced in this section.

For any set $\mathcal{L}$ of background attribute implications and any attribute $\mathbf{L}$-set $X \in \mathbf{L}^M$ we define an $\mathbf{L}$-set $X^{\mathcal{L}} \in \mathbf{L}^M$ and an $\mathbf{L}$-set $X^{\mathcal{L}_n} \in \mathbf{L}^M$ for each non-negative integer n by

$$X^{\mathcal{L}} := X \cup \bigcup \{B \otimes \mathrm{S}(A, X)^* \mid A \Rightarrow B \in \mathcal{L}\},$$

$$X^{\mathcal{L}_n} := \begin{cases} X, & n = 0, \\ (X^{\mathcal{L}_{n-1}})^{\mathcal{L}}, & n \geq 1. \end{cases}$$

An operator $\mathcal{L}$ on these sets is defined by

$$\mathcal{L}(X) := \bigcup_{n=0}^{\infty} X^{\mathcal{L}_n}. \tag{3}$$

From [20] we know that $\mathcal{L}(-)$ is an $\mathbf{L}^*$-closure operator for a finite set M of attributes and a finite residuated lattice $\mathbf{L}$.

Definition 2. *For an* $\mathbf{L}$*-context (G, M, I), a subset $\mathcal{P} \subseteq \mathbf{L}^M$ is called a* ***system of*** $\mathcal{L}$***-pseudo-intents*** *of (G, M, I) if for each $P \in \mathbf{L}^M$ the following holds*

$$P \in \mathcal{P} \iff (P = \mathcal{L}(P) \neq P^{\downarrow\uparrow} \text{ and } ||Q \Rightarrow Q^{\downarrow\uparrow}||_P = 1 \text{ for each } Q \in \mathcal{P} : Q \neq P).$$

As in the case without background knowledge, the usage of the globalisation simplifies the definition of the system of $\mathcal{L}$-pseudo-intents. We have that $\mathcal{P} \subseteq \mathbf{L}^M$ is a system of pseudo-intents if

$$P \in \mathcal{P} \iff (P = \mathcal{L}(P) \neq P^{\downarrow\uparrow} \text{ and } Q^{\downarrow\uparrow} \subseteq P \text{ for each } Q \in \mathcal{P} \text{ with } Q \subsetneq P).$$

Theorem 3. *The set $\mathcal{B}_{\mathcal{L}} := \{P \Rightarrow P^{\downarrow\uparrow} \mid P \text{ is an } \mathcal{L}\text{-pseudo-intent}\}$ is an $\mathcal{L}$-base of $\mathbb{K}$. We call it the* $\mathcal{L}$***-Duquenne-Guigues-base*** *or the* $\mathcal{L}$***-stem base****.*

Proof. First note that all implications from $\mathcal{B}_{\mathcal{L}}$ are implications of (G, M, I). We start by showing that $\mathcal{B}_{\mathcal{L}}$ is complete, i.e., $||A \Rightarrow B||_{\mathcal{B}_{\mathcal{L}} \cup \mathcal{L}} = ||A \Rightarrow B||_{(G,M,I)}$ for every fuzzy implication $A \Rightarrow B$. By Equation (2) follows $||A \Rightarrow B||_{(G,M,I)} = ||A \Rightarrow B||_{\mathrm{Int}(G^*,M,I)}$. Hence, it is sufficient to prove that $||A \Rightarrow B||_{\mathcal{B}_{\mathcal{L}} \cup \mathcal{L}} = ||A \Rightarrow B||_{\mathrm{Int}(G^*,M,I)}$ for every fuzzy attribute implication $A \Rightarrow B$. For any $\mathbf{L}$-set $N \in \mathbf{L}^M$, $N \Rightarrow \mathcal{L}(N)$ is entailed by $\mathcal{L}$, therefore we have $N = \mathcal{L}(N)$.

Each intent $N \in \mathrm{Int}(G^*, M, I)$ is a model of $\mathcal{B}_{\mathcal{L}}$. Now let $N \in \mathrm{Mod}(\mathcal{B}_{\mathcal{L}})$ and assume that $N \neq N^{\downarrow\uparrow}$, i.e., N is not an intent. Since $N \in \mathrm{Mod}(\mathcal{B}_{\mathcal{L}})$ we have

$||Q \Rightarrow Q^{\downarrow\uparrow}||_N = 1$ for every $\mathcal{L}$-pseudo-intent $Q \in \mathcal{P}$. By definition, N is an $\mathcal{L}$-pseudo-intent and hence $N \Rightarrow N^{\downarrow\uparrow}$ belongs to $\mathcal{B}_{\mathcal{L}}$. But now, we have

$$||N \Rightarrow N^{\downarrow\uparrow}||_N = \mathrm{S}(N,N)^* \to \mathrm{S}(N^{\downarrow\uparrow},N) = 1^* \to \mathrm{S}(N^{\downarrow\uparrow},N) = \mathrm{S}(N^{\downarrow\uparrow},N) \neq 1,$$

which is a contradiction because N does not respect this implication.

To finish the proof, we still have to show that $\mathcal{B}_{\mathcal{L}}$ is $\mathcal{L}$-non-redundant. To this end let $P \Rightarrow P^{\downarrow\uparrow} \in \mathcal{B}_{\mathcal{L}}$. We show that this implication is not entailed by $\underline{\mathcal{L}} := (\mathcal{B}_{\mathcal{L}} \setminus \{P \Rightarrow P^{\downarrow\uparrow}\}) \cup \mathcal{L}$. As $P = \mathcal{L}(P)$, it is obviously a model of $\mathcal{L}$. We have $||Q \Rightarrow Q||_P = 1$ for every $\mathcal{L}$-pseudo-intent $Q \in \mathcal{P}$ different from P since P is an $\mathcal{L}$-pseudo-intent. Therefore, $P \in \mathrm{Mod}(\underline{\mathcal{L}})$. We also have that $||P \Rightarrow P^{\downarrow\uparrow}||_P = \mathrm{S}(P^{\downarrow\uparrow},P) \neq 1$ and thus P is not a model of $\mathcal{B}_{\mathcal{L}} \cup \mathcal{L}$. Hence, we have $||P \Rightarrow P^{\downarrow\uparrow}||_{(G,M,I)} = ||P \Rightarrow P^{\downarrow\uparrow}||_{\mathcal{B}_{\mathcal{L}} \cup \mathcal{L}} \neq ||P \Rightarrow P^{\downarrow\uparrow}||_{\underline{\mathcal{L}}}$, showing that $\underline{\mathcal{L}}$ is not complete and thus $\mathcal{B}_{\mathcal{L}} \cup \mathcal{L}$ is non-redundant. ☐

In general we write $P \Rightarrow P^{\downarrow\uparrow} \setminus \{m \in M \mid P(m) = P^{\downarrow\uparrow}(m)\}$ instead of $P \Rightarrow P^{\downarrow\uparrow}$.

Note that computing the stem-base and closing the implications from it with respect to the operator $\mathcal{L}(-)$ will yield a different set of implications than the $\mathcal{L}$-stem-base. Let us take a look at the following example.

Example 1. Consider the **L**-context given in Figure 1. In order to ensure that its stem-base and $\mathcal{L}$-stem-base exist, we use the globalisation. Further, we use the Gödel logic. The stem-base is displayed in the left column of Figure 1. For the background implications $\mathcal{L} := \{b \Rightarrow a,\ d \Rightarrow a,\ \{a,c\} \Rightarrow b\}$ we obtain the $\mathcal{L}$-stem-base displayed in the middle column of the figure. If we close the pseudo-intents of the stem-base with respect to the operator $\mathcal{L}(-)$, we obtain implications of the form $\mathcal{L}(P) \Rightarrow P^{\downarrow\uparrow}$, which are displayed in the right column of the figure. As one easily sees, the latter set of implications and the $\mathcal{L}$-stem-base are different. The set $\{\mathcal{L}(P) \Rightarrow P^{\downarrow\uparrow} \mid P \text{ is a pseudo-intent with } \mathcal{L}(P) \neq P^{\downarrow\uparrow}\}$ contains an additional implication, namely $\{a,b,c\} \Rightarrow d$.

	a	b	c	d
x	1	0.5	0	0
y	1	0	0	0
z	0	0	1	0
t	0	0	0	0.5

stem base	$\mathcal{L}$-stem base	$\mathcal{L}(P) \Rightarrow P^{\downarrow\uparrow}$
${}^{0.5}/b \Rightarrow a,$	${}^{0.5}/b \Rightarrow a,$	${}^{0.5}/b \Rightarrow a,$
${}^{0.5}/a \Rightarrow a,$	${}^{0.5}/a \Rightarrow a,$	${}^{0.5}/a \Rightarrow a,$
$d \Rightarrow a,b,c,$	$c, {}^{0.5}/d \Rightarrow a,b,d,$	$a,d \Rightarrow b,c,$
$c, {}^{0.5}/d \Rightarrow a,b,d,$	$a, {}^{0.5}/d \Rightarrow b,c,d,$	$c, {}^{0.5}/d \Rightarrow a,b,d,$
$b \Rightarrow a,c,d,$	$a,d \Rightarrow b,c,$	$a,b \Rightarrow c,d,$
$a, {}^{0.5}/d \Rightarrow b,c,d,$	$a,b \Rightarrow c,d.$	$a, {}^{0.5}/d \Rightarrow b,c,d,$
$a,c \Rightarrow b,d.$		$a,b,c \Rightarrow d.$

Fig. 1. An **L**-context and its different stem bases.

For developing our theory about fuzzy attribute exploration with background knowledge, the following results are useful. First, the set of all intents and all

$\mathcal{L}$-pseudo-intents is an $\mathbf{L}^*$-closure system, as stated below. Due to lack of space we omit the proofs of the following two lemmas.

Lemma 2. *Let (G, M, I) be an $\mathbf{L}$-context, let $\mathcal{L}$ be a set of fuzzy implications of (G, M, I). Further, let P and Q be intents or $\mathcal{L}$-pseudo-intents such that*

$$\mathrm{S}(P,Q)^* \le \mathrm{S}(P^{\downarrow\uparrow}, P \cap Q) \quad \textit{and} \quad \mathrm{S}(Q,P)^* \le \mathrm{S}(Q^{\downarrow\uparrow}, P \cap Q).$$

Then, $P \cap Q$ is an intent.

Note that if we choose the globalisation for $(-)^*$, then $P \cap Q$ is an intent provided that P and Q are ($\mathcal{L}$-pseudo-)intents with $P \nsubseteq Q$ and $Q \nsubseteq P$.

Now we are interested in the closure of an $\mathbf{L}$-set with respect to the implications of the $\mathcal{L}$-base $\mathcal{B}_\mathcal{L}$. Therefore, we first define for each $\mathbf{L}$-set $X \in \mathbf{L}^M$ and each non-negative integer n the $\mathbf{L}$-sets $X^{\mathcal{L}^\bullet}, X^{\mathcal{L}^\bullet_n} \in \mathbf{L}^M$ as follows:

$$X^{\mathcal{L}^\bullet} := X \cup \bigcup \{B \otimes \mathrm{S}(A,X)^* \mid A \Rightarrow B \in \mathcal{B}_\mathcal{L}, A \neq X\},$$

$$X^{\mathcal{L}^\bullet_n} := \begin{cases} X, & n = 0, \\ (X^{\mathcal{L}^\bullet_{n-1}})^{\mathcal{L}^\bullet}, & n \ge 1. \end{cases}$$

Further, we define an operator $\mathcal{L}^\bullet(-)$ on these sets by

$$\mathcal{L}^\bullet(X) := \bigcup_{n=0}^{\infty} X^{\mathcal{L}^\bullet_n}. \tag{4}$$

Lemma 3. *If $(-)^*$ is the globalisation, then $\mathcal{L}^\bullet$ given by (4) is an $\mathbf{L}^*$-closure operator and $\{\mathcal{L}^\bullet(X) \mid X \in \mathbf{L}^M\}$ coincides with the set of all $\mathcal{L}$-pseudo-intens and intents of (G, M, I).*

Remark 1. Note that for a general hedge, $\mathcal{L}^\bullet(-)$ does not need be an $\mathbf{L}^*$-closure operator. For instance, choose the Goguen structure and the identity for the hedge $(-)^*$. Further, let $\mathcal{L} := \{{}^{0.3}/y \Rightarrow y\}$. Then,

$$\begin{aligned} \mathcal{L}^\bullet(\{{}^{0.2}/y\})(y) \ge (\{{}^{0.2}/y\})^{\mathcal{L}^\bullet}(y) &= \{{}^{0.2}/y\} \cup \{y \otimes (0.3 \to 0.2)\} \\ &= \{{}^{0.2}/y\} \cup \{{}^{0.(66)}/y\} = \{{}^{0.(66)}/y\}, \end{aligned}$$

and $\mathcal{L}^\bullet(\{{}^{0.3}/y\})(y) = \{{}^{0.3}/y\}$. Hence, $\mathcal{L}^\bullet(-)$ does not satisfy the monotony property, because we have $\{{}^{0.2}/y\} \subseteq \{{}^{0.3}/y\}$ but $\mathcal{L}^\bullet(\{{}^{0.2}/y\}) \nsubseteq \mathcal{L}^\bullet(\{{}^{0.3}/y\})$.

4 Attribute exploration with background knowledge

Particularly appealing is the usage of background knowledge in the exploration process. This proves itself to be very useful and time saving for the user. He/she will have to answer less questions, as the algorithm does not start from scratch.

Due to Remark 1 and the fact that we are only able to perform a successful attribute exploration if the chosen hedge is the globalisation, we will consider only this hedge in this section. In order to arrive at the exploration with background knowledge we will present the lectic order, the "key proposition" and an appropriate algorithm for attribute exploration in this setting.

The lectic order is defined analogously as in Section 2, see (1). The only difference lies in the definition of "$\oplus$". This time we are using the $\mathbf{L}^*$-closure operator $(-)^{\mathcal{L}^\bullet}$ instead of $(-)^{\downarrow\uparrow}$.

Theorem 4. *The lectically first intent or $\mathcal{L}$-pseudo-intent is $\varnothing^{\mathcal{L}^\bullet}$. For a given $\mathbf{L}$-set $A \in \mathbf{L}^M$, the lectically next intent or $\mathcal{L}$-pseudo-intent is given by the $\mathbf{L}$-set $A \oplus (m,l)$, where $(m,l) \in M \times L$ is the greatest tuple such that $A <_{(m,l)} A \oplus (m,l)$. The lectically last intent or $\mathcal{L}$-pseudo-intent is M.*

Now we are prepared to present the main proposition regarding attribute exploration with background knowledge in a fuzzy setting.

Proposition 2. *Let $\mathbf{L}$ be a finite, linearly ordered residuated lattice with globalisation. Further, let $\mathcal{P}$ be the unique system of $\mathcal{L}$-pseudo-intents of a finite $\mathbf{L}$-context $\mathbb{K}$ with $P_1, \ldots, P_n \in \mathcal{P}$ being the first n $\mathcal{L}$-pseudo-intents in $\mathcal{P}$ with respect to the lectic order. If $\mathbb{K}$ is extended by an object g, the object intent $g^\uparrow$ of which respects the implications from $\mathcal{L} \cup \{P_i \Rightarrow P_i^{\downarrow\uparrow} \mid i \in \{1, \ldots, n\}\}$, then $P_1, \ldots, P_n$ remain the lectically first n $\mathcal{L}$-pseudo-intents of the extended context.*

Proof. Let $\mathbb{K} = (H, M, J)$ be the initial context and let (G, M, I) be the extended context, namely $G = H \cup \{g\}$ and $J = I \cap (H \times M)$. To put it briefly, since g^I is a model of $P_i \Rightarrow P_i^{JJ}$ for all $i \in \{1, \ldots, n\}$ we have that $P_i^{JJ} = P_i^{II}$. By the definition of $\mathcal{L}$-pseudo-intents and the fact that every $\mathcal{L}$-pseudo-intent Q of (H, M, J) with $Q \subset P_i$ is lectically smaller than P_i, we have that $P_1, \ldots, P_n$ are the lectically first n $\mathcal{L}$-pseudo-intents of (G, M, I).

We now have the key to a successful attribute exploration with background knowledge in the fuzzy setting, at least when we use the globalisation. With this result we are able to generalise the attribute exploration algorithm as presented by Algorithm 1. Its input is the $\mathbf{L}$-context $\mathbb{K}$, the residuated lattice $\mathbf{L}$ and the set of background implications $\mathcal{L}$. The first intent or $\mathcal{L}$-pseudo-intent is the empty set. If it is an intent, add it to the set of intents of the context. Otherwise, ask the expert whether the implication is true in general. If so, add this implication to the $\mathcal{L}$-stem base, otherwise ask for a counterexample and add it to the context (line $2-11$). Until A is different from M, repeat the following steps: Search for the largest attribute i in M with its largest value l such that $A(i) < l$. For this attribute compute its closure with respect to the $\mathcal{L}^\bullet(-)$-closure operator given by (4) and check whether the result is the lectically next intent or $\mathcal{L}$-pseudo-intent (line $12-16$). Thereby, $A \searrow i := A \cap \{1, \ldots, i-1\}$. In lines $17-25$ we repeat the same procedure as in lines $2-11$.

The algorithm generates interactively the $\mathcal{L}$-stem base of the $\mathbf{L}$-context. We enumerate the intents and pseudo-intents in the lectic order. Due to Proposition 2 we are allowed to extend the context by objects whose object intents respect the already confirmed implications. This way, the $\mathcal{L}$-pseudo-intents already contained in the stem base do not change. Hence, the algorithm is sound and correct.

Algorithm 1: FuzzyExploration($\mathbb{K}, \mathbf{L}, \mathcal{L}$)

```
$\mathcal{L} := \varnothing$; $A := \varnothing$;
if $A = A^{\downarrow\uparrow}$ then
    add $A$ to Int($\mathbb{K}$)
else
    Ask expert whether $A \Rightarrow A^{\downarrow\uparrow}$ is valid;
    if yes then
        add $A \Rightarrow A^{\downarrow\uparrow}$ to $\mathcal{L}$
    else
        Ask for counterexample $g$ and add it to $\mathbb{K}$
    end
end
while $A \neq M$ do
    for $i = n, \dots, 1$ and $l = \max L, \dots, \min L$ with $A(i) < l$ do
        $B := \mathcal{L}^{\bullet}(A)$;
        if $A \searrow i = B \searrow i$ and $A(i) < B(i)$ then
            $A := B$;
            if $A = A^{\downarrow\uparrow}$ then
                add $A$ to Int($\mathbb{K}$)
            else
                Ask expert whether $A \Rightarrow A^{\downarrow\uparrow}$ is valid;
                if yes then
                    add $A \Rightarrow A^{\downarrow\uparrow}$ to $\mathcal{L}$
                else
                    Ask for counterexample $g$ and add it to $\mathbb{K}$
                end
            end
        end
    end
end
```

5 Illustrative example

For our illustrative example we will consider the data from Figure 2. The objects are different universities from Germany and the attributes are indicators rating these institutions. **Study situation overall**: M.Sc. students were asked about their overall rating of their studies. **IT-infrastructure**: the availability of subject-specific software, PC pools and wifi were taken into account. **Courses offered**: amount of courses and the interdisciplinary references were relevant. **Possibility of studying**: timing of the courses and content of the curriculum were decisive. **Passage to M.Sc.**: clearly formulated admission requirements and assistance of the students with the organisational issues were relevant.

Suppose we want to explore the implications between the attributes from the **L**-context from Figure 2. We also know some examples, namely the TUs. These will be the objects of the **L**-context we start with. Further, we heard from others that the implications $\{a,b\} \Rightarrow c$, $d \Rightarrow \{{}^{0.5}/a, e\}$, $\{a,e\} \Rightarrow \{c,d\}$ and $a \Rightarrow \{{}^{0.5}/b, {}^{0.5}/c, {}^{0.5}/e\}$ hold. These will be considered the set of background implications $\mathcal{L}$. The exploration process is displayed in the left column of Figure 3. The first $\mathcal{L}$-pseudo-intent is $\varnothing$ and we ask the expert whether $\varnothing \Rightarrow \varnothing^{\downarrow\uparrow}$ holds. This is not the case and a counterexample is Uni Bochum. For implications that hold, for instance in step no. 3, the expert answers just "yes" and the implication is added to the $\mathcal{L}$-base. Afterwards, the validity of the implication induced by the next $\mathcal{L}$-pseudo-intent is asked, and so on. The algorithm continues until we reach M as an intent or $\mathcal{L}$-pseudo-intent. In our case, however, the algorithm stops at step no. 10. This is due to the fact that the implications induced by the $\mathcal{L}$-pseudo-intents after $\{b, {}^{0.5}/d\}$ are confirmed by the implications from the background knowledge. The result of the exploration is an extended context, namely

	study situation overall a	IT- infra-structure b	courses offered c	possibility of studying d	passage to M.Sc. e
TU Braunschweig	0.5	0	0.5	0.5	0
TU Chemnitz	0.5	1	0.5	0.5	0.5
TU Clausthal	1	1	1	1	1
TU Darmstadt	0.5	0.5	0.5	0.5	0.5
TU Ilmenau	0.5	1	0.5	0.5	1
TU Kaiserslautern	1	0.5	0.5	0.5	0
Uni Bielefeld	0.5	0	0.5	1	1
Uni Bochum	0	0.5	0.5	0.5	1
Uni Duisburg	0.5	0.5	0	0.5	0.5
Uni Erlangen	0.5	0.5	0.5	0.5	0
Uni Heidelberg	0.5	1	0.5	1	1
Uni Koblenz	0.5	0.5	0.5	0.5	0.5
Uni Saarbrücken	1	0.5	1	1	1

Fig. 2. The data is an extract from the data published in the journal Zeit Campus in January 2013. The whole data can be found under https://bit.ly/ZCinsr-informatik.

that contains our initial examples and the counterexamples we (or the expert) has entered. We also obtain an $\mathcal{L}$-base consisting of the background implications $\mathcal{L}$ and the implications we have confirmed during the exploration process. In the right column of Figure 3 the exploration without background knowledge is displayed. One immediately sees that there are 4 additional steps. The first difference between the explorations appears in step no. 8. Without using background knowledge we have to answer the implication from the right column, while this implication is already confirmed if we use background knowledge. The exploration without background knowledge yields the same extended context whereas the stem base contains the implications we have confirmed during the process.

no.	expl. w. background know.	simple expl.
1	Q: $\{\} \Rightarrow \{{}^{0.5}/a, {}^{0.5}/c, {}^{0.5}/d\}$ E: no, ex. Uni Bochum	Q: $\{\} \Rightarrow \{{}^{0.5}/a, {}^{0.5}/c, {}^{0.5}/d\}$ E: no, ex. Uni Bochum
2	Q: $\{\} \Rightarrow \{{}^{0.5}/c, {}^{0.5}/d\}$ E: no, ex. Uni Duisburg	Q: $\{\} \Rightarrow \{{}^{0.5}/c, {}^{0.5}/d\}$ E: no, ex. Uni Duisburg
3	Q: $\{\} \Rightarrow {}^{0.5}/d$ E: yes	Q: $\{\} \Rightarrow {}^{0.5}/d$ E: yes
4	Q: $\{{}^{0.5}/d, {}^{0.5}/e\} \Rightarrow {}^{0.5}/b$ E: no, ex. Uni Bielefeld	Q: $\{{}^{0.5}/d, {}^{0.5}/e\} \Rightarrow {}^{0.5}/b$ E: no, ex. Uni Bielefeld
5	Q: $\{{}^{0.5}/b, {}^{0.5}/d\} \Rightarrow {}^{0.5}/e$ E: no, ex. Uni Erlangen	Q: $\{{}^{0.5}/b, {}^{0.5}/d\} \Rightarrow {}^{0.5}/e$ E: no, ex. Uni Erlangen
6	Q: $\{{}^{0.5}/d, e\} \Rightarrow {}^{0.5}/c$ E: yes	Q: $\{{}^{0.5}/d, e\} \Rightarrow {}^{0.5}/c$ E: yes
7	Q: $\{{}^{0.5}/a, {}^{0.5}/b, {}^{0.5}/c, {}^{0.5}/d, e\} \Rightarrow b$ E: no, ex. Uni Saarbrücken	Q: $\{{}^{0.5}/a, {}^{0.5}/b, {}^{0.5}/c, {}^{0.5}/d, e\} \Rightarrow b$ E: no, ex. Uni Saarbrücken
8	Q: $\{{}^{0.5}/a, {}^{0.5}/b, {}^{0.5}/c, d, e\} \Rightarrow \{a, c\}$ E: no, ex. Uni Heidelberg	Q: $d \Rightarrow \{{}^{0.5}/a, {}^{0.5}/c, e\}$ E: yes
9	Q: $\{c, {}^{0.5}/d\} \Rightarrow \{a, {}^{0.5}/b, d, {}^{0.5}/e\}$ E: yes	Q: $\{{}^{0.5}/a, {}^{0.5}/b, {}^{0.5}/c, d, e\} \Rightarrow \{a, c\}$ E: no, ex. Uni Heidelberg
10	Q: $\{b, {}^{0.5}/d\} \Rightarrow \{{}^{0.5}/a, {}^{0.5}/c\}$ E: yes **exploration stopped.**	Q: $\{c, {}^{0.5}/d\} \Rightarrow \{a, {}^{0.5}/b, d, {}^{0.5}/e\}$ E: yes
11		Q: $\{b, {}^{0.5}/d\} \Rightarrow \{{}^{0.5}/a, {}^{0.5}/c, {}^{0.5}/e\}$ E: yes
12		Q: $\{a, {}^{0.5}/d\} \Rightarrow \{{}^{0.5}/b, {}^{0.5}/c, {}^{0.5}/e\}$ E: yes
13		Q: $\{a, {}^{0.5}/b, {}^{0.5}/c, {}^{0.5}/d, e\} \Rightarrow \{c, d\}$ E: yes
14		Q: $\{a, b^{0.5}/c, {}^{0.5}/d, {}^{0.5}/e\} \Rightarrow \{c, d, e\}$ E: yes **exploration stopped.**

Fig. 3. Exploration with and without background knowledge.

References

1. Ganter, B., Wille, R.: Formale Begriffsanalyse: Mathematische Grundlagen. Springer, Berlin, Heidelberg (1996)
2. Stumme, G.: Attribute exploration with background implications and exceptions. In Bock, H.H., Polasek, W., eds.: Data Analysis and Information Systems. Statistical and Conceptual approaches. Proc. GfKl'95. Studies in Classification, Data Analysis, and Knowledge Organization 7, Heidelberg, Springer (1996) 457–469

3. Glodeanu, C.: Attribute exploration in a fuzzy setting. In Domenach, F., Ignatov, D., Poelmans, J., eds.: Contributions to International Conference on Formal Concept Analysis, Katholieke Universiteit Leuven (2012)
4. Glodeanu, C.: Conceptual Factors and Fuzzy Data. PhD thesis, TU Dresden (2012)
5. Sacarea, C.: Towards a theory of contextual topology. PhD thesis, TH Darmstadt, Aachen (2001)
6. Kwuida, L., Pech, C., Reppe, H.: Generalizations of Boolean algebras. an attribute exploration. Math. Slovaca **56**(2) (2006) 145–165
7. Revenko, A., Kuznetsov, S.: Attribute exploration of properties of functions on ordered sets. In: Proc. CLA 2010. (2010) 313–324
8. Bartel, H.G., Nofz, M.: Exploration of nmr data of glasses by means of formal concept analysis. Chemom. Intell. Lab. Syst. **36** (1997) 53–63
9. Stumme, G.: Acquiring expert knowledge for the design of conceptual information systems. In Fensel, D., Studer, R., eds.: EKAW. Volume 1621 of Lecture Notes in Computer Science., Springer (1999) 275–290
10. Ganter, B.: Attribute exploration with background knowledge. Theoretical Computer Science **217**(2) (1999) 215–233
11. Hájek, P.: The Metamathematics of Fuzzy Logic. Kluwer (1998)
12. Klir, G., Yuan, B.: Fuzzy sets and fuzzy logic - theory and applications. Prentice Hall P T R, Upper Saddle River, New Jersey 07458 (1995)
13. Belohlávek, R., Vychodil, V.: Fuzzy concept lattices constrained by hedges. JACIII **11**(6) (2007) 536–545
14. Umbreit, S.: Formale Begriffsanalyse mit unscharfen Begriffen. PhD thesis, Martin-Luther-Universitaet Halle-Wittenberg (1994)
15. Burusco, A., Fuentes-Gonzales: The study of L-Fuzzy Concept Lattice. Mathware and soft computing **1**(3) (1994) 209–218
16. Pollandt, S.: Fuzzy Begriffe. Springer Verlag, Berlin Heidelberg New York (1997)
17. Bělohlávek, R.: Fuzzy Relational Systems: Foundations and Principles. Systems Science and Engineering. Kluwer Academic/Plenum Press (2002)
18. Bělohlávek, R.: Algorithms for fuzzy concept lattices. In: International Conference on Recent Advances in Soft Computing. (2002) 200–205
19. Bělohlávek, R., Vychodil, V.: Attribute implications in a fuzzy setting. In Missaoui, R., Schmid, J., eds.: International Conference on Formal Concept Analysis. Lecture Notes in Computer Science, Springer (2006) 45–60
20. Bělohlávek, R., Vychodil, V.: Fuzzy attribute logic: attribute implications, their validity, entailment, and non-redundant basis. In Liu, Y., Chen, G., Ying, M., eds.: Eleventh International Fuzzy Systems Association World Congress,. Volume 1 of Fuzzy Logic, Soft Computing & Computational Intelligence., Tsinghua University Press and Springer (2005) 622–627

An efficient Java implementation of the immediate successors calculation

Clément Guérin, Karell Bertet, and Arnaud Revel

L3i, Laboratory in Computer Science, Image and Interaction,
University of La Rochelle
{clement.guerin,karell.bertet,arnaud.revel}@univ-lr.fr

Abstract. The authors present in this paper an effective Java implementation of the concept immediate successors calculation. It is based on the lattice Java library, developed by K. Bertet and the Limited Objects Access algorithm, proposed by C. Demko and K. Bertet [6] with Java-specific enhancements. This work was motivated by the need of an efficient tool delivering this service in an accessible and popular programming language for a wider research project: eBDtheque. Performances are compared and analyzed.

Keywords: concept lattice, Java implementation, immediate successors, Limited Object Access algorithm, comicbooks

1 Introduction

A Galois lattice, or concept lattice, is a graph providing a representation of all the possible associations between a set of objects, or observations, and their describing attributes. Lattices can be queried, browsed and therefore provide a smart way to represent and retrieve information through a description context. Although they have been introduced a long time ago [1], they were hardly considered helpful for information retrieval before the end of the twentieth century and the work of [9, 3, 18] due to an exponential computational time issue. Even today, using concept lattices to handle a large set of objects described by an even wider set of attributes can be challenging. Indeed, it is still not trivial to use lattice's properties in a big development project, mainly because of the lack of available software frameworks. Being in need of such a tool to browse and query a dataset on comic books, described by an ontology [10], we propose an efficient Java implementation for the calculation of the immediate successors of a concept (i.e. to get the closest refined concepts according to the description of the starting point).

This paper is organized as follows. After introducing our motivations in part 2, the third section reminds how are calculated the immediate successors in the state of the art. The next section details the implemented algorithm and how it has been done in order to be as effective as possible. Section 5 shows some experimentations related to the classical immediate successors algorithm. Finally the last section concludes this paper and brings up some perspectives on our ongoing work.

2 Context and motivation

We are developing a multi-faceted solution to automatically analyze comic books in a) an image processing point of view [19], b) a semantic point of view [10]. One key point is the possibility for a user to retrieve comic books panels, the rectangular frames in which the pictures are drawn, similar to an input query, based on the semantic description of each panel from the dataset [11]. Two panels may share the same kind of content, such as speech balloons, spoken words, objects, characters or even some localized pixel's visual features (e.g colors, textures, shapes, etc.). They can be of the same shape, at the same location in the page or in the narration, drawn by the same person or come from the same volume (Table 1 and Fig. 1 show an example of what could be such shared properties). Possibilities are only limited by the ability of our system to recognize and deduce information. All these heterogeneous pieces of description have to be expressed in a way that can be interrogated and browsed efficiently.

	panel 1	panel 2	panel 3	panel 4	panel 5
contains:balloon	1	1	1	1	0
shape:wide	0	1	0	0	1
shape:high	1	0	0	1	0
size:medium	0	0	1	0	1
contains:tree	0	0	0	1	1

Table 1. Sample context

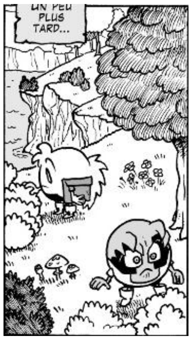


Fig. 1. Panel 4 from Table 1. Credit [5]

One way to browse data is to guide the user by a relevance feedback mechanism as it can classically be seen in content based image retrieval (CBIR) [20] and machine learning [21]. To an input query, which can be a panel or a set of features, follows a loop of interactions between the computer and the final user that will guide him, hopefully sooner than later, to his answer. At each step, the system returns a set of result panels that share the same attributes, weighted by the estimated relevance of these attributes to the query. The user is then invited to label the panels based on what he considers to be important in his query. The relevant (resp. irrelevant) features are identified with the right (resp. wrong) labeled results, their weight is dynamically adjusted, so the query can be refined to produce a more accurate output. The interaction loop between the user and the system goes on and on until the user is satisfied.

The structure of concept lattices seems to fit particularly well to the task as each concept is made of a set of panels described by a shared set of attributes. The output can be composed of panels from several distinct concepts, chosen for the weight of their attributes. The user is then guided through the lattice structure by his choices without being aware of the underlying mechanism.

As we were working with the Java language, we started looking for a way to handle lattices that could easily be integrated to the main solution. The set of observations is meant to be quite large as it does not become very interesting to retrieve data until the volume gets critical. Because of the extreme heterogeneity of comic books' panels, the growth of the observation set (the pictures) automatically implies the growth of the attribute set (the description of those pictures). Therefore, the implementation has to be efficient when dealing with large sets of both attributes and observations.

3 State of the art

More formally, a concept lattice is defined from a binary table, also denoted a formal context, $(O, I, (\alpha, \beta))$ where O is the set of objects, I the set of attributes, $\alpha(A)$ the set of attributes shared by a subset A of objects, and $\beta(B)$ the set of objects sharing a subset B of attributes. Each node of a concept lattice is denoted a concept (A, B), i.e. a maximal objects-attributes correspondence, verifying $\alpha(A) = B$ and $\beta(B) = A$. A is called the extent of the concept, and B its intent. Two formal concepts (A_1, B_1) and (A_2, B_2) are in relation in the lattice when they verify the following extension-generalization property:

$$(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow \left\| \begin{array}{l} A_1 \subseteq A_2 \\ \text{(equivalent to } B_1 \supseteq B_2\text{)} \end{array} \right.$$

The whole set of formal concepts fitted out by the order relation $\leq$ is called a concept lattice or a Galois lattice because it verifies the lattice properties, and the cover relation $\prec$ corresponds to the Hasse diagram of the lattice. The concepts $\bot = (O, \alpha(O))$ and $\top = (\beta(I), I)$ respectively correspond to the bottom and the top of the concept lattice. See the book of Ganter and Wille [8] for a more complete description of formal concept lattices.

Numerous generation algorithms have been proposed in the literature [16, 7, 2, 17]. All of these proposed algorithms have a polynomial complexity with respect to the number of concepts (at best quadratic in [17]). The complexity is therefore determined by the size of the lattice (i.e. the number of concepts in the lattice), this size being exactly bounded by $2^{|O+I|}$ in the worst case (when the table context is a diagonal matrix of zeros) and by $|O+I|$ in the best case (diagonal matrix of ones). A formal and experimental comparative study of the different algorithms has been published in [13]. Although all these algorithms generate the same lattice, they propose different strategies. Ganter's NextClosure [7] is the reference algorithm that determines the concepts in lectical order (next, the concepts may be ordered by $\leq$ or $\prec$ to form the concept lattice) while Bordat's algorithm [2] is the first algorithm that computes directly the Hasse diagram of the lattice, by computing immediate successors for each concept, starting from the bottom concept, until all concepts are generated. Immediate successor calculation is appropriate for an on-demand generation inside the structure, useful for a navigation without generating the whole set of concepts.

Bordat's algorithm, independently rediscovered by [14], is issued from a corollary of Bordat's theorem [2] stating that (A', B') is an immediate successor of a concept (A, B) if and only if A' is inclusion-maximal in the following set system $\mathcal{F}_A$ defined on the objects set O by:

$$\mathcal{F}_A = \{\beta(x) \cap A \ : \ x \in I \backslash B\} \tag{1}$$

Immediate successors algorithm first generates the set system $\mathcal{F}_A$ in a linear time ; then inclusion-maximal subsets of $\mathcal{F}_A$ can easily be computed in $\mathcal{O}(|O|^3)$, using an inclusion graph for example. Notice that the inclusion-maximal subsets problem is known to be resolved in $\mathcal{O}(|O|^{2,5})$ using sophisticated data structures ([15, 12]).

It is possible to consider the restriction of a concept lattice to the attributes set. Indeed, a nice result establishes that any concept lattice is isomorphic to the closure lattice defined on the set I of attributes, where each concept is restricted to its attributes part. The closure lattice is composed of all closures - i.e. fixed points - for the closure operator $\varphi = \alpha \circ \beta$. Using the closure lattice instead of the whole concept lattice gives rise to a storage improvement, for example in case of large amount of objects. See the survey of Caspard and Monjardet [4] for more details about closure lattices.

Closure lattices can be generated by an extension of Bordat's algorithm (see Alg. 1) issued from a reformulation of Bordat's Theorem. Indeed, each immediate successor B' of a closure B is obtained by $B' = \alpha(A')$ with $A' \in \mathcal{F}_A$. Since $A' = \beta(x) \cap A = \beta(x) \cap \beta(B) = \beta(x+B)$, thus $B' = \alpha(A') = \alpha(\beta(x+B)) = \varphi(x+B)$. Therefore, immediate successors of a closure B are inclusion-minimal subsets in a set system $\mathcal{F}_B$ defined on the attributes set I by:

$$\mathcal{F}_B = \{\varphi(x + B) \ : \ x \in I \backslash B\} \tag{2}$$

The Limited Object Access algorithm (see Algorithm 2), introduced by Demko and Bertet in 2011, is another extension of Bordat's immediate successors gen-

```
Name: Immediates_Successors
Data: A context $K = (O, I, (\alpha, \beta))$ ; A closure $B$ of the closure lattice of $K$
Result: The immediate successors of $B$ in the closure lattice
begin
    Init the set system $\mathcal{F}_B$ with $\emptyset$;
    foreach $x \in I \backslash B$ do
        Add $\varphi(x + B)$ to $\mathcal{F}_B$
    end
    $Succ$=minimal inclusion subsets of $\mathcal{F}_B$;
    return $Succ$
end
```

Algorithm 1: Immediate successors algorithm

eration. Please refer to [6] for a complete description. The key idea behind this algorithm is to improve its efficiency by counting the objects, inspired from relational databases. Instead of considering the subset of observations as the extent of a subset of attributes, the computation of the inclusion-maximal subsets on $\mathcal{F}_A$ is made considering only its cardinality, and an incremental strategy. Four cases have to be considered to test, for each attribute x of $I \backslash B$ and each already inserted potential successor $X \subseteq I \backslash B$, the inclusion between $\varphi(B + X)$ and $\varphi(B + x)$:

1. Merge x with X when $\varphi(B + x) = \varphi(B + X)$.
2. Eliminate x as potential successor of B when $\varphi(B + x) \supset \varphi(B + X)$
3. Eliminate X as potential successor of B when $\varphi(B + x) \subset \varphi(B + X)$
4. Insert x as potential successor of B when x is neither eliminated or merged with X.

The inclusion test between $\varphi(B + X)$ and $\varphi(B + x)$ can easily be performed using the count function c and the following proposition from [6].

Proposition 1 ([6]):

$$\varphi(B + X) \subseteq \varphi(B + x) \Longleftrightarrow c(B + X + x) = c(B + X) \tag{3}$$

The count function c associates to any subset X of attributes the cardinality of the subset $\beta(X)$:

$$c(X) = |\beta(X)| \tag{4}$$

$$\varphi(B) = B + \{x \in I \backslash B \ : \ c(B) = c(B + x)\} \tag{5}$$

The count function corresponds to the notion of support introduced in association rule-mining, and is in particular used by Titanic algorithm [22].

It has been proven to be effective on a large amount of objects with a complexity of $\mathcal{O}((|I| - |B|)^2 * \mathcal{O}(c(B + X)))$. This has to be compared to the complexity of the classical immediate successors algorithm in $\mathcal{O}(|I|^2 * |O|)$.

```
Name: Immediates_Successors_LOA
Data: A context $K = (O; I, (\alpha, \beta))$ ; A closed set $B$ of the closed set lattice
      $(\mathbb{C}_I, \subseteq)$ of $K$
Result: The immediate successors of $B$ in the lattice
begin
    Init the set system $Succ_B$ with $\emptyset$;
    foreach $x \in I \setminus B$ do
        add = true;
        foreach $X \in Succ_B$ do
            \\ Case 1: Merge x and X in the same potential successor
            if $c(B+x) = c(B+X)$ then
                if $c(B+X+x) = c(B+x)$ then
                    replace $X$ by $X+x$ in $Succ_B$;
                    add=false; break;
                end
            end
            \\ Case 2: Eliminate x as potential successor
            if $c(B+x) < c(B+X)$ then
                if $c(B+X+x) = c(B+x)$ then
                    add=false; break;
                end
            end
            \\ Case 3: Eliminate X as potential successor
            if $c(B+x) > c(B+X)$ then
                if $c(B+X+x) = c(B+X)$ then
                    delete $X$ from $Succ_B$
                end
            end
        end
        \\ Case 4: Insert x as a new potential successor ;
        if add then  add $\{x\}$ to $Succ_B$;
    end
    return $Succ_B$;
end
```

Algorithm 2: LOA immediate successors algorithm

The ability of the algorithm to handle huge sets of data is very interesting. The impact of the number of observations on the performances can be limited, depending on the implementation of c, using for example multiple keys and robust algorithms used in databases that do not need to load all data for computing a cardinality [6].

4 Implementation

4.1 Data structures

We choose to implement the Limited Object Access algorithm in Java. The performance of the Limited Object Access presented in [6] comes from a PHP/MySQL implementation, backed up by the efficient SQL indexation system. Its behavior without the help of SQL remains to be seen.

While profiling the execution of a naive implementation, conducted without paying attention to optimization, we noticed that up to 86% of the computation time was used for the calculation of the candidate sets of attributes' extent. The extent of a set of attributes is the intersection of the extents of each of its elements. It appears that the most time consuming step of the extent calculation (up to 87% of the running time) is the intersection of two sets of elements. While it only takes around 50 microseconds to perform, the calls pile grows fast with the size of the dataset, resulting in a delivery time of full seconds, even minutes. A particular attention must be paid to the optimization of the extent calculation, both in terms of calls amount and processing time.

The number of calls is limited to three for a foreach loop, as $c(B+x)$, $c(B+X)$ and $c(B+X+x)$ are consistent and can be computed once and for all at the beginning of the second loop. If they were not, the *count* function would have been called up to 8 times (2 times per if, plus 2 times for the last if).

Classical Java containers, like *HashSet* or *TreeSet*, are well suited for the task of storing the observations and attributes but are a bit too sophisticated for the representation of a simple context's binary table. In fact, observations do not necessarily have to be directly manipulated to compute the extent, a fortiori its cardinality, of a set of attributes. Assuming that the observations are sorted in a final order, each of them can be represented by its index in this order. So the extent of an attribute becomes a binary word whose length is the cardinality of the observations set. In this word, a 1 (resp. a 0) means this index's observation is (resp. is not) part of the extent. Java, as many programming languages, has a *BitSet* class to manipulate and execute operations over such data type.

The extent (resp. intent) of each attribute (resp. observation) of the context is computed and stored as a binary word once and for all at the beginning of the execution. Then, the extent of a set of attributes can be computed using a *logical AND* on the successive extents of all of its elements (see Table 2). The immediate benefit comes from the rapidity of the *AND* operation, performed in less than one microsecond (with a complexity of $\mathcal{O}(\mathrm{E}(n/w))$, n being the length of the bitset and w the size of the memory words used to store the basic *AND* operations). It is more than 50 times as fast as an intersection between two TreeSets for the same number of calls. Furthermore, this sticks to the primal boolean representation of a context (see Table 1) and the lattice (nor any part of it) does not have to be generated at any time.

Each attribute and each object is mapped to its bit index in the binary word for output readability purpose.

	p1	p2	p3	p4	p5
contains:balloon	1	1	1	1	0
shape:high	1	0	0	1	0
contains:tree	0	0	0	1	1
Extent	0	0	0	1	0

Table 2. Extent of the set of attributes $\{b, h, t\}$

5 Experiment

The dataset is made of 848 comic books panels, the observations, and two kinds of attributes. The first category, made of 100 attributes, is shared by the whole set of observations with a proportion going from 2 to 11 attributes for an average of 7. 28 attributes are assigned to a single observation. The second category includes the first one and adds 3433 attributes, leading to an average distribution of 15 attributes per observation. 3403 out of the 3533 attributes belong to less than 3 observations. Only 15 are shared by more than 100 objects.

As this system is meant to be used in a browsing context through relevance feedback, it has to be efficient going both ways from a concept (towards its successors and its predecessors). We ran our algorithm both on the calculation of the immediate successors of the bottom concept $\perp$ and the immediate predecessors of the top concept $\top$. We computed the latter as the immediate successors of $\perp$ on the inverted context (which is rigorously the same thing as calculating the immediate predecessors of $\top$ in the regular context – see Fig. 2 and 3). We choose $\perp$ as starting point because, according to our dataset, it is the concept that is supposed to have the most immediate successors.

Table 3 shows the processing times in seconds of the classical immediate successors algorithm and the Limited Object Access algorithm, both tuned with TreeSet and BitSet data structures for different scenarios corresponding to different complexity values. Processes have been run on a 8GB DDR3 machine, powered by a 2.7GHz quad core Xeon. The results show a significant improvement of the computation time which can be attributed, for one part, to LOA and, for the other part, to the use of binary words.

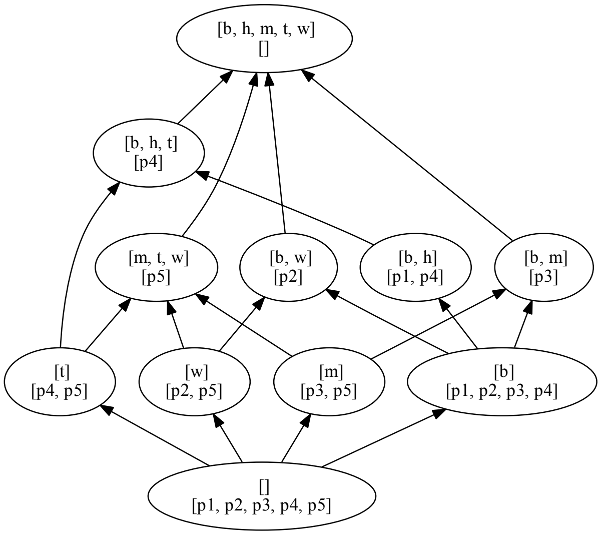


Fig. 2. Concept lattice generated from the context presented in Table 1. Intent: Attributes, Extent: Observations

	Immediate successors		Immediate predecessors	
	$\|O\| = 848$ $\|I\| = 100$	$\|O\| = 848$ $\|I\| = 3533$	$\|O\| = 100$ $\|I\| = 848$	$\|O\| = 3533$ $\|I\| = 848$
Classical + TreeSet	3.06	11767.52	549.76	994.00
Classical + BitSet	0.77	196.58	62.39	9.77
LOA + TreeSet	0.29	11.26	5.65	1183.75
LOA + BitSet	0.02	0.15	0.24	1.20

Table 3. Computation time (in seconds) of the immediate successors of $\perp$ and immediate predecessors of $\top$.

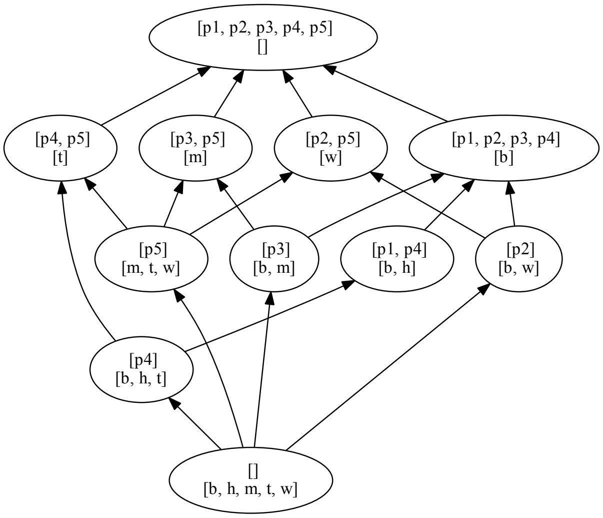


Fig. 3. Concept lattice generated from the inversion of the context presented in Table 1 Intent: Observations, Extent: Attributes

The bitset improvement over the LOA implementation shows a reduction of the execution time going from 14 to over 900 times. The calculation of the immediate predecessors of $\top$ over the set of 3533 attributes is the worst possible case as it results in 848 different concepts of one observation, each of them with a set of around 20 attributes. The running time is almost entirely taken by the million of extent calculations, which is why the gain is more spectacular here.

A test on 500 randomly picked concepts has been run over the third dataset ($|O| = 100, |I| = 848$) resulting in a mean processing time of 0.18 second.

Computation time shrinkage is minimized on the classical method as the optimization only applies on the closure operation, which is a fraction of the global computation time.

We deal here with computation times kept below the second on a reasonable machine. This starts to be interesting in terms of human-machine interaction capabilities where at least a dozen of concept's immediate successors have to be calculated at each step.

6 Conclusion and ongoing work

We presented an efficient Java implementation of the immediate successors calculation. The short processing times on quite large datasets are promising and make the query by navigation through a lattice possible. The source code will be made available soon, along a full Java library to handle lattices.

7 Acknowledgment

This work was supported by the European Regional Development Fund, the region Poitou-Charentes (France), the General Council of Charente Maritime (France) and the town of La Rochelle (France).

References

1. G. Birkhoff. *Lattice theory.* American Mathematical Society, 3d edition, 1967.
2. J.P. Bordat. Calcul pratique du treillis de Galois d'une correspondance. *Math. Sci. Hum.*, 96:31–47, 1986.
3. Claudio Carpineto and Giovanni Romano. Ulysses: A lattice-based multiple interaction strategy retrieval interface. In Brad Blumenthal, Juri Gornostaev, and Claus Unger, editors, *Human-Computer Interaction*, volume 1015 of *Lecture Notes in Computer Science*, pages 91–104. Springer Berlin Heidelberg, 1995.
4. N. Caspard and B. Monjardet. The lattice of closure systems, closure operators and implicational systems on a finite set: a survey. *Discrete Applied Mathematics*, 127(2):241–269, 2003.
5. Cyb. *Bubblegôm Gôm vol. 1, pp. 3*, volume 1. Studio Cyborga, Goven, France, 2009.
6. C. Demko and K. Bertet. Generation algorithm of a concept lattice with limited object access. In *Proc. of Concept lattices and Applications (CLA'11)*, pages 113–116, Nancy, France, October 2011.
7. B. Ganter. Two basic algorithms in concept analysis. *Technische Hochschule Darmstadt (Preprint 831)*, 1984.
8. B. Ganter and R. Wille. *Formal Concept Analysis, Mathematical foundations.* Springer Verlag, Berlin, 1999.
9. R. Godin, C. Pichet, and J. Gecsei. Design of a browsing interface for information retrieval. *SIGIR Forum*, 23(SI):32–39, May 1989.
10. Clément Guérin. Ontologies and spatial relations applied to comic books reading. In *PhD Symposium of Knowledge Engineering and Knowledge Management (EKAW)*, Galway, Ireland, October 2012.
11. Clément Guérin, Karell Bertet, and Arnaud Revel. An approach to Semantic Content Based Image Retrieval using Logical Concept Analysis. Application to comicbooks. In *International Workshop "What can FCA do for Artificial Intelligence?" FCA4AI*, pages 53–56, France, August 2012.
12. Munro I. Efficient determination of the transitive closure of a directed graph. *Information Processing Letter*, pages 56–58, 1971.
13. S. Kuznetsov and S. Obiedkov. Comparing performance of algorithms for generating concept lattices. *Journal of Experimental and Theorical Artificial Intelligence*, 14(2-3):189–216, 2002.

14. Christian Lindig. Fast concept analysis. *Working with Conceptual Structures-Contributions to ICCS*, pages 235–248, 2002.
15. Fisher M.J. and Meyer A.R. Boolean matrix multiplication and transitive closure. In 12^{th} *Annual Sympsosium on Switching and Automata Theory*, pages 129–131, 1971.
16. E. Norris. An algorithm for computing the maximal rectangles in a binary relation. *Revue Roumaine de Mathématiques Pures et Appliquées*, 23(2), 1978.
17. L. Nourine and O. Raynaud. A fast algorithm for building lattices. *Information Processing Letters*, 71:199–204, 1999.
18. Uta Priss. Lattice-based information retrieval. *Knowledge Organization*, 27:132–142, 2000.
19. Christophe Rigaud, Norbert Tsopze, Jean-Christophe Burie, and Jean-Marc Ogier. Robust frame and text extraction from comic books. In Young-Bin Kwon and Jean-Marc Ogier, editors, *Graphics Recognition. New Trends and Challenges*, volume 7423 of *Lecture Notes in Computer Science*, pages 129–138. Springer Berlin Heidelberg, 2013.
20. Yong Rui, Thomas S Huang, Michael Ortega, and Sharad Mehrotra. Relevance feedback: a power tool for interactive content-based image retrieval. *Circuits and Systems for Video Technology, IEEE Transactions on*, 8(5):644–655, 1998.
21. Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.
22. G. Stumme, R. Taouil, Y. Bastide, N. Pasquier, and L. Lakhal. Computing iceberg concept lattices with TITANIC. *Data and Knowledge Engineering*, 42(2):189–222, August 2002.

CryptoLat - a Pedagogical Software on Lattice Cryptomorphisms and Lattice Properties

Florent Domenach

Computer Science Department, University of Nicosia
46 Makedonitissas Ave., 1700 Nicosia, Cyprus
`domenach.f@unic.ac.cy`

Abstract. Although lattice theory is a rich field dating from Dedekind, Birkhoff and Öre, few studies in FCA use lattice properties to enhance their results. Moreover, out of the many cryptomorphisms associated with lattices, only the ones associating context, lattice and implicational system are effectively studied.
`CryptoLat` is a software implemented in C$\sharp$ which provides an intuitive view on different cryptomorphisms of lattices as well as on different properties of lattices. Its purpose is pedagogical, for students and researchers likewise, and work by showing incremental changes in the lattice and the associated cryptomorphisms.

Keywords: Lattice Cryptomorphisms, Lattice Properties, Pedagogy

1 Introduction

This article, together with the software, originates from an observation about the current state of research in the lattice and the FCA communities. Lattice theory dates back to the works of Dedekind, Birkhoff [7] and Öre [19], and is a rich field in which FCA is deeply and fundamentally rooted. Despite that fact, relatively few studies in FCA takes advantage of lattice properties to enhance their results. This dichotomy of the two communities is apparent in the research interests: on the one hand, researchers from the graph and ordered set communities are focusing on equivalence theorem and characterizations, while on the other hand data mining researchers are more focused on practical or practically oriented results.

Although it is natural that this dichotomy exists because of the nature of each community's focus of interest, it is the author's suggestion that improvement of the communication between the two communities may produce a synergistic effect. The study of lattice cryptomorphisms and lattice properties is important for many reasons: other than giving a better understanding of the tools being used, it was shown to lead to the discovery of efficient algorithms in counting the number of Moore families [16] or calculating the stem basis [6] for example.

There are two main purposes for this software: first, to promote and to give a better understanding of the many shapes and forms of lattices thanks to an incremental approach - any change made in any equivalent cryptomorphism is transcribed to the other cryptomorphisms. Second, to help familiarize users to fundamental latticial properties that are recalculated after any modification. Its purpose is pedagogical for both students and researchers. Students can learn and get a better understanding on what is happening during the construction of the lattice and the implicational basis, while researchers can test hypotheses and find small illustrating examples. The program is written in C♯ and is freely available (executable and source code) at `http://www.cs.unic.ac.cy/florent/software.htm`

This paper is organized as follows: we start in Sec. 2 by describing the software itself. Sec 3 will recall the well-known definitions of lattices and some of the existing cryptomorphisms implemented in CryptoLat, together with a short explanation on how the equivalence is done. Finally, we compare in Sec. 5 CryptoLat with already existing software.

2 Description of CryptoLat

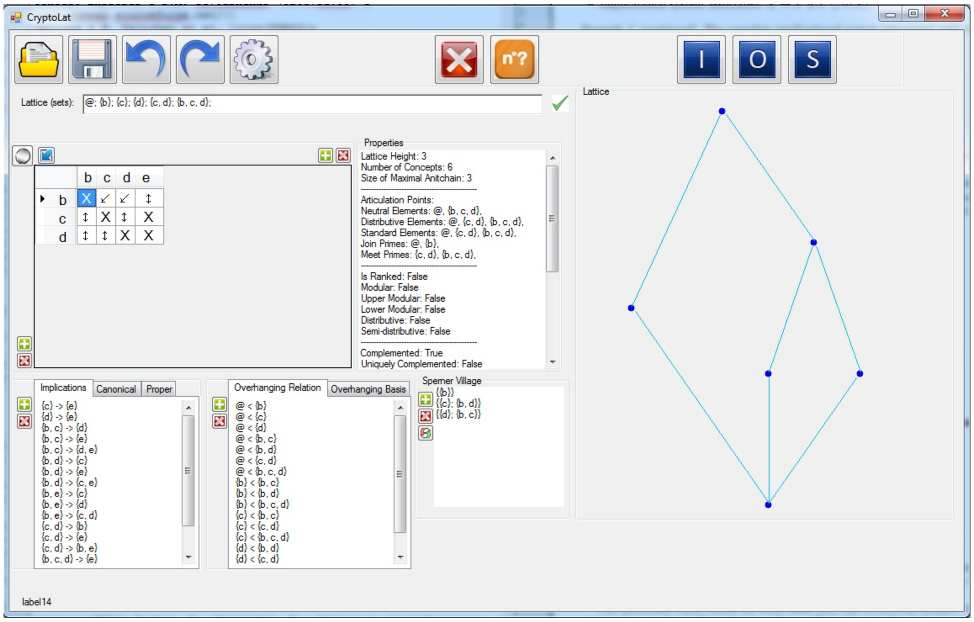
`CryptoLat` software is structured around a model view controller type of software architecture: each cryptomorphism described in Sec. 3 is a view with which the user can interact. The model is the reduced context associated with a Moore family on which all the calculations are based. When a view is updated, by adding or deleting a Sperner family for example, then the Moore family associated is calculated, the associated concept lattice, together with all the properties, is computed, and the other views get updated.

Depending on which view getting updated, the process may involve different steps. If the overhanging list or the Sperner village change, then first the Moore family associated with the new list, and the reduced context is then calculated. Similarly, if the input is an arbitrary set system, then the closure system is calculated as in Sec. 3.2. If the input is on the set of implications, since implications are usually on the set of attributes, the corresponding Moore family will be dual to the previous ones. From the reduced context, we are using Closed by One algorithm to calculate the concept lattice.

Fig. 1 shows a screen shot of the software with the five different views: the Moore family on top, the context as a datagrid, the list of implications, the list of overhanging and the list of Sperner families. The drawing of the lattice is done with a simple algorithm that optimise the position of each concept depending on the number (and position) of concepts covered.

3 Cryptomorphisms of Lattices

In this section, we present the different cryptomorphisms of lattices implemented in `CryptoLat` in details; we do not claim to be exhaustive, so we refer the readers to [9] for a survey on those cryptomorphisms. Since we are describing implementations, the following descriptions needs only to be for the finite case.



Lattice (sets): @; {b}; {c}; {d}; {c, d}; {b, c, d};

	b	c	d	e
b	X	↙	↙	↕
c	↕	X	↕	X
d	↕	↕	X	X

```
Lattice Height: 3
Number of Concepts: 6
Size of Maximal Anitchain: 3
--------------------------------
Articulation Points:
Neutral Elements: @, {b, c, d},
Distributive Elements: @, {c, d}, {b, c, d},
Standard Elements: @, {c, d}, {b, c, d},
Join Primes: @, {b},
Meet Primes: {c, d}, {b, c, d},
--------------------------------
Is Ranked: False
Modular: False
Upper Modular: False
Lower Modular: False
Distributive: False
Semi-distributive: False
--------------------------------
Complemented: True
Uniquely Complemented: False
```

```
{c} -> {e}
{d} -> {e}
{b, c} -> {d}
{b, c} -> {e}
{b, c} -> {d, e}
{b, d} -> {c}
{b, d} -> {e}
{b, d} -> {c, e}
{b, e} -> {c}
{b, e} -> {d}
{b, e} -> {c, d}
{c, d} -> {b}
{c, d} -> {e}
{c, d} -> {b, e}
{b, c, d} -> {e}
```

```
@ < {b}
@ < {c}
@ < {d}
@ < {b, c}
@ < {b, d}
@ < {c, d}
@ < {b, c, d}
{b} < {b, c}
{b} < {b, d}
{b} < {b, c, d}
{c} < {b, c}
{c} < {c, d}
{c} < {b, c, d}
{d} < {b, d}
{d} < {c, d}
```

```
{{b}}
{{c}; {b, d}}
{{d}; {b, c}}
```

Fig. 1. `CryptoLat` software with the different views displaying our running example

3.1 Lattice and Irreducible Elements

Different yet equivalent definitions for lattices exist: here, we will define a *lattice* $(L, \wedge, \vee)$ as a set L with two binary operators satisfying the following four properties:

- Associativity: $x \wedge (y \wedge z) = (x \wedge y) \wedge z$ and $x \vee (y \vee z) = (x \vee y) \vee z$
- Commutativity: $x \wedge y = y \wedge x$ and $x \vee y = y \vee x$
- Idempotence: $x \wedge x = x$ and $x \vee x = x$
- Absorption: $x \wedge (y \vee x) = x$ and $x \vee (y \wedge x) = x$

One can associate an order on the elements of L to these operators as follow: $\forall x, y \in L, x \leq y$ if $x \wedge y = x$ or, equivalently, $x \vee y = y$. Fig. 2 shows a lattice having six elements using its Hasse diagram representation: the elements of L are depicted by nodes in the plane, and a line segment connects the nodes corresponding to a covering pair $x \prec y$ (i.e. $x \leq y$ and $\forall z, x \leq z \leq y$ implies $x = z$ or $y = z$), where the node representing y is placed 'higher' in the plane than the node representing x.

Some elements of lattices are of particular interest: in any lattice, we have a minimal element, usually denoted by 0_L or $\bot$, and a maximal element 1_L or $\top$. Irreducible elements of the lattice are also the subject of studies since they constitute minimal generating sets, and are often used to characterize classes of lattices. An element j of L is *join-irreducible* if $j = \vee X$ then $j \in X$, or,

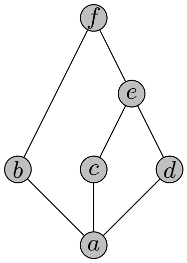


Fig. 2. Example of a lattice with 6 elements

equivalently, j covers only one element in L. Dually, an element m of L is *meet-irreducible* if $m = \wedge X$ then $m \in X$, or m is covered by only one element of L. Let J_L and M_L be the sets of join and meet-irreducible of L.

3.2 Moore Family and Closure Operator

Let S be a finite set. A *closure system* (or *Moore family*) $\mathcal{C}$ on S is a family of subsets of S satisfying $S \in \mathcal{C}$ and, for all $A, B \in \mathcal{C}$, $A \cap B \in \mathcal{C}$. A closure system is a lattice with the following operations:

- $C_1 \wedge C_2 = C_1 \cap C_2$
- $C_1 \vee C_2 = \cap\{C \in \mathcal{C} : C_1 \cup C_2 \subseteq C\}$

Let $\mathcal{A}$ be an arbitrary set system on a set S. Then one can associate a closure system $\mathcal{C}_A$ to $\mathcal{A}$:

$$\mathcal{C}_A = \{\cap\{A_i : A_i \in \mathcal{A}\}\} \cup \{S\}$$

Example 1 (continued). The family $\{\emptyset, \{G_0\}, \{G_1\}, \{G_2\}, \{G_1, G_2\}, \{G_0, G_1, G_2\}\}$, with the operators previously defined, constitute a lattice isomorphic to the lattice of Fig. 2.

Another fundamental cryptomorphism of lattice concerns closure operators. A *closure operator* on a set S is a map $\sigma : \mathcal{P}(S) \to \mathcal{P}(S)$ satisfying $\forall X, Y \subseteq S$:

- Isotone: $X \subseteq Y \Rightarrow \sigma(X) \subseteq \sigma(Y)$
- Extensive: $X \subseteq \sigma(X)$
- Idempotent: $\sigma(\sigma(X)) = \sigma(X)$

These three properties together are equivalent to extensivity and path independence property [20]: $\sigma(X \cup Y) = \sigma(\sigma((X) \cup \sigma((Y))$.

The well-known equivalence between closure operators and closure systems is the following: given the closure system $\mathcal{C}$, the associated closure operator is $\sigma_{\mathcal{C}}(X) = \cap\{C \in \mathcal{C} : X \subseteq C\}$. Conversely, given a closure operator σ, the associated closure system is the set of fixed points of σ, i.e. $\mathcal{C}_\sigma = \{X \subseteq S : X = \sigma(X)\}$.

3.3 Context Associated with a Lattice

One of the most celebrated cryptomorphism in the FCA community is the one between a lattice and a binary relation, also known as the basic theorem of Formal Concept Analysis [3, 13].

Formally, a context (G, M, I) is a binary relation I between a set of objects G and a set of attributes M. One can associate the Galois connection between $\mathcal{P}(G)$ and $\mathcal{P}(M)$ to this binary relation as $\forall A \subseteq G, B \subseteq M$,

$$A' = \{m \in M : \forall g \in A, (g, m) \in I\}$$

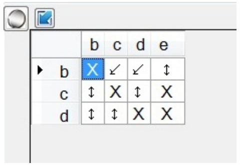
$$B' = \{g \in G : \forall m \in B, (g, m) \in I\}$$

The lattice associated with the context (G, M, I), also called the concept lattice or Galois lattice, is the set of all concepts (A, B), with $A' = B$ and $B' = A$, together with the order $(A_1, B_1) \leq (A_2, B_2) \iff A_1 \subseteq A_2$ $(\iff B_1 \supseteq B_2)$. Conversely, any lattice L is isomorphic to the Galois lattice of the context $(J_L, M_L, \leq)$, also called the reduced context.

Two other binary relations, the arrows relations, on $\mathcal{P}(J) \times \mathcal{P}(M)$ are often used for characterization theorems. Let $j \in J, m \in M$, we define:

- $j \uparrow m \iff j \nleq m$ and $j < m^+$
- $j \downarrow m \iff j \nleq m$ and $j^- < m$
- $j \updownarrow m \iff j \uparrow m$ and $j \downarrow m$

Example 1 (continued). The lattice associated with the context of Fig. 3 is isomorphic to the lattice of Fig. 2.



	b	c	d	e
b	X	↙	↙	↕
c	↕	X	↕	X
d	↕	↕	X	X

Fig. 3. The reduced context, together with the arrows relations, associated with our running example.

3.4 Implications

An implicational system $\mathcal{I}$ on S is a binary relation on $\mathcal{P}(S)$, and if $(A, B) \in \mathcal{I}$, we denote it $A \rightarrow_{\mathcal{I}} B$ or $A \rightarrow B$ if no confusion is possible. We say that A implies B or $A \rightarrow B$ is an implication (of $\mathcal{I}$). A *complete implicational system* $\mathcal{I}$ is an implicational system on $\mathcal{P}(S)$ satisfying, $\forall A, B, C, D \subseteq S$:

- $B \subseteq A$ implies $A \rightarrow B$;

- $A \to B$ and $B \to C$ imply $A \to C$;
- $A \to B$ and $C \to D$ imply $(A \cup C) \to (B \cup D)$.

[2] showed that a one-to-one correspondence between closure operators and complete implicational system exists. The closure system associated to a complete implicational system is $\mathcal{C}_{\mathcal{I}} = \{C \subseteq S : \forall X \subseteq S, [X \subseteq C \text{ and } X \to Y] \Rightarrow Y \subseteq C\}$, while the implicational system associated to a closure mapping is $\{X \to Y : Y \subseteq \sigma(X)\}$.

Example 1 (continued). The complete implicational system associated with the lattice in Fig. 2 is the following:

$$\begin{array}{rcc} c \to e; & d \to e; & b, c \to d; \\ b, c \to e; & bc \to d, e; & b, d \to c; \\ b, d \to e; & b, d \to c, e; & b, e \to c; \\ b, e \to d; & b, e \to c, d; & c, d \to b; \\ c, d \to e; & c, d \to b, e; & c, d, e \to b; \\ b, d, e \to c; & b, c, e \to d; & b, c, d \to e; \end{array}$$

As the above example shows, many implications in a complete implicational system can be redundant: in our example, it is evident that if we have $c \to e$ then we have $b, c \to e$. So smaller implicational system allowing a smaller representation of a complete implicational system is of interest. We say that an implicational system Σ is a *generating system* for $\mathcal{I}$ if any implication of $\mathcal{I}$ can be obtained from Σ by applying recursively the rules defining a complete implicational system. A minimal generating system for $\mathcal{I}$ is called a basis for $\mathcal{I}$, and a minimum basis is called a *canonical basis*.

One particular basis, called the stem basis [15], can be directly defined: the stem basis is made of all implications $X \to \sigma(X) - X$, with X a critical set. A set $Q \subseteq S$ is called a quasi-closed set if $Q \notin \mathcal{C}$ and $\mathcal{C} + \{Q\}$ is a closure system, and Q is a F-quasi-closed set if Q is quasi-closed and $\sigma(Q) = F$. So a set $C \subseteq S$ is critical if there exists $F \in \mathcal{C}$ such that C is a minimal F-quasi-closed set.

Example 1 (continued). Canonical basis associated with the lattice of Fig.2.

$$b, e \to c, d; \;\; c \to e; \;\; d \to e; \;\; c, d, e \to b;$$

3.5 Overhanging Relation

An *overhanging relation* $\mathcal{O}$ is a binary relation on $\mathcal{P}(S)$ that was originated in [1] in consensus theory on trees under the term of nesting, and was generalized in [12] to any lattice. We will indifferently write $(A, B) \in \mathcal{O}$ or $A \; \mathcal{O} \; B$ to signify that a set A is overhanged in B. An overhanging relation satisfies the following properties:

- for all $A, B \in S$, $A \; \mathcal{O} \; B$ implies $A \subset B$;
- for all $A, B, C \in S$, $A \subset B \subset C$ implies $[A \; \mathcal{O} \; C \iff A \; \mathcal{O} \; B \text{ or } B \; \mathcal{O} \; C]$;
- for all $A, B \in S$, $A \; \mathcal{O} \; (A \cup B)$ implies $(A \cap B) \; \mathcal{O} \; B$.

We showed in [12] that overhanging relations and closure operators (and so lattices) are in a one-to-one correspondence by associating the overhanging relation defined as $A \mathcal{O} B \iff A \subset B$ and $\sigma(A) \subset \sigma(B)$ to any closure operator, and, conversely, associating the closure operator $\sigma(X) = \{x \in S : x\ \mathcal{O}^c\ X \cup \{x\}\}$ to any overhanging relation. In fact, an overhanging relation can be seen as a kind of negative implication, as given by the equivalence that $A \to B$ if and only if $(A, A \cup B) \notin \mathcal{O}$

Example 1 (continued). The overhanging relation associated with the lattice of Fig.2 is as follows:

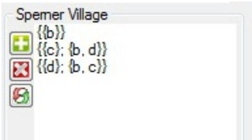
$$\begin{array}{ccc}
\emptyset\ \mathcal{O}\ b & \emptyset\ \mathcal{O}\ c & \emptyset\ \mathcal{O}\ d \\
\emptyset\ \mathcal{O}\ b,c & \emptyset\ \mathcal{O}\ b,d & \emptyset\ \mathcal{O}\ c,d \\
\emptyset\ \mathcal{O}\ b,c,d & b\ \mathcal{O}\ b,c & b\ \mathcal{O}\ b,d \\
b\ \mathcal{O}\ b,c,d & c\ \mathcal{O}\ b,c & c\ \mathcal{O}\ c,d \\
c\ \mathcal{O}\ b,c,d & d\ \mathcal{O}\ b,d & d\ \mathcal{O}\ c,d \\
d\ \mathcal{O}\ b,c,d & c,d\ \mathcal{O}\ b,c,d &
\end{array}$$

3.6 Sperner Family

A *Sperner family* $\mathbb{F}$ on S is a family s.t. $\forall X, Y \in \mathbb{F}$, X and Y are incomparable for set inclusion. A *Sperner village* $\mathcal{V} = (\mathbb{F}_1, ..., \mathbb{F}_n)$ on S is a set of n Sperner families. The equivalence between a closure operator and a Sperner village is obtained as follows: given σ closure operator, for any $x \in S$ we create the Sperner family $\mathcal{C}_x = \{X \subseteq S : x \in \sigma(X)$ and X minimal for that property$\}$. Conversely, the closure operator associated with $\mathcal{V}$ is defined as $\sigma(X) = \{x \in S : X$ contains a set of $\mathcal{C}_x\}$.

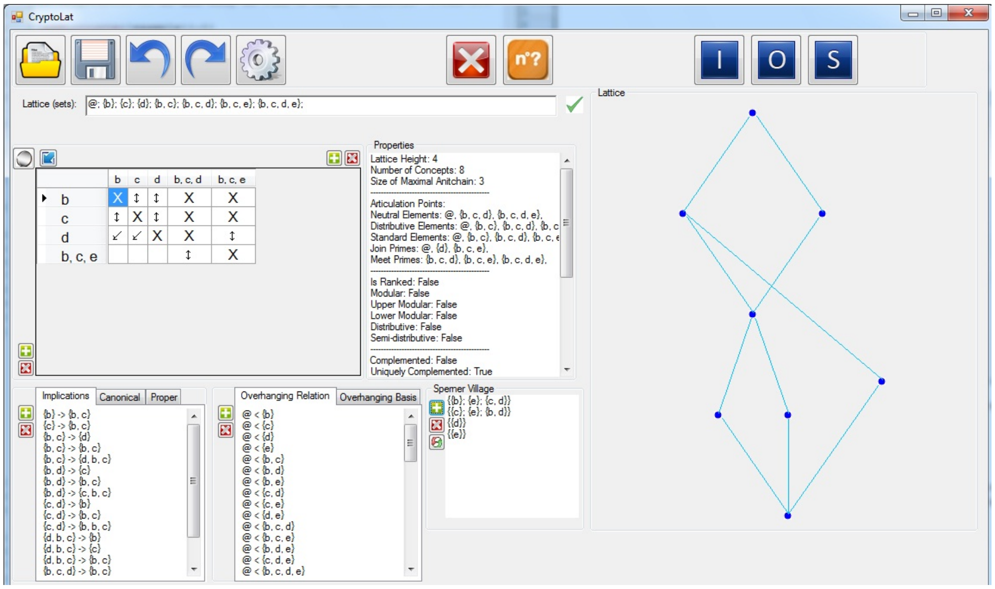
Example 1 (continued). The Sperner village equivalent with our running example is the following Fig. 4.

When we add to the existing Sperner village the Sperner family $\{\{e\}, \{b, c\}\}$, the software computes the smallest Sperner village possible from the existing village together with the new family, and from this Sperner village finds the associated concept lattice. Fig. 5 shows the results.



{{b}}
{{c}; {b, d}}
{{d}; {b, c}}

Fig. 4. Sperner village associated with our running example.



Lattice (sets): @; {b}; {c}; {d}; {b, c}; {b, c, d}; {b, c, e}; {b, c, d, e};

	b	c	d	b, c, d	b, c, e
b	X	↕	↕	X	X
c	↕	X	↕	X	X
d	↙	↙	X	X	↕
b, c, e				↕	X

```
Lattice Height: 4
Number of Concepts: 8
Size of Maximal Anitchain: 3
-------------------------------
Articulation Points:
Neutral Elements: @, {b, c, d}, {b, c, d, e},
Distributive Elements: @, {b, c}, {b, c, d}, {b, c
Standard Elements: @, {b, c}, {b, c, d}, {b, c, e
Join Primes: @, {d}, {b, c, e},
Meet Primes: {b, c, d}, {b, c, e}, {b, c, d, e},
-------------------------------
Is Ranked: False
Modular: False
Upper Modular: False
Lower Modular: False
Distributive: False
Semi-distributive: False
-------------------------------
Complemented: False
Uniquely Complemented: True
```

```
{b} -> {b, c}
{c} -> {b, c}
{b, c} -> {d}
{b, c} -> {b, c}
{b, c} -> {d, b, c}
{b, d} -> {c}
{b, d} -> {b, c}
{b, d} -> {c, b, c}
{c, d} -> {b}
{c, d} -> {b, c}
{c, d} -> {b, b, c}
{d, b, c} -> {b}
{d, b, c} -> {c}
{d, b, c} -> {b, c}
{b, c, d} -> {b, c}
```

```
@ < {b}
@ < {c}
@ < {d}
@ < {e}
@ < {b, c}
@ < {b, d}
@ < {b, e}
@ < {c, d}
@ < {c, e}
@ < {d, e}
@ < {b, c, d}
@ < {b, c, e}
@ < {b, d, e}
@ < {c, d, e}
@ < {b, c, d, e}
```

```
{{b}; {e}; {c, d}}
{{c}; {e}; {b, d}}
{{d}}
{{e}}
```

Fig. 5. Screenshot of the software after adding a new Sperner family

4 Properties of lattices

The main focus of the software concerns the dependency between the different cryptomorphisms associated with lattices. In the same pedagogical spirit, we also implemented different properties of lattices and elements of lattices, briefly described in Tab. 1 and in Tab. 2 for completeness sake. We refer the readers to [11, 14, 22] for more information on lattices and lattice properties.

5 Existing Software

All of the software mentioned in Table 3 have different goals. In terms of ease of use, Concept Explorer is the closest to our goal, but offers limited functionalities. OpenFCA is similar to Concept Explorer, offering an interesting, web based and intuitive approach to context creation, lattice visualization and attribute exploration. FCAstone's main purpose is file formats conversion to improve interoperability between FCA and graph editing. The other software (Galicia, ToscanaJ, Lattice Miner, FCART) were designed to push the boundaries of the scalability of analysis and lattice drawing. Research wise, Formal Concept Analysis Research Toolbox (FCART) is a particularly interesting approach of a universal

Table 1. List of lattice properties

Types of Lattices	
Atomistic	Every join-irreducible element of L is an atom
Co-atomistic	Every meet-irreducible element is covered by 1_L
Ranked	Tthere is a ranking function $r : L \to \mathbb{N}$ such that $\forall x, y \in L, x \prec y$ implies $r(y) = r(x) + 1$
Upper semi-modular	$\forall x, y \in L$: $x \wedge y \prec x \Rightarrow y \prec x \vee y$
Lower semi-modular	$\forall x, y \in L$: $y \prec x \vee y \Rightarrow x \wedge y \prec x$
Modular	Upper and lower semi-modular
Distributive	$\forall x, y, z \in L$, $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$
Semi-distributive	$\forall x, y, z \in L$, we have both $x \wedge y = x \wedge z \Rightarrow x \wedge y = x \wedge (y \vee z)$ and $x \vee y = x \vee z \Rightarrow x \vee y = x \vee (y \wedge z)$

Families of Lattices	
Chain	$\forall x \in L, x \neq 0_L$, $x \in J_L$
Hierarchy	$\forall x \in S, \{x\} \in L$ and $\forall A, B \in L, A \cap B \in \{\emptyset, A, B\}$
Weak hierarchy	$\forall x \in S, \{x\} \in L$ and $\forall A, B, C \in L, A \cap B \cap C \in \{A \cap B, A \cap C, B \cap C\}$
Boolean	Isomorphic to the power set of some set

Complemented Lattices	
Complemented	$\forall x \in L, \exists y \in L$: $x \wedge y = 0_L$ and $x \vee y = 1_L$
Uniquely complemented	The complement is unique
Pseudo-complemented	$\forall x \in L, \exists y \in L$: $x \wedge y = 0_L$ and $\forall z \in L, x \wedge z = 0_L \Rightarrow z \leq y$
Relatively complemented	Any $[u, v]$ is relatively complemented, i.e. $\forall x$ in $[u, v]$, $\exists y$ such that $x \wedge y = u$ and $x \vee y = v$
Uniquely relatively complemented	The complement in $[u, v]$ is unique
Sectionally complemented	Any $[0_L, v]$ is complemented

Other Lattices Properties	
δ-hard	$\forall j, j' \in J_L, j \delta j'$, i.e. there exists $x \in L$ such that $j \not\leq x, j' \not\leq x$ and $j < j' \vee x$
δ-strong	The transitive graph of the δ relation is complete
Sperner poset	L is ranked, and the size of the maximal antichain is equal to the maximal size rank

Table 2. List of elements properties

x is distributive	$\forall y, z \in L$, $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$
x is standard	$\forall y, z \in L$, $y \wedge (x \vee z) = (y \wedge x) \vee (y \wedge z)$
x is neutral	$\forall y, z \in L$, $(x \wedge y) \vee (y \wedge z) \vee (x \wedge z) = (x \vee y) \wedge (x \vee z) \wedge (y \vee z)$
x is an articulation point	$\forall y \in L$, either $x \leq y$ or $y \leq x$
x is join-prime	$x \leq a \vee b$ implies $x \leq a$ or $x \leq b$
x is meet-prime	$a \wedge b \leq x$ implies $a \leq x$ or $b \leq x$

integrated environment to lattice construction and Formal Concept Analysis techniques, where one can test new algorithms for drawing lattice for example.

Table 3. Some FCA software tools (taken from [18])

Program title	Authors	Web-site
Concept Explorer	S.A. Yevtushenko [25]	`conexp.sourceforge.net`
Galicia	P. Valtchev et al. [23]	`www.iro.umontreal.ca/~galicia`
ToscanaJ	U. of Queensland, TU Darmstadt [24]	`toscanaj.sourceforge.net`
FcaStone	U. Priss et al. [21]	`fcastone.sourceforge.net`
Lattice Miner	B. Lahcen et al. [17]	`lattice-miner.sourceforge.net`
Conexp-clj	TU-Dresden, D. Brochman	`daniel.kxpq.de/math/conexp-clj`
OpenFCA	P. Borza, O. Sabou et al. [8]	`code.google.com/p/openfca`
FCART	A. Neznanov and D. Ilvovsky [18]	

6 Concluding Remarks

`CryptoLat` is a pedagogical and research oriented software in active development phase, and it is open source and freely available. It is still in alpha testing, and the next milestone of development is to make it cloud based with a web interface.

Many other cryptomorphisms of concept lattices exist but are not implemented yet. For example, [4] proved that we can associate a minimal separator of the co-bipartite graph to every concept of the concept lattice, and [16] showed a bijection between Moore families and ideal colour sets of the coloured poset. Similarly we are planning to increase the number of properties available, adding for example the dismantlable property (one can repeatedly remove a doubly irreducible element until the lattice becomes a chain) or being a convex geometry ($\emptyset \in L$ and $\forall x \in L$, there is a unique minimal (for inclusion) generator of x). the lattice drawing component is also under consideration, to be enhance and more user friendly.

References

1. Adams III, E.N.: N-trees as nestings: complexity, similarity and consensus. Journal of Classification 3, 299–317 (1986)
2. Armstrong, W.W.: Dependency structures of data base relationships. Information Processing 74, 580583 (1974)
3. Barbut, M., Monjardet, B.: Ordres et classification: Algèbre et combinatoire (tome II). Hachette, Paris (1970)
4. Berry, A., Sigayret, A.: Representing a concept lattice by a graph. Discrete Applied Mathematics 144 (12), 27–42 (2004)
5. Berry, A., Sigayret, A.: A Peep through the Looking Glass: Articulation Points in Lattices. In: Domenach, F., Ignatov, D., Poelmans, J. (Eds.) Formal Concept Analysis, Springer Berlin Heidelberg, 7278, 45-60 (2012)

6. Bertet, K.: The dependence graph of a lattice. In Szathmary, L., Priss, U. (Eds.): CLA 2012, 223–231 (2012)
7. Birkhoff, G.: Lattice Theory (Third ed.). Providence, R.I.: Amer. Math.Soc.(1967)
8. Borza, P.V., Sabou, O., Sacarea, C.: OpenFCA, an open source formal concept analysis toolox. Proc. of IEEE International Conference on Automation Quality and Testing Robotics (AQTR), 1–5 (2010)
9. Caspard, N., Monjardet, B.: The lattices of Moore families and closure operators on a finite set: a survey. Disc. App. Math. 127, 241–269 (2003)
10. Caspard, N., Leclerc, B., Monjardet, B.: Finite Ordered Sets: Concepts, Results and Uses. Cambridge University Press (2012)
11. Davey, B.A., Priestley, H. A.: Introduction to Lattices and Order, 2nd ed. Cambridge University Press (2002)
12. Domenach, F., Leclerc, B.: Closure systems, Implicational Systems, Overhanging Relations and the case of Hierarchical Classification. Math. Soc. Sci. 47 (3), 349–366 (2004)
13. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical Foundations, Springer Verlag (1999)
14. Grätzer, G.: General lattice theory. Academic Press (1978).
15. Guigues, J.-L., Duquenne, V.: Familles minimales d'implications informatives résultant d'un tableau de données binaires. Math. Sci. Hum. **95** , 5-18 (1986)
16. Habib, M., Nourine, L.: The number of Moore families on n=6. Disc. Math. 294(3), 291–296 (2005)
17. Lahcen, B., Kwuida, L.: Lattice Miner: A Tool for Concept Lattice Construction and Exploration. In Boumedjout, L., Valtchev, P., Kwuida, L., Sertkaya, B. (eds.): Supplementary Proceeding of International Conference on Formal concept analysis (ICFCA'10), 59-66 (2010)
18. Neznanov, A.A., Ilvovsky, D.A., Kusnetsov, S.O.: FCART: A New FCA-based System for Data Analysis and Knowledge Discovery. In Cellier, P., Distel, F., Ganter, B. (eds.): Contributions to the 11th International Conference on Formal Concept Analysis, 65–78 (2013)
19. Öre, O.: Galois connexions. Trans. Amer. Math. Soc. 55(3), 494-513 (1944)
20. Plott, C.R.: Path independence, rationality and social choice. Econometrica 41 (6), 10751091 (1973)
21. Priss, U.: FCA Software Interoperability. In: Belohlavek, R., Kuznetsov, S.O. (eds.), CLA 2008, 33144, (2008)
22. Roman, S: Lattices and Ordered Sets. Springer (2000)
23. Valtchev, P., Grosser, D., Roume, C., Rouane Hacene, M.: Galicia: An Open Platform for Lattices. In Using Conceptual Structures: Contributions to the 11th Intl. Conference on Conceptual Structures (ICCS'03), 241–254 (2003)
24. Vogt, F., Wille, R.: TOSCANA - a graphical tool for analyzing and exploring data. In Tamassia R., Tollis, I.G. (eds.), Graph Drawing, LNCS 894, 226-233 (1994)
25. Yevtushenko, S.A.: System of data analysis "Concept Explorer". Proceedings of the 7th national conference on Artificial Intelligence KII-2000, Russia (2000) 127-134

CRL-Chu correspondences

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

[1] University of Pavol Jozef Šafárik, Košice, Slovakia*
[2] Dept. Matemática Aplicada, Univ. Málaga, Spain**

Abstract. We continue our study of the general notion of L-Chu correspondence by introducing the category CRL-ChuCors incorporating residuation to the underlying complete lattice L, specifically, on the basis of a residuation-preserving isotone Galois connection λ. Then, the L-bonds are generalized within this same framework, and its structure is related to that of the extent of a suitably defined λ-direct product.

1 Introduction

Morphisms have been suggested [7] as fundamental structural properties for the modelling of, among other applications, communication, data translation, and distributed computing. Our approach can be seen within a research topic linking concept lattices with the theory of Chu spaces [10,11]; in the latter, it is shown that the notion of state in Scott's information system corresponds precisely to that of formal concepts in FCA with respect to all finite Chu spaces, and the entailment relation corresponds to *association rules* (another link between FCA with database theory) and, specifically, on the identification of the categories associated to certain constructions.

Other researchers have studied as well the relationships between Chu constructions and L-fuzzy FCA. For instance, in [1] FCA is linked to both order-theoretic developments in the theory of Galois connections and to Chu spaces; as a result, not surprisingly from our previous works, they obtain further relationships between formal contexts and topological systems within the category of Chu systems. Recently, Solovyov, in [9], extends the results of [1] to clarify the relationships between Chu spaces, many-valued formal contexts of FCA, lattice-valued interchange systems and Galois connections.

This work is based on the notion, introduced by Mori in [8], of Chu correspondences as morphisms between formal contexts. This categorical approach has been used in previous works [3,5,6]. For instance, in [6], the categories associated to L-formal contexts and L-CLLOS were defined and a constructive proof was given of the equivalence between the categories of L-formal contexts with L-Chu correspondences as morphisms and that of completely lattice L-ordered sets and their corresponding morphisms. Similar results can be found in [2], where a

---

* 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 Andaluci¿$\frac{1}{2}$ï¿$\frac{1}{2}$a project P09-FQM-5233.

new notion of morphism on formal contexts resulted in a category equivalent to both the category of complete algebraic lattices and Scott continuous functions, and a category of information systems and approximable mappings.

We are concerned with the category of fuzzy formal contexts and λ-Chu correspondences, built on the basis of a residuation-preserving isotone Galois connection λ. Then, the corresponding extension of the notion of bond between contexts is generalized to this framework, and its properties are studied.

2 Preliminaries

2.1 Residuated lattice

Definition 1. *A* complete residuated lattice *is an algebra* $\langle L, \wedge, \vee, 0, 1, \otimes, \to \rangle$ *where*

- $\langle L, \wedge, \vee, 0, 1 \rangle$ *is a complete lattice with the top* 1 *and the bottom* 0*,*
- $\langle L, \otimes, 1 \rangle$ *is a commutative monoid,*
- $\langle \otimes, \to \rangle$ *is an adjoint pair, i.e. for any* $a, b, c \in L$*:*

$$a \otimes b \leq c \text{ is equivalent to } a \leq b \to c$$

Definition 2. *A complete residuated lattice* $\mathcal{L} = \langle L, \wedge, \vee, 0, 1, \otimes, \to \rangle$ *such that for any value* $k \in L$ *holds* $\neg\neg k = k$ *where* $\neg k = k \to 0$ *is said to be endowed with* double negation law*.*

Lemma 1. *Let* $\mathcal{L}$ *be a complete residuated lattice satisfying the double negation law. Then for any* $k, m \in L$ *holds* $\neg k \to m = \neg m \to k$*.*

2.2 Basics of Fuzzy FCA

Definition 3. *An* L-fuzzy formal context $\mathcal{C}$ *is a triple* $\langle B, A, \mathcal{L}, r \rangle$*, where* B*,* A *are sets,* $\mathcal{L}$ *is a complete residuated lattice, and* $r\colon B \times A \to L$ *is an* L*-fuzzy binary relation.*

Definition 4. *Let* $\mathcal{C} = \langle B, A, \mathcal{L}, r \rangle$ *be an* L*-fuzzy formal context. A pair of* derivation operators $\langle \uparrow, \downarrow \rangle$ *of the form* $\uparrow\colon L^B \to L^A$ *and* $\downarrow\colon L^A \to L^B$*, is defined as follows*

$$\uparrow(f)(a) = \bigwedge_{b \in B} (f(b) \to r(b, a)) \text{ for any } f \in L^B \text{ and } a \in A,$$

$$\downarrow(g)(b) = \bigwedge_{a \in A} (g(a) \to r(b, a)) \text{ for any } g \in L^A \text{ and } b \in B.$$

Lemma 2. *Let* $\langle \uparrow, \downarrow \rangle$ *be a pair of derivation operators defined on an* L*-fuzzy formal context* $\mathcal{C}$*. A pair* $\langle \uparrow, \downarrow \rangle$ *forms a Galois connection between complete lattices of all* L*-sets of objects* L^B *and attributes* L^A*.*

Definition 5. *Let* $\mathcal{C} = \langle B, A, \mathcal{L}, r\rangle$ *be an* L*-fuzzy formal context. A* formal concept *is a pair of* L*-sets* $\langle f, g\rangle \in L^B \times L^A$ *such that* $\uparrow(f) = g$ *and* $\downarrow(g) = f$*. The set of all* L*-concepts of* $\mathcal{C}$ *will be denoted as* $\mathrm{FCL}(\mathcal{C})$*. The object (resp. attribute) part of any concept is called* extent *(resp.* intent*). The sets of all extents or intents of* $\mathcal{C}$ *will be denoted as* $\mathrm{Ext}(\mathcal{C})$ *or* $\mathrm{Int}(\mathcal{C})$*, respectively.*

2.3 L-Bonds and L-Chu correspondences

Definition 6. *Let* X *and* Y *be two sets. An* L-multifunction *from* X *to* Y *is said to be a mapping from* X *to* L^Y*.*

Definition 7. *Let* $\mathcal{C}_i = \langle B_i, A_i, \mathcal{L}, r_i\rangle$ *for* $i \in \{1, 2\}$ *be two* L*-fuzzy formal contexts. A pair of* L*-multifunctions* $\varphi = \langle\varphi_L, \varphi_R\rangle$ *such that*

- $\varphi_L\colon B_1 \longrightarrow \mathrm{Ext}(\mathcal{C}_2)$,
- $\varphi_R\colon A_2 \longrightarrow \mathrm{Int}(\mathcal{C}_1)$,

where $\uparrow_2(\varphi_L(o_1))(a_2) = \downarrow_1(\varphi_R(a_2))(o_1)$ *for any* $(o_1, a_2) \in B_1 \times A_2$*, is said to be an* L*-Chu correspondence between* $\mathcal{C}_1$ *and* $\mathcal{C}_2$*. A set of all* L*-Chu corresepondences between* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *will be denoted by* $L\text{-}\mathrm{ChuCors}(\mathcal{C}_1, \mathcal{C}_2)$*.*

Definition 8. *Let* $\mathcal{C}_i = \langle B_i, A_i, \mathcal{L}, r_i\rangle$ *for* $i \in \{1, 2\}$ *be two* L*-fuzzy formal contexts. An* L*-multifunction* $\beta\colon B_1 \longrightarrow \mathrm{Int}(\mathcal{C}_2)$*, such that* $\beta^{\mathrm{t}}\colon A_2 \longrightarrow \mathrm{Ext}(\mathcal{C}_1)$*, where* $\beta^{\mathrm{t}}(a_2)(o_1) = \beta(o_1)(a_2)$ *for any* $(o_1, a_2) \in B_1 \times A_2$*, is said to be an* L-bond*. A set of all* L*-bonds between* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *will be denoted by* $L\text{-}\mathrm{Bonds}(\mathcal{C}_1, \mathcal{C}_2)$*.*

Lemma 3. *Let* $\mathcal{C}_i = \langle B_i, A_i, \mathcal{L}, r_i\rangle$ *for* $i \in \{1, 2\}$ *be two* L*-fuzzy formal contexts. The sets* $L\text{-}\mathrm{Bonds}(\mathcal{C}_1, \mathcal{C}_2)$ *and* $L\text{-}\mathrm{ChuCors}(\mathcal{C}_1, \mathcal{C}_2)$ *form complete lattices and, moreover, there exists a dual isomorphism between them.*

3 Residuation-preserving isotone Galois connections

Definition 9. *An* isotone Galois connection *between two complete lattices* $\mathcal{L}_1 = (L_1, \leq_1)$ *and* $\mathcal{L}_2 = (L_2, \leq_2)$ *is a pair of monotone mappings* $\lambda = \langle\lambda_L, \lambda_R\rangle$ *with*

$$\lambda_L\colon L_1 \longrightarrow L_2 \qquad \text{and} \qquad \lambda_R\colon L_2 \longrightarrow L_1$$

such that, for any $k_1 \in L_1$ *and* $k_2 \in L_2$*, the following equivalence holds*

$$k_1 \leq_1 \lambda_R(k_2) \qquad \Longleftrightarrow \qquad \lambda_L(k_1) \leq_2 k_2. \tag{1}$$

The general theory of adjunctions provides the following result:

Lemma 4. *Let* $\langle\lambda_L, \lambda_R\rangle$ *be an isotone Galois connection, then for all* $k_1 \in L_1$ *and* $k_2 \in L_2$

$$\lambda_R(k_2) = \bigvee\{m \in L_1 : \lambda_L(m) \leq_2 k_2\} \tag{2}$$

$$\lambda_L(k_1) = \bigwedge\{m \in L_2 : k_1 \leq_1 \lambda_R(m)\} \tag{3}$$

Definition 10. *An isotone Galois connection λ between two complete residuated lattices $\mathcal{L}_1 = (L_1, \otimes_1, \to_1)$ and $\mathcal{L}_2 = (L_2, \otimes_2, \to_2)$ is said to be a* residuation-preserving isotone Galois connection *if for any $k_1, m_1 \in L_1$ and $k_2, m_2 \in L_2$ the following equalities hold:*

$$\lambda_L(k_1 \otimes_1 m_1) = \lambda_L(k_1) \otimes_2 \lambda_L(m_1) \tag{4}$$
$$\lambda_R(k_2 \otimes_2 m_2) = \lambda_R(k_2) \otimes_1 \lambda_R(m_2) \tag{5}$$
$$k_2 \to_2 \lambda_L(m_1) \geq_2 \lambda_L(\lambda_R(k_2) \to_1 m_1) \tag{6}$$

The set of all residuation-preserving isotone Galois connections from $\mathcal{L}_1$ to $\mathcal{L}_2$ will be denoted as $\mathrm{CRL}(\mathcal{L}_1, \mathcal{L}_2)$.

There is no need to consider other $\to$-preserving rules, since they follow from the previous ones, as stated by the following lemmas.

Lemma 5. *For all $k \in L_1$ and $m \in L_2$ the following equality holds*

$$k \to_1 \lambda_R(m) = \lambda_R(\lambda_L(k) \to_2 m) \tag{7}$$

Proof. Consider the following chain of equivalences

$$\begin{aligned} l \otimes_1 k \leq_1 \lambda_R(m) \quad &\overset{(1)}{\iff} \quad \lambda_L(l \otimes_1 k) \leq_2 m \\ &\overset{(4)}{\iff} \quad \lambda_L(l) \otimes_2 \lambda_L(k) \leq_2 m \\ &\overset{(\mathrm{adj})}{\iff} \quad \lambda_L(l) \leq_2 \lambda_L(k) \to_2 m \end{aligned}$$

As a result, we can write

$$\begin{aligned} k \to_1 \lambda_R(m) &= \bigvee \{ l \in L_1 : l \otimes_1 k \leq \lambda_R(m) \} \\ &= \bigvee \{ l \in L_1 : \lambda_L(l) \leq \lambda_L(k) \to_2 m \} \\ &\overset{(2)}{=} \lambda_R(\lambda_L(k) \to_2 m) \end{aligned}$$

It is worth to note that this proof does not work in the case of (6) because, for the construction of λ_L, one had to use (3) instead of (2). □

Lemma 6. *For all $k_i, m_i \in L_i$ for $i \in \{1, 2\}$, the following inequalities hold*

$$\lambda_L(k_1 \to_1 m_1) \leq_2 \lambda_L(k_1) \to_2 \lambda_L(m_1) \tag{8}$$
$$\lambda_R(k_2 \to_2 m_2) \leq_1 \lambda_R(k_2) \to_1 \lambda_R(m_2) \tag{9}$$

Proof. By the adjoint property and the following chain of inequalities

$$\lambda_L(k_1 \to_1 m_1) \otimes_2 \lambda_L(k_1) \overset{(4)}{=} \lambda_L((k_1 \to_1 m_1) \otimes_1 k_1) \leq_2 \lambda_L(m_1)$$

Similarly, we obtain the other one. □

Below, we recall the notion of fixpoint of a Galois connection, the definition is uniform to the different types of Galois connection, either antitone or isotone, or with any other extra requirement.

Definition 11. *Let λ be a Galois connection between complete residuated lattices $\mathcal{L}_1$ and $\mathcal{L}_2$. The set of all fixpoints of λ is defined as*

$$\mathrm{FP}_\lambda = \{\langle k_1, k_2\rangle \in L_1 \times L_2 : \lambda_L(k_1) = k_2, \lambda_R(k_2) = k_1\}.$$

Lemma 7. *Given $\lambda \in \mathrm{CRL}(\mathcal{L}_1, \mathcal{L}_2)$, the set of its fixpoints can be provided with the structure of complete residuated lattice $\Phi_\lambda = \langle \mathrm{FP}_\lambda, \wedge, \vee, 0, 1, \otimes, \to\rangle$ where $0 = \langle \lambda_R(0_2), 0_2\rangle$, $1 = \langle 1_1, \lambda_L(1_1)\rangle$, and $\otimes$ and $\to$ are defined componentwise.*

Proof. We have to check just that the componentwise operations provide a residuated structure to the set of fixed point of λ.

Conditions (4) and (5) allow to prove that componentwise product $\otimes$ is a closed operation in FP_λ, whereas condition (6) allows to prove that the componentwise implication is a closed operation in FP_λ.

It is not difficult to show that, in fact, $\langle \mathrm{FP}_\lambda, \otimes, 1\rangle$ is a commutative monoid: commutativity and associativity follow directly; for the neutral element just consider the following chain of equalities: For any $\langle k_1, k_2\rangle \in \mathrm{FP}_\lambda$ holds

$$\begin{aligned}\langle k_1, k_2\rangle \otimes \langle 1_1, \lambda_L(1_1)\rangle &= \langle k_1 \otimes_1 1_1, \lambda_L(k_1) \otimes_2 \lambda_L(1_1)\rangle \\ &= \langle k_1, \lambda_L(k_1 \otimes_1 1_1)\rangle \\ &= \langle k_1, \lambda_L(k_1)\rangle = \langle k_1, k_2\rangle\end{aligned}$$

The adjoint property follows by definition. □

4 CRL-Chu correspondences and their category

In this section, the notion of L-Chu correspondence is generalized on the basis of a residuation-preserving isotone Galois connection λ. The formal definition is the following:

Definition 12. *Let $\mathcal{C}_i = \langle B_i, A_i, \mathcal{L}_i, r_i\rangle$ for $i \in \{1, 2\}$ be two fuzzy formal contexts, and consider $\lambda \in \mathrm{CRL}(\mathcal{L}_1, \mathcal{L}_2)$. A pair of fuzzy multifunctions $\varphi = \langle \varphi_L, \varphi_R\rangle$ of types*

$$\varphi_L \colon B_1 \longrightarrow \mathrm{Ext}(\mathcal{C}_2) \qquad \textit{and} \qquad \varphi_R \colon A_2 \longrightarrow \mathrm{Int}(\mathcal{C}_1)$$

such that for any $(o_1, a_2) \in B_1 \times A_2$ the following inequality holds

$$\lambda_L(\downarrow_1(\varphi_R(a_2))(o_1)) \le_2 \uparrow_2(\varphi_L(o_1))(a_2) \tag{10}$$

is said to be a λ-Chu correspondence*.*

Note that (10) *is equivalent to* $\downarrow_1(\varphi_R(a_2))(o_1) \le_1 \lambda_R(\uparrow_2(\varphi_L(o_1))(a_2))$*.*

It is not difficult to check that the definition of λ-Chu correspondence generalizes the previous one based on a complete (residuated) lattice L; formally, we have the following

Definition 13. *Let X be an arbitrary set. Mapping id^X defined by $\mathrm{id}^X(x) = x$ for any $x \in X$ is said to be an identity mapping on X.*

Lemma 8. *Any L-Chu correspondence is a $\langle \mathrm{id}^L, \mathrm{id}^L \rangle$-Chu correspondence.*

We are now in position to define the category of parameterized fuzzy formal contexts and λ-Chu correspondences between them:

Definition 14. *We introduce a new category whose objects are parameterized fuzzy formal contexts $\langle B, A, \mathcal{L}, r \rangle$ and λ-Chu correspondences between them.*

The identity arrow *of an object $\langle B, A, \mathcal{L}, r \rangle$ is the $\langle \mathrm{id}^L, \mathrm{id}^L \rangle$-Chu correspondence ι such that*

- $\iota_L(o) = \downarrow\uparrow(\chi_o)$ *for any* $o \in B$
- $\iota_R(a) = \uparrow\downarrow(\chi_a)$ *for any* $a \in A$
- *where* $\chi_x(x) = 1$ *and* $\chi_x(y) = 0$ *for any* $y \neq x$.

Composition of arrows[3] $\langle \lambda, \varphi \rangle \colon \mathcal{C}_1 \to \mathcal{C}_2$ *and* $\langle \mu, \psi \rangle \colon \mathcal{C}_2 \to \mathcal{C}_3$ *where* $\mathcal{C}_i = \langle B_i, A_i, \mathcal{L}_i, r_i \rangle$ *for* $i \in \{1, 2, 3\}$ *is defined as:*

- $(\langle \mu, \psi \rangle \circ \langle \lambda, \varphi \rangle)_L(o_1) = \downarrow_3\uparrow_3(\psi_{L+}(\varphi_L(o_1)))$
- $(\langle \mu, \psi \rangle \circ \langle \lambda, \varphi \rangle)_R(a_3) = \uparrow_1\downarrow_1(\varphi_{R+}(\psi_R(a_3)))$

where, for any $(o_i, a_i) \in B_i \times A_i$, $i \in \{1, 3\}$,

$$\psi_{L+}(\varphi_L(o_1))(o_3) = \bigvee_{o_2 \in B_2} \psi(o_2)(o_3) \otimes_3 \mu_L(\varphi_L(o_1)(o_2))$$

$$\varphi_{R+}(\psi_R(a_3))(a_1) = \bigvee_{a_2 \in A_2} \varphi_R(a_2)(a_1) \otimes_1 \lambda_R(\psi_R(a_3)(a_2))$$

Obviously, one has to check that the proposed notions of composition and identity are well-defined, and this is stated in the following lemmas.

Lemma 9. *The identity arrow of any fuzzy formal context $\langle B, A, \mathcal{L}, r \rangle$ is a $\langle \mathrm{id}^L, \mathrm{id}^L \rangle$-Chu correspondence.*

Lemma 10. *Consider $\langle \lambda, \varphi \rangle : \mathcal{C}_1 \to \mathcal{C}_2$ and $\langle \mu, \psi \rangle : \mathcal{C}_2 \to \mathcal{C}_3$, then $\langle \mu, \psi \rangle \circ \langle \lambda, \varphi \rangle$ is a $(\mu \circ \lambda)$-Chu correspondence. Moreover, composition of λ-correspondences is associative.*

---

[3] Any λ-Chu correspondence φ can be conveniently denoted by $\langle \lambda, \varphi \rangle$.

5 λ-Bonds and λ-direct product of two contexts

We proceed with the corresponding extension of the notion of bond between contexts, and the study of its properties.

Definition 15. *Given a multifunction* $\omega\colon X \to (L_1 \times L_2)^Y$*, its projections* ω^i *for* $i \in \{1,2\}$ *are defined by* $\omega^i(x)(y) = k_i$*, provided that* $\omega(x)(y) = (k_1,k_2)$*. Transposition of such multifunction is defined by* $\omega^{\mathrm{t}}(y)(x) = \omega(x)(y)$*.*

Definition 16. *Given two fuzzy formal contexts* $\mathcal{C}_i = \langle B_i, A_i, \mathcal{L}_i, r_i\rangle$*,* $i \in \{1,2\}$*, and* $\lambda \in \mathrm{CRL}(\mathcal{L}_1,\mathcal{L}_2)$*. A* λ*-bond is a multifunction* $\beta\colon B_1 \to (L_1 \times L_2)^{A_2}$ *such that, for any* $(o_1,a_2) \in B_1 \times A_2$*:*

$$\beta^2(o_1) \text{ is an intent of } \mathcal{C}_2 \tag{11}$$

$$(\beta^{\mathrm{t}})^1(a_2) \text{ is an extent of } \mathcal{C}_1 \tag{12}$$

$$\beta^1(o_1)(a_2) \leq_1 \lambda_R(\beta^2(o_1)(a_2)) \text{ or equivalently } \lambda_L(\beta^1(o_1)(a_2)) \leq_2 \beta^2(o_1)(a_2) \tag{13}$$

The known relation between L-bonds and L-Chu correspondences is preserved in the λ-case. Formally,

Lemma 11. *Let* β *be a* λ*-bond between two fuzzy contexts* $\mathcal{C}_i = \langle B_i, A_i, \mathcal{L}_i, r_i\rangle$ *for* $i \in \{1,2\}$*. Then* φ_β *defined as*

$$\varphi_{\beta L}(o_1) = \ \downarrow_2(\beta^2(o_1)) \tag{14}$$

$$\varphi_{\beta R}(a_2) = \ \uparrow_1((\beta^{\mathrm{t}})^1(a_2)) \tag{15}$$

is λ*-Chu correspondence.*

Proof. By calculation

$$\begin{aligned}
\uparrow_2(\varphi_{\beta L}(o_1))(a_2) &\overset{(14)}{=} \uparrow_2\downarrow_2(\beta^2(o_1))(a_2) \overset{(11)}{=} \beta^2(o_1)(a_2)\\
&\overset{(13)}{\geq} \lambda_L(\beta^1(o_1)(a_2)) = \lambda_L((\beta^{\mathrm{t}})^1(a_2)(o_1))\\
&\overset{(12)}{=} \lambda_L(\downarrow_1\uparrow_1((\beta^{\mathrm{t}})^1(a_2))(o_1))\\
&\overset{(15)}{=} \lambda_L(\downarrow_1(\varphi_{\beta R}(a_2))(o_1))
\end{aligned}$$

□

Lemma 12. *Let* $\mathcal{L}_1, \mathcal{L}_2$ *be two complete residuated lattices satisfying the double negation law and let* $\lambda \in \mathrm{CRL}(\mathcal{L}_1,\mathcal{L}_2)$*. Then* Φ_λ *satisfies double negation law.*

Proof. Consider an arbitrary $\langle k_1,k_2\rangle \in \mathrm{FP}_\lambda$. We have that, by definition,

$$\begin{aligned}
\neg\neg\langle k_1,k_2\rangle &= (\langle k_1,k_2\rangle \to 0) \to 0 = (\langle k_1,k_2\rangle \to \langle\lambda_R(0_2),0_2\rangle) \to \langle\lambda_R(0_2),0_2\rangle\\
&= \langle(k_1 \to_1 \lambda_R(0_2)) \to_1 \lambda_R(0_2), (k_2 \to_2 0_2) \to_2 0_2\rangle
\end{aligned}$$

The result for the second component is obvious; for the first one, taking into account that $(\lambda_R(0_2), 0_2)$ is a fixed point, we have

$$\begin{aligned}
\lambda_L(k_1) = k_2 &= (k_2 \to_2 0_2) \to_2 0_2 \\
&= \big(k_2 \to_2 \lambda_L\lambda_R(0_2)\big) \to_2 \lambda_L\lambda_R(0_2) \\
&\overset{(6)}{\geq_2} \lambda_L(\lambda_R\lambda_L(\lambda_R(k_2) \to_1 \lambda_R(0_2)) \to_1 \lambda_R(0_2)) \\
&\overset{(\star)}{=} \lambda_L((\lambda_R(k_2) \to_1 \lambda_R(0_2)) \to_1 \lambda_R(0_2)) \\
&= \lambda_L((k_1 \to_1 \lambda_R(0_2)) \to_1 \lambda_R(0_2)) \\
&\overset{(*)}{\geq_2} \lambda_L(k_1)
\end{aligned}$$

where equality $(\star)$ follows because, by Lemma 7, $\lambda_R(k_2) \to_1 \lambda_R(0_2)$ is a closed value in L_1 for the composition $\lambda_R\lambda_L$, and inequality $(*)$ follows from the monotonicity of λ_L.

As a result of the previous chain we obtain the following equality

$$\lambda_L(k_1) = \lambda_L((k_1 \to_1 \lambda_R(0_2)) \to_1 \lambda_R(0_2))$$

and, again by Lemma 7, since k_1 and $\lambda_R(0_2)$ are closed for $\lambda_R\lambda_L$, as a result $(k_1 \to_1 \lambda_R(0_2)) \to_1 \lambda_R(0_2)$ is closed too, and $k_1 = (k_1 \to_1 \lambda_R(0_2)) \to_1 \lambda_R(0_2)$. □

We are now ready to include the characterization result on the structure of λ-bonds, but we have to introduce the notion of λ-direct product of contexts.

Definition 17. *Let $\mathcal{C}_i = \langle B_i, A_i, \mathcal{L}_i, r_i\rangle$ for $i \in \{1,2\}$ be two fuzzy formal contexts, $\lambda \in \mathrm{CRL}(\mathcal{L}_1, \mathcal{L}_2)$ and $\mathcal{L}_1$, $\mathcal{L}_2$ satisfy the double negation law. The fuzzy formal context $\langle B_1 \times A_2, B_2 \times A_1, \Phi_\lambda, \Delta_\lambda\rangle$ where $\Delta_\lambda((o_1, a_2), (o_2, a_1)) = \neg(\overline{\lambda_1}(r_1)) \to \overline{\lambda_2}(r_2(o_2, a_2))$ is said to be the* λ-direct product *of $\mathcal{C}_1$ and $\mathcal{C}_2$, where*

$$\overline{\lambda_1}(k) = \langle \lambda_R\lambda_L(k), \lambda_L(k)\rangle \text{ for all } k \in L_1 \tag{16}$$

$$\overline{\lambda_2}(k) = \langle \lambda_R(k), \lambda_L\lambda_R(k)\rangle \text{ for all } k \in L_2 \tag{17}$$

Lemma 13. *Let $\mathcal{C}_1\Delta_\lambda\mathcal{C}_2$ be the λ-direct product of fuzzy formal contexts $\mathcal{C}_1$ and $\mathcal{C}_2$, and $\lambda \in \mathrm{CRL}(\mathcal{L}_1, \mathcal{L}_2)$. For any extent of $\mathcal{C}_1\Delta_\lambda\mathcal{C}_2$ there exists a λ-bond between $\mathcal{C}_1$ and $\mathcal{C}_2$.*

Proof. Let $\langle\beta, \gamma\rangle$ be a concept of $\mathcal{C}_1\Delta_\lambda\mathcal{C}_2$. Then $\beta \in \mathrm{FP}_\lambda^{B_1\times A_2} \subseteq (L_1 \times L_2)^{B_1\times A_2}$.

$$\begin{aligned}
\beta(o_1, a_2) &= \downarrow_{\Delta_\lambda}(\gamma)(o_1, a_2) \\
&= \bigwedge_{o_2\in B_2}\bigwedge_{a_1\in A_1}(\gamma(o_2, a_1) \to \Delta_\lambda((o_1, a_2), (o_2, a_1))) \\
&= \bigwedge_{o_2\in B_2}\bigwedge_{a_1\in A_1}(\gamma(o_2, a_1) \to (\neg\overline{\lambda_1}(r_1(o_1, a_1))) \to \overline{\lambda_2}(r_2(o_2, a_2)))
\end{aligned}$$

Then

$$\begin{aligned}
\beta^1(o_1,a_2) &= \bigwedge_{o_2\in B_2}\ \bigwedge_{a_1\in A_1} (\gamma^1(o_2,a_1) \rightarrow_1 (\neg\lambda_R\lambda_L(r_1(o_1,a_1))) \rightarrow_1 \lambda_R(r_2(o_2,a_2))) \\
&= \bigwedge_{o_2\in B_2}\ \bigwedge_{a_1\in A_1} ((\gamma^1(o_2,a_1) \otimes_1 \neg\lambda_R\lambda_L(r_1(o_1,a_1))) \rightarrow_1 \lambda_R(r_2(o_2,a_2))) \\
&\overset{(7)}{=} \bigwedge_{o_2\in B_2}\ \bigwedge_{a_1\in A_1} \lambda_R(\lambda_L(\gamma^1(o_2,a_1) \otimes_1 \neg\lambda_R\lambda_L(r_1(o_1,a_1))) \rightarrow_2 r_2(o_2,a_2)) \\
&= \lambda_R(\bigwedge_{o_2\in B_2}\ \bigwedge_{a_1\in A_1} (\lambda_L(\gamma^1(o_2,a_1) \otimes_1 \neg\lambda_R\lambda_L(r_1(o_1,a_1))) \rightarrow_2 r_2(o_2,a_2))) \\
&= \lambda_R(\bigwedge_{o_2\in B_2} (\bigvee_{a_1\in A_1} (\lambda_L(\gamma^1(o_2,a_1) \otimes_1 \neg\lambda_R\lambda_L(r_1(o_1,a_1))))) \rightarrow_2 r_2(o_2,a_2))) \\
&= \lambda_R(\bigwedge_{o_2\in B_2} (\sigma(o_1)(o_2) \rightarrow_2 r_2(o_2,a_2))) \\
&= \lambda_R(\uparrow_2 (\sigma(o_1))(a_2))
\end{aligned}$$

Similarly

$$\begin{aligned}
\beta^2(o_1,a_2) &= \bigwedge_{o_2\in B_2}\ \bigwedge_{a_1\in A_1} ((\gamma^2(o_2,a_1) \otimes_2 \neg\lambda_L\lambda_R(r_2(o_2,a_2))) \rightarrow_2 \lambda_L(r_1(o_1,a_1))) \\
&\overset{(6)}{\geq}_2 \bigwedge_{o_2\in B_2}\ \bigwedge_{a_1\in A_1} \lambda_L(\lambda_R(\gamma^2(o_2,a_1) \otimes_2 \neg\lambda_L\lambda_R(r_2(o_2,a_2))) \rightarrow_2 r_1(o_1,a_1)) \\
&\geq_2 \lambda_L(\bigwedge_{o_2\in B_2}\ \bigwedge_{a_1\in A_1} (\lambda_R(\gamma^2(o_2,a_1) \otimes_2 \neg\lambda_L\lambda_R(r_2(o_2,a_2))) \rightarrow_1 r_1(o_1,a_1))) \\
&= \lambda_L(\bigwedge_{a_1\in A_1} (\bigvee_{o_2\in B_2} (\lambda_R(\gamma^2(o_2,a_1) \otimes_2 \neg\lambda_L\lambda_R(r_2(o_2,a_2))))) \rightarrow_1 r_1(o_1,a_1))) \\
&= \lambda_L(\bigwedge_{a_1\in A_1} (\tau(a_2)(a_1) \rightarrow_1 r_1(o_1,a_1))) = \lambda_L(\downarrow_1 (\tau(a_2))(o_1))
\end{aligned}$$

Then let us define a multifunction $\widehat{\beta}\colon B_1 \longrightarrow (L_1 \times L_2)^{A_2}$ as follows

$$\begin{aligned}
\widehat{\beta}^2(o_1) &= \uparrow_2 (\sigma(o_1)) \\
(\widehat{\beta}^{\mathrm{t}})^1(a_2) &= \downarrow_1 (\tau(a_2))
\end{aligned}$$

where σ and τ are multifunctions above. We see that $\widehat{\beta}^2(o_1)$ is the intent of $\mathcal{C}_2$, $(\widehat{\beta}^{\mathrm{t}})^1(a_2)$ is the extent of $\mathcal{C}_1$ and moreover $\beta(o_1,a_2) \in \mathrm{FP}_\lambda$, hence $\lambda_L(\beta^1(o_1,a_2)) = \beta^2(o_1,a_2)$ and

$$\begin{aligned}
\lambda_L((\widehat{\beta}^{\mathrm{t}})^1(a_2)(o_1)) &= \lambda_L(\downarrow_1 (\tau(a_2))) \\
&\leq_2 \beta^2(o_1,a_2) \\
&= \lambda_L(\beta^1(o_1,a_2)) \\
&= \lambda_L\lambda_R(\uparrow_2 (\sigma(o_1))(a_2)) \\
&\leq_2 \uparrow_2 (\sigma(o_1))(a_2) = \widehat{\beta}^2(o_1)(a_2)
\end{aligned}$$

Therefore $\widehat{\beta}$ is a λ-bond between $\mathcal{C}_1$ and $\mathcal{C}_2$. $\square$

6 Motivation example

Lets have two tables of the following data. First table of students, school subjects and study results. Second table of universities (or areas of study) and their requirements for results of students. We would like to find the assignment of students and universities that depends on study results and requirements of universities.

$\mathcal{S}$	Math	Phi	Chem	Bio
Anna	A	A	B	C
Boris	C	B	A	B
Cyril	D	E	B	C

$\mathcal{U}$	CS	Tech	Med
Math	Ex	G	W
Phi	G	Ex	G
Chem	W	Ex	Ex
Bio	W	W	Ex

First table is filled by degrees from well known structure {A,B,C,D,E,F} where A is best and F means failed. Second one is filled by degrees {Ex,G,W} that means Ex-excelent, G-good, W-weak. Now lets define a λ-translation between such truth-degrees structures.

	A	B	C	D	E	F
$\lambda_L(-)$	Ex	G	W	W	W	W

	Ex	G	W
$\lambda_R(-)$	A	B	C

$\langle \lambda_L, \lambda_R \rangle$ is an isotone Galois connection. In fact, $\langle \lambda_L, \lambda_R \rangle$ is a residuation-preserving isotone Galois connection over Łukasiewicz logic, whose set of fix-points is

$$\mathrm{FP}_\lambda = \{(\mathrm{A}, \mathrm{Ex}); (\mathrm{B}, \mathrm{G}); (\mathrm{C}, \mathrm{W})\}$$

The λ-direct product $\mathcal{S}\Delta_\lambda\mathcal{U}$ is the following table that has 510 concepts. Lets simplify the table with translation (A,Ex) as 1, (B,G) as 0.5 and (C,W) as 0.

1	1	0.5	0	1	1	1	0.5	1	1	1	1	1	1	1	1
1	1	1	0.5	1	1	1	1	1	1	1	1	1	1	0.5	0
1	1	1	1	1	1	1	0.5	1	1	0.5	0	1	1	0.5	0
0	0.5	1	0.5	0.5	1	1	1	1	1	1	1	1	1	1	1
0.5	1	1	1	1	1	1	1	1	1	1	1	0	0.5	1	0.5
1	1	1	1	0.5	1	1	1	0	0.5	1	0.5	0	0.5	1	0.5
0	0	0.5	0	0.5	0.5	1	0.5	1	1	1	1	1	1	1	1
0.5	0.5	1	0.5	1	1	1	1	1	1	1	1	0	0	0.5	0
1	1	1	1	0.5	0.5	1	0.5	0	0	0.5	0	0	0	0.5	0

Extents of $\mathcal{S}\Delta_\lambda\mathcal{U}$ are tables of the form students×universities and intents are tables of the form subjects×subjects. One of the concepts that their intents has 1 on diagonal (it means that any subject is assigned to itself) is shown below.

	Med	Tech	CS
Anna	B,G	B,G	A,Ex
Boris	A,Ex	B,G	C,W
Cyril	B,G	B,G	C,W

	Math	Phi	Chem	Bio
Math	A,Ex	A,Ex	B,G	C,W
Phi	B,G	A,Ex	A,Ex	B,G
Chem	C,W	B,G	A,Ex	B,G
Bio	C,W	B,G	A,Ex	B,G

Such concept should be translated into $\{1; 0.5; 0\}$ structure.

	Med	Tech	CS
Anna	0.5	0.5	1
Boris	1	0.5	0
Cyril	0.5	0.5	0

	Math	Phi	Chem	Bio
Math	1	1	0.5	0
Phi	0.5	1	1	0.5
Chem	0	0.5	1	0.5
Bio	0	0.5	1	0.5

Now we can see the result. Due to results of students and requirements of universities we can advise Anna to study Computer science; similarly, we can advise Boris to study Medicine or Technical area; finally, it is hard to advise anything to Cyril. The assignment is right as it is obvious from study results.

Relaxing the connection

Let's change the λ-connection between such truth degrees structures as follows:

	A	B	C	D	E	F
$\lambda_L(-)$	Ex	G	G	W	W	W

	Ex	G	W
$\lambda_R(-)$	A	B	D

The set of fixpoints is $\mathrm{FP}_\lambda = \{(\mathrm{A}, \mathrm{Ex}); (\mathrm{B}, \mathrm{G}); (\mathrm{D}, \mathrm{W})\}$ such that is easy to translate $(\mathrm{A}, \mathrm{Ex})$ as 1, (B, G) as 0.5 and (D, W) as 0. The direct product is shown below, and has 104 concepts.

1	1	1	1	1	1	1	1	1	1	0.5	0.5	1	1	0.5	0.5
1	1	1	1	1	1	1	1	1	1	1	1	1	1	0.5	0.5
1	1	0.5	0.5	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	0.5	0.5	1	0.5	0.5	0.5	1	0.5
1	1	1	1	1	1	1	1	1	1	1	1	0.5	0.5	1	0.5
0.5	0.5	1	0.5	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	0.5	0.5	1	1	0	0	0.5	0.5	0	0	0.5	0.5
0.5	0.5	1	1	1	1	1	1	1	1	1	1	0	0	0.5	0.5
0	0	0.5	0.5	0.5	0.5	1	1	1	1	1	1	1	1	1	1

We have chosen one with 1-diagonal in the intent.

	Med	Tech	CS
Anna	B,G	A,Ex	A,Ex
Boris	A,Ex	A,Ex	B,G
Cyril	B,G	B,G	D,W

	Math	Phi	Chem	Bio
Math	A,Ex	A,Ex	B,G	B,G
Phi	A,Ex	A,Ex	A,Ex	A,Ex
Chem	B,G	B,G	A,Ex	B,G
Bio	B,G	B,G	B,G	B,G

Translating the context into $\{1; 0.5; 0\}$ structure, we obtain

	Med	Tech	CS
Anna	0.5	1	1
Boris	1	1	0.5
Cyril	0.5	0.5	0

	Math	Phi	Chem	Bio
Math	1	1	0.5	0.5
Phi	1	1	1	0.5
Chem	0.5	0.5	1	0.5
Bio	0.5	0.5	0.5	0.5

It can seen that our advice is more generous but still coincide to input data.

7 Conclusion

We continue our study of the general notion of L-Chu correspondence by introducing the category CRL-ChuCors incorporating residuation to the underlying complete lattice L, specifically, on the basis of a residuation-preserving isotone Galois connection λ. Then, the L-bonds are generalized within this same framework, and its structure is related to that of the extent of a suitably defined λ-direct product. A first relationship between extents of λ-direct product have been proved; it is expected to find a proof of the stronger result which states an isomorphism between the extents of the λ-direct product and the λ-bonds between $\mathcal{C}_1$ and $\mathcal{C}_2$.

Potential applications are primary motivations for further future work, for instance, to consider possible classes of formal L-contexts induced from existing datamining notions, and study its associated categories.

References

1. J. T. Denniston, A. Melton, and S. E. Rodabaugh. Formal concept analysis and lattice-valued Chu systems. *Fuzzy Sets and Systems*, 216:52–90, 2013.
2. P. Hitzler and G.-Q. Zhang. A cartesian closed category of approximable concept structures. *Lecture Notes in Computer Science*, 3127:170–185, 2004.
3. S. Krajči. A categorical view at generalized concept lattices. *Kybernetika*, 43(2):255–264, 2007.
4. 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.
5. O. Krídlo and M. Ojeda-Aciego. On L-fuzzy Chu correspondences. *Intl J of Computer Mathematics*, 88(9):1808–1818, 2011.
6. O. Krídlo and M. Ojeda-Aciego. Linking L-Chu Correspondences and Completely Lattice L-ordered Sets. Proceedings of the Intl Conf. on Concept Lattices and its Applications (CLA'12), pp 233–244, 2012
7. M. Krötzsch, P. Hitzler, and G.-Q. Zhang. Morphisms in context. *Lecture Notes in Computer Science*, 3596:223–237, 2005.
8. H. Mori. Chu correspondences. *Hokkaido Mathematical Journal*, 37:147–214, 2008.
9. S. Solovyov. Lattice-valued topological systems as a framework for lattice-valued formal concept analysis. *Journal of Mathematics*, 2013. To appear.
10. G.-Q. Zhang. Chu spaces, concept lattices, and domains. *Electronic Notes in Theoretical Computer Science*, 83, 2004.
11. G.-Q. Zhang and G. Shen. Approximable concepts, Chu spaces, and information systems. *Theory and Applications of Categories*, 17(5):80–102, 2006.

Formal Concept Analysis of higher order

Ondrej Krídlo, Patrik Mihalčin, Stanislav Krajči and Lubomír Antoni

University of Pavol Jozef Šafárik, Košice, Slovakia*

Abstract. The second order formal context is a formal context such that its object and attribute sets are disjoint unions of object and attribute sets of external formal contexts. Every subset of object or attribute set will be evaluated from concept lattice of corresponding external formal context. The paper provides a method how to compute such second order formal concepts by using of bonds between external formal contexts or by using of heterogeneous formal contexts methods. Last part of the paper shows how this structure generalizes homogenic fuzzy formal context and its derivation operators.

1 Motivation example

Imagine the following situation as a motivation. Lets have a group of people that everybody knows each other (schoolmates, co-workers, etc.). All of them are going to travel somewhere together and you (as an organizer of the trip) would like to know their requirements for accommodation. Consider the following formal context. Set of objects represents a group of co-workers (Anna, Bob, Cyril, David, Erik). Set of attributes expresses their requirements (TV, Wellness, Closeness to a city center, Restaurant in a hotel). An example of such context is in the following table. Lets denote the following table as $\mathcal{P}$ as preferences.

$\mathcal{P}$	TV	W	Ce	R
Anna	●		●	●
Bob	○	○	●	●
Cyril	○	●	●	○
David	●	●	○	
Erik	●	●	○	●

A particular formal concept of the given context describes a set of co-workers such that these people together have a common requirements for accommodation.

In addition, there are another two formal contexts. The first one describes a friendship relation inside the group of such people (denoted as $\mathcal{F}$). The second one describes a situation about hotels and services they offer (denoted as $\mathcal{H}$).

---

[*] Partially supported by grant VEGA 1/0832/12 and APVV-0035-10.

© paper author(s), 2013. Published in Manuel Ojeda-Aciego, Jan Outrata (Eds.): CLA 2013, pp. 117–128, ISBN 978–2–7466–6566–8, Laboratory L3i, University of La Rochelle, 2013. Copying permitted only for private and academic purposes.

$\mathcal{F}$	Anna	Bob	Cyril	David	Erik
Anna	●	○	○		
Bob	○	●	●	○	○
Cyril	○	●	●	●	●
David		○	●	●	●
Erik		○	●	●	●

$\mathcal{H}$	TV	W	C	R
H1	●	●	●	●
H2	●	●		●
H3			●	
H4	●		●	●

All contexts are filled by truth degrees from the following set $\{\bullet = \text{true}, \circ = \text{middle}, \text{" "} = \text{false}\}$. Computing of L-concepts is based on Łukasiewicz logic.

Now, we aim at connecting such table data with known intercontextual mechanisms in order to obtain closed sets of friends from $\mathcal{F}$ that are able to stay in any of a closed set of hotels from $\mathcal{H}$. The hotels of this closed set offer as much requirements from $\mathcal{P}$ as it gets.

2 Preliminaries

2.1 Basics

Formal Concept Analysis (FCA) as an applied Lattice Theory [7] has become a very useful tool for discovering of hidden knowledge inside a data of object-attribute table, so called formal contexts. Fundamental construction of FCA is a Galois connection between complete lattices of all subsets of objects and attributes. A Galois connection consists of two mappings such that a composition of these mappings form a closure operators on each subsets of complete lattice. Pair of closed subset of objects and subset of attributes connected to each other by the Galois connection is called formal concept. The set of formal concepts forms a complete lattice. The mentioned notions were generalized over a fuzzy logic based on a complete residuated lattice. The notions of order, Galois connection and complete lattice were also generalized by Bělohlávek in [3–6].

Definition 1. Complete residuated lattice *is an algebra* $\langle L, \wedge, \vee, 0, 1, \otimes, \to \rangle$, *where*

- $\langle L, \wedge, \vee, 0, 1 \rangle$ *is a complete lattice with top* 1 *and bottom* 0,
- $\langle L, \otimes, 1 \rangle$ *is a commutative monoid,*
- $\langle \otimes, \to \rangle$ *is an adjoint pair, i.e.*

$$a \otimes b \leq c \text{ is equivalent to } a \leq b \to c$$

for any $a, b, c \in L$.

Definition 2. L-fuzzy formal context $\mathcal{C}$ *is a triple* $\langle B, A, r \rangle$, *where* $r : B \times A \to L$ *is an L-fuzzy binary relation and L is the complete residuated lattice.*

Definition 3. *Let* $\langle B, A, r \rangle$ *be an L-fuzzy formal context. Lets define a pair of* derivation operators $\langle \uparrow, \downarrow \rangle$ *of the form* $\uparrow : L^B \longrightarrow L^A$ *and* $\downarrow : L^A \longrightarrow L^B$, *where*

$$\uparrow(f)(a) = \bigwedge_{b \in B} (f(b) \to r(b, a)) \text{ for any } f \in L^B \text{ and } a \in A,$$

$$\downarrow(g)(b) = \bigwedge_{a \in A} (g(a) \to r(b, a)) \text{ for any } g \in L^A \text{ and } b \in B.$$

Lemma 1. *Let* $\langle\uparrow,\downarrow\rangle$ *be a pair of derivation operators defined on an* L*-fuzzy formal context* $\langle B, A, r\rangle$*. A pair* $\langle\uparrow,\downarrow\rangle$ *forms a Galois connection between complete lattices of all* L*-sets of objects* L^B *and attributes* L^A*.*

Definition 4. *Let* $\mathcal{C} = \langle B, A, r\rangle$ *be an* L*-fuzzy formal context.* Formal concept *is a pair of* L*-sets* $\langle f, g\rangle \in L^B \times L^A$ *such that* $\uparrow(f) = g$ *and* $\downarrow(g) = f$*. The set of all* L*-concepts of* $\mathcal{C}$ *will be denoted by* $\mathrm{FCL}(\mathcal{C})$*. Object or attribute part of any concept is called* extent *or* intent*. Sets of all extents or intents of* $\mathcal{C}$ *will be denoted as* $\mathrm{Ext}(\mathcal{C})$ *or* $\mathrm{Int}(\mathcal{C})$*, respectively.*

2.2 Bonds and Chu correspondences

FCA provides the useful methods how to connect two formal contexts. A structure of the so called Chu correspondence was introduced by Mori [13, 14] that is very close to the notion of bond [7]. The notions of Chu correspondence and bond were extended into L-fuzzy Chu correspondence and L-bond in [8]. The corresponding notions are introduced now.

Definition 5. *Let* $\mathcal{C}_i = \langle B_i, A_i, r_i\rangle$ *for* $i \in \{1, 2\}$ *be two* L*-fuzzy formal contexts. Pair of* L*-multimappings* $\varphi = \langle\varphi_L, \varphi_R\rangle$ *such that*

- $\varphi_L : B_1 \longrightarrow \mathrm{Ext}(\mathcal{C}_2)$,
- $\varphi_R : A_2 \longrightarrow \mathrm{Int}(\mathcal{C}_1)$,

where $\uparrow_2(\varphi_L(o_1))(a_2) = \downarrow_1(\varphi_R(a_2))(o_1)$ *for any* $(o_1, a_2) \in B_1 \times A_2$*, is said to be an* L*-Chu correspondence between* $\mathcal{C}_1$ *and* $\mathcal{C}_2$*. Set of all* L*-Chu correspondences between* L*-contexts* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *will be denoted by* $L\text{-}\mathrm{ChuCors}(\mathcal{C}_1, \mathcal{C}_2)$*.*

Definition 6. *Let* $\mathcal{C}_i = \langle B_i, A_i, r_i\rangle$ *for* $i \in \{1, 2\}$ *be two* L*-fuzzy formal contexts.* L*-multimapping* $\beta : B_1 \longrightarrow \mathrm{Int}(\mathcal{C}_2)$*, such that* $\beta^{\mathrm{t}} : A_2 \longrightarrow \mathrm{Ext}(\mathcal{C}_1)$*, where* $\beta^{\mathrm{t}}(a_2)(o_1) = \beta(o_1)(a_2)$ *for any* $(o_1, a_2) \in B_1 \times A_2$*, is said to be an* L-bond*. Set of all* L*-bonds beyween* L*-contexts* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *will be denoted by* $L\text{-}\mathrm{Bonds}(\mathcal{C}_1, \mathcal{C}_2)$*.*

Lemma 2. *Let* $\mathcal{C}_i = \langle B_i, A_i, r_i\rangle$ *for* $i \in \{1, 2\}$ *be two* L*-fuzzy formal contexts. Each set* $L\text{-}\mathrm{Bonds}(\mathcal{C}_1, \mathcal{C}_2)$ *and* $L\text{-}\mathrm{ChuCors}(\mathcal{C}_1, \mathcal{C}_2)$ *forms a complete lattice and, moreover, there exists a dual isomorphism between them.*

The dual isomorphism between bonds and Chu correspondences is based on the following construction. Consider two L-fuzzy formal contexts $\mathcal{C}_i = \langle B_i, A_i, r_i\rangle$ for $i \in \{1, 2\}$ and let $\beta \in L\text{-}\mathrm{Bonds}(\mathcal{C}_1, \mathcal{C}_2)$, then $\langle\varphi_{\beta L}, \varphi_{\beta R}\rangle$ such that for any $(o_1, a_2) \in B_1 \times A_2$

$$\varphi_{\beta L}(o_1) = \downarrow_2(\beta(o_1)) \text{ and } \varphi_{\beta R}(a_2) = \uparrow_1(\beta^{\mathrm{t}}(a_2))$$

is an L-Chu correspondence from $L\text{-}\mathrm{ChuCors}(\mathcal{C}_1, \mathcal{C}_2)$.

On the other hand, let $\varphi \in L\text{-}\mathrm{ChuCors}(\mathcal{C}_1, \mathcal{C}_2)$. Then β_φ defined as

$$\beta_\varphi(o_1)(a_2) = \downarrow_1(\varphi_R(a_2))(o_1) = \uparrow_2(\varphi_L(o_1))(a_2)$$

for any $(o_1, a_2) \in B_1 \times A_2$ is an L-bond from $L\text{-}\mathrm{Bonds}(\mathcal{C}_1, \mathcal{C}_2)$.

2.3 Categorical relationship to fuzzy Galois connection

Categories ChuCors and L-ChuCors of classical or fuzzy formal contexts and classical or fuzzy Chu correspondences are described in [9, 13]. Important categorical property of $*$-autonomism is also proved in mentioned papers. The continuation of categorical research in [10] resulted in a categorical equivalence of L-ChuCors and a category L-CLLOS of so called completely lattice L-ordered sets and monotone fuzzy Galois connections. Equivalence is proved by constructing of equivalence functor between these categories.

Definition 7. *Lets define a functor* $\Gamma : L\text{-ChuCors} \longrightarrow L\text{-CLLOS}$ *in the following way:*

1. $\Gamma(\mathcal{C}) = \langle\langle L\text{-FCL}(\mathcal{C}), \approx\rangle, \preceq\rangle$ *for any* L*-context* $\mathcal{C}$
2. $\Gamma(\varphi) = \langle\lambda_L^\varphi, \lambda_R^\varphi\rangle$ *for any* $\varphi \in L\text{-ChuCors}(\mathcal{C}_1, \mathcal{C}_2)$ *such that* $\lambda_L^\varphi : \mathrm{FCL}(\mathcal{C}_1) \longrightarrow \mathrm{FCL}(\mathcal{C}_2)$ *and* $\lambda_R^\varphi : \mathrm{FCL}(\mathcal{C}_2) \longrightarrow \mathrm{FCL}(\mathcal{C}_1)$

$$\lambda_L^\varphi(\langle f, \uparrow_1(f)\rangle) = \langle \downarrow_2\uparrow_2(\varphi_{L+}(f)), \uparrow_2(\varphi_{L+}(f))\rangle$$
$$\lambda_R^\varphi(\langle \downarrow_2(g), g\rangle) = \langle \downarrow_1(\varphi_{R+}(g)), \uparrow_1\downarrow_1(\varphi_{R+}(g))\rangle$$

for any two L*-concepts* $\langle f, \uparrow_1(f)\rangle \in \mathrm{FCL}(\mathcal{C}_1)$ *and* $\langle \downarrow_2(g), g\rangle \in \mathrm{FCL}(\mathcal{C}_2)$*, where for any multifunction* $\omega : X \longrightarrow L^Y$ *is* $\omega_+ : L^X \longrightarrow L^Y$ *defined as* $\omega_+(f)(y) = \bigvee_{x\in X} f(x) \otimes \omega(x)(y)$ *for any* $f \in L^X$ *and* $y \in Y$*.*

In [10] is proved that Γ is the equivalence functor. Hence, it holds for the particular two L-concepts that

$$\Big(\langle f, \uparrow_1(f)\rangle \preceq_1 \lambda_R^\varphi(\langle \downarrow_2(g), g\rangle)\Big) = \Big(\lambda_L^\varphi(\langle f, \uparrow_1(f)\rangle) \preceq_2 \langle \downarrow_2(g), g\rangle\Big).$$

Lemma 3. *Consider two* L*-contexts* $\mathcal{C}_i = \langle B_i, A_i, r_i\rangle$ *for* $i \in \{1, 2\}$ *and let* $\varphi \in L\text{-ChuCors}(\mathcal{C}_1, \mathcal{C}_2)$*. A functor* $\Gamma(\varphi)$ *is a fuzzy Galois connection between* $\langle\langle \mathrm{FCL}(\mathcal{C}_1), \approx_1\rangle, \preceq_1\rangle$ *and* $\langle\langle \mathrm{FCL}(\mathcal{C}_2^*), \approx_2\rangle, \preceq_2\rangle$ *where* $\mathcal{C}_2^* = \langle A_2, B_2, r_2^{\mathrm{t}}\rangle$*.*

Proof. Due to order reversing of dual L-context $\mathcal{C}_2^*$ for any two L-concepts we obtain $\Big(\langle f, \uparrow_1(f)\rangle \preceq_1 \lambda_R^\varphi(\langle \downarrow_2(g), g\rangle)\Big) = \Big(\langle \downarrow_2(g), g\rangle \preceq_2 \lambda_L^\varphi(\langle f, \uparrow_1(f)\rangle)\Big)$. □

3 Formal concept analysis of second order

Once we have introduced preliminaries, the formal context of second order and the corresponding results are presented now in details.

Definition 8. *Consider two non-empty index sets* I *and* J *and an* L*-fuzzy formal context* $\langle \bigcup_{i\in I} B_i, \bigcup_{j\in J} A_j, r\rangle$*, whereby*

- $B_{i_1} \cap B_{i_2} = \emptyset$ *for any* $i_1, i_2 \in I$*,*
- $A_{j_1} \cap A_{j_2} = \emptyset$ *for any* $j_1, j_2 \in J$*,*
- $r : \bigcup_{i\in I} B_i \times \bigcup_{j\in J} A_j \longrightarrow L$*.*

Moreover, consider two non-empty sets of L-contexts notated

- $\{\mathcal{C}_i = \langle B_i, T_i, p_i\rangle : i \in I\}$
- $\{\mathcal{D}_j = \langle O_j, A_j, q_j\rangle : j \in J\}$.

Formal context of second order is a tuple

$$\Big\langle \bigcup_{i\in I} B_i,\ \{\mathcal{C}_i; i \in I\},\ \bigcup_{j\in J} A_j,\ \{\mathcal{D}_j; j \in J\},\ \bigcup_{(i,j)\in I\times J} r_{i,j}\Big\rangle,$$

where $r_{i,j} : B_i \times A_j \longrightarrow L$ defined as $r_{i,j}(o,a) = r(o,a)$ for any $o \in B_i$ and $a \in A_j$.

In what follows, consider the below used notation. Lets have an L-set $f : \prod_{i\in I} X_i \longrightarrow L$ for a non-empty universe set $X = \bigcup_{i\in I} X_i$, where $X_{i_1} \cap X_{i_2} = \emptyset$ for any $i_1, i_2 \in I$. Then $f^i : X_i \longrightarrow L$ is defined as $f^i(x) = f(x)$ for an arbitrary $x \in X_i$ and $i \in I$.

With the help of functor $\varGamma$, we define the mappings between products of fuzzy concept lattices of objects and attributes formal contexts of the following form:

Definition 9. *Lets define the mappings $\langle \Uparrow, \Downarrow\rangle$ as follows*

$$\Uparrow : \prod_{i\in I} \mathrm{FCL}(\mathcal{C}_i) \longrightarrow \prod_{j\in J} \mathrm{FCL}(\mathcal{D}_j) \text{ and } \Downarrow : \prod_{j\in J} \mathrm{FCL}(\mathcal{D}_j) \longrightarrow \prod_{i\in I} \mathrm{FCL}(\mathcal{C}_i)$$

$$\Uparrow (\varPhi)^j = \bigwedge_{i\in I} \lambda_{ijL}(\varPhi^i), \text{ for any } \varPhi \in \prod_{i\in I} \mathrm{FCL}(\mathcal{C}_i)$$

$$\Downarrow (\varPsi)^i = \bigwedge_{j\in J} \lambda_{ijR}(\varPsi^j), \text{ for any } \varPsi \in \prod_{j\in J} \mathrm{FCL}(\mathcal{D}_j)$$

such that $\lambda_{ij} = \langle \lambda_{ijL}, \lambda_{ijR}\rangle = \varGamma(\varphi_{\rho_{ij}})$, where

$$\rho_{ij} = \bigvee\{\beta \in L\text{-Bonds}(\mathcal{C}_i, \mathcal{D}_j) : (\forall (o_i, a_j) \in B_i \times A_j)\beta(o_i)(a_j) \leq r_{ij}(o_i, a_j)\}.$$

Lemma 4. *Let $\{\langle f, g\rangle\} \cup \{\langle f_k, g_k\rangle : k \in K\}$ be a non-empty set of L-concepts of any L-context and K be a non-empty index set. Then*

$$\langle f, g\rangle \preceq \bigwedge_{k\in K} \langle f_k, g_k\rangle = \bigwedge_{k\in K} (\langle f, g\rangle \preceq \langle f_k, g_k\rangle).$$

Proof. Let the L-context be of the form $\langle B, A, r\rangle$. Hence

$$\begin{aligned}\langle f, g\rangle \preceq \bigwedge_{k\in K} \langle f_k, g_k\rangle &= \bigwedge_{o\in B} \Big(f(o) \to \Big(\bigwedge_{k\in K} f_k\Big)(o)\Big) = \bigwedge_{o\in B} \Big(f(o) \to \bigwedge_{k\in K} f_k(o)\Big) \\ &= \bigwedge_{k\in K} \bigwedge_{o\in B} \Big(f(o) \to f_k(o)\Big) = \bigwedge_{k\in K} \Big(\langle f, g\rangle \preceq \langle f_k, g_k\rangle\Big).\end{aligned}$$

□

Lemma 5. *Pair of mappings* $\langle\Uparrow,\Downarrow\rangle$ *forms a Galois connection between complete lattices* $\langle\prod_{i\in I}\mathrm{FCL}(\mathcal{C}_i),\sqsubseteq_I\rangle$ *and* $\langle\prod_{j\in J}\mathrm{FCL}(\mathcal{D}_j),\sqsubseteq_J\rangle$.

Proof. Proof is provided in fuzzy ordering as a generalization of classical one.

$$\begin{aligned}\Psi\sqsubseteq_J\Uparrow(\Phi)&=\bigwedge_{j\in J}(\Psi^j\preceq_j\Uparrow(\Phi)^j)=\bigwedge_{j\in J}\Big(\Psi^j\preceq_j\bigwedge_{i\in I}\lambda_{ijL}(\Phi^i)\Big)\\&=\bigwedge_{i\in I}\bigwedge_{j\in J}(\Psi^j\preceq_j\lambda_{ijL}(\Phi^i))=\bigwedge_{i\in I}\bigwedge_{j\in J}(\Phi^i\preceq_i\lambda_{ijR}(\Psi^j))\\&=\bigwedge_{i\in I}\Big(\Phi^i\preceq_i\bigwedge_{j\in J}\lambda_{ijR}(\Psi^j)\Big)=\bigwedge_{i\in I}(\Phi^i\preceq_i\Downarrow(\Psi)^i)\\&=\Phi\sqsubseteq_I\Downarrow(\Psi).\end{aligned}$$

□

3.1 Simplification

In this subsection will be presented a method that simplifies the previous consideration.

Definition 10. *Let* $\mathcal{C}_i=\langle B_i,A_i,r_i\rangle$ *for* $i\in\{1,2\}$ *be two* L*-fuzzy contexts and let* β *be an arbitrary* L*-bond between* $\mathcal{C}_1$ *and* $\mathcal{C}_2$*. Consider the following pair of mappings* $\uparrow_\beta\colon L^{B_1}\longrightarrow L^{A_2}$ *and* $\downarrow_\beta\colon L^{A_2}\longrightarrow L^{B_1}$ *such that*

$$\uparrow_\beta(f)(a)=\bigwedge_{o\in B_1}(f(o)\to\beta(o)(a)),\qquad\downarrow_\beta(g)(o)=\bigwedge_{a\in A_2}(g(a)\to\beta(o)(a))$$

for any $f\in L^{B_1}$ *and* $g\in L^{A_2}$.

Lemma 6. *Let* $\mathcal{C}_i=\langle B_i,A_i,r_i\rangle$ *for* $i\in\{1,2\}$ *be two* L*-fuzzy contexts and let* β *be an arbitrary* L*-bond between* $\mathcal{C}_1$ *and* $\mathcal{C}_2$*. A pair* $\langle\uparrow_\beta,\downarrow_\beta\rangle$ *forms a Galois connection between complete lattices* $\langle\mathrm{Ext}(\mathcal{C}_1),\leq\rangle$ *and* $\langle\mathrm{Int}(\mathcal{C}_2),\leq\rangle$*, where* $\leq$ *is ordering based on fuzzy sets inclusion.*

Proof. Proof of the fact that $\langle\uparrow_\beta,\downarrow_\beta\rangle$ forms a Galois connection between $\langle L^{B_1},\leq\rangle$ and $\langle L^{A_2},\leq\rangle$ is simple, $\langle\uparrow_\beta,\downarrow_\beta\rangle$ is a pair of derivation operators for L-context $\langle B_1,A_2,\beta^{\mathrm{r}}\rangle$, where binary L-relation β^{r} is defined as $\beta^{\mathrm{r}}(o_1,a_2)=\beta(o_1)(a_2)$.

Now, we will show that $\langle\uparrow_\beta,\downarrow_\beta\rangle$ is a pair of mappings between complete lattices of extents and intents of C_1 and C_2, respectively. First, let f be an extent of C_1.

$$\begin{aligned}\uparrow_\beta(f)(a)&=\bigwedge_{o\in B_1}(f(o)\to\beta(o)(a))\\&=\bigwedge_{o\in B_1}(f(o)\to\uparrow_2(\varphi_{\beta L}(o))(a))\\&=\bigwedge_{o\in B_1}\Big(f(o)\to\bigwedge_{b\in B_2}(\varphi_{\beta L}(o)(b)\to r_2(b,a))\Big)\end{aligned}$$

$$
\begin{aligned}
&= \bigwedge_{b\in B_2}\bigwedge_{o\in B_1}(f(o)\to(\varphi_{\beta L}(o)(b)\to r_2(b,a)))\\
&= \bigwedge_{b\in B_2}\bigwedge_{o\in B_1}((\varphi_{\beta L}(o)(b)\otimes f(o))\to r_2(b,a)))\\
&= \bigwedge_{b\in B_2}\Big(\bigvee_{o\in B_1}(\varphi_{\beta L}(o)(b)\otimes f(o))\to r_2(b,a))\Big)\\
&= \bigwedge_{b\in B_2}(\varphi_{\beta L+}(f)(b)\to r_2(b,a)))\\
&=\uparrow_2(\varphi_{\beta L+}(f))(a).
\end{aligned}
$$

So $\uparrow_\beta(f)$ is an intent of C_2.

Proof of $\downarrow_\beta(g)$ is an extent of C_1 is easy to obtain similarly with equality $\beta(o)(a)=\downarrow_1(\varphi_{\beta R}(o))(a)$. □

We define an L-context such that its L-concept lattice is isomorphic to a complete lattice of all second order formal concepts.

Definition 11. *Let $\mathcal{K}$ be a second order formal context of the form*

$$\mathcal{K}=\Big\langle\bigcup_{i\in I}B_i,\ \{\mathcal{C}_i : i\in I\},\ \bigcup_{j\in J}A_j,\ \{\mathcal{D}_j : j\in J\},\ \bigcup_{(i,j)\in I\times J}r_{ij}\Big\rangle.$$

Lets define an L-context $\widehat{\mathcal{K}}$

$$\widehat{\mathcal{K}}=\Big\langle\bigcup_{i\in I}B_i,\ \bigcup_{j\in J}A_j,\ \bigcup_{(i,j)\in I\times J}\rho_{ij}\Big\rangle,$$

where

$$\rho_{ij}=\bigvee\{\beta\in L\text{-Bonds}(\mathcal{C}_i,\mathcal{D}_j) : (\forall(o_i,a_j)\in B_i\times A_j)\beta(o_i)(a_j)\le r_{ij}(o_i,a_j)\}.$$

Lemma 7. *Concept lattices of $\mathcal{K}$ and $\widehat{\mathcal{K}}$ are isomorphic.*

Proof. Let $\langle\Phi,\Psi\rangle$ be an L-concept of $\widehat{\mathcal{K}}$ and $o\in B_i$.

$$
\begin{aligned}
\Phi^i(o)=(\downarrow_{\widehat{\mathcal{K}}}(\Psi))^i(o) &= \bigwedge_{j\in J}\bigwedge_{a\in A_j}(\Psi^j(a)\to\rho_{ij}(o)(a))\\
&= \bigwedge_{j\in J}\bigwedge_{a\in A_j}(\Psi^j(a)\to\downarrow_i(\varphi_{\rho_{ij}R}(a))(o))\\
&= \bigwedge_{j\in J}\downarrow_i(\varphi_{\rho_{ij}R+}(\Psi^j))(o).
\end{aligned}
$$

$$
\begin{aligned}
\Phi^i=(\downarrow_{\widehat{\mathcal{K}}}(\Psi))^i &= \bigwedge_{j\in J}\downarrow_i(\varphi_{\rho_{ij}R+}(\Psi^j)) = \bigwedge_{j\in J}\operatorname{ext}(\lambda_R^{\varphi_{\rho_{ij}}}(\Psi^j))\\
&= \operatorname{ext}\Big(\bigwedge_{j\in J}\lambda_R^{\varphi_{\rho_{ij}}}(\Psi^j)\Big)=\operatorname{ext}(\Downarrow(\overline{\Psi})^i),
\end{aligned}
$$

where $\overline{\Psi}^j = \langle \downarrow_j (\Psi^j), \Psi^j \rangle$ for any $j \in J$. Then $\overline{\Phi} = \Downarrow (\overline{\Psi})$ and $\langle \overline{\Phi}, \overline{\Psi} \rangle$ is a second order concept of $\mathcal{K}$. $\square$

4 Motivation example – solution

The motivation example introduced in Section 1 can be considered as the second order formal context

$$\langle \{\text{Anna,Bob,Cyril,David,Eva}\}, \mathcal{F}, \{\text{TV,W,Ce,R}\}, \mathcal{H}, r \rangle,$$

whereby r represents the L-relation from $\mathcal{P}$. Firstly, we find a bond ρ that is the closest to r.

$\mathcal{P}$	TV	W	Ce	R
Anna	●		●	●
Bob	○	○	●	●
Cyril	○	●	●	○
David	●	●	○	
Erik	●	●	○	●

$\rho(\mathcal{P})$	TV	W	Ce	R
Anna	●		●	●
Bob	○	○	●	○
Cyril	○	○	●	○
David			○	
Erik			○	

There are just six (instead of twenty-eight L-concepts of $\mathcal{P}$) L-concepts of $\rho(\mathcal{P})$ such that we can easily convert into the form of second order concepts. The following table contains the list of the all second order concepts.

concepts of $\mathcal{F}$	concepts of $\mathcal{H}$

{○/A, ●/B, ●/C, ○/D, ○/E}
{○/A, ●/B, ●/C, ○/D, ○/E}
	{○/TV, ○/W, ●/Ce, ○/R}
	{●/H1, /H2, ○/H3, ○/H4}

{●/A, ●/B, ●/C, ●/D, ●/E}
{ /A, ○/B, ○/C, /D, /E}
	{ /TV, /W, ○/Ce, /R}
	{●/H1, ○/H2, ●/H3, ●/H4}

{●/A, ●/B, ●/C, ○/D, ○/E}
{○/A, ○/B, ○/C, /D, /E}
	{○/TV, /W, ●/Ce, ○/R}
	{●/H1, /H2, ○/H3, ●/H4}

{●/A, ○/B, ○/C, /D, /E}
{●/A, ○/B, ○/C, /D, /E}
	{●/TV, /W, ●/Ce, ●/R}
	{●/H1, /H2, /H3, ●/H4}

{○/A, ○/B, ○/C, /D, /E}
{●/A, ●/B, ●/C, ○/D, ○/E}
	{●/TV, ○/W, ●/Ce, ●/R}
	{●/H1, /H2, /H3, ○/H4}

{ /A, ○/B, ○/C, /D, /E}
{●/A, ●/B, ●/C, ●/D, ●/E}
	{●/TV, ●/W, ●/Ce, ●/R}
	{●/H1, /H2, /H3, /H4}

The first concept can be interpreted as follows. Friends Bob and Cyril with their common requirements should stay in hotel H1. They should stay also in H3 and H4 with a little relaxation of their requirements. The fourth concept is saying that Anna as a very lonely person should stay in H1 or H4. The second concept includes the whole group of co-workers who have very poor common requirements. Thus, all people should stay in an arbitrary hotel together.

5 Connection to heterogeneous formal contexts

The fruitful idea is to view the second order formal context in terms of a heterogeneous formal context proposed in [2]. The corresponding notions of the underlying structures are introduced now.

Definition 12. *Heterogeneous formal context is a tuple* $\langle B, A, \mathcal{P}, R, \mathcal{U}, \mathcal{V}, \odot \rangle$, *where*

- B *and* A *are non-empty sets,*
- $\mathcal{P} = \{\langle P_{b,a}, \leq_{P_{b,a}} \rangle : (b,a) \in B \times A\}$ *is a system of posets,*
- R *is a mapping from* $B \times A$ *such that* $R(b,a) \in P_{b,a}$ *for any* $b \in B$ *and* $a \in A$*,*
- $\mathcal{U} = \{\langle U_b, \leq_{U_b} \rangle : b \in B\}$ *and* $\mathcal{V} = \{\langle V_a, \leq_{V_a} \rangle : a \in A\}$ *are systems of complete latices,*
- $\odot = \{\circ_{b,a} : (b,a) \in B \times A\}$ *is a system of isotone and left-continuous mappings* $\circ_{b,a} : U_b \times V_a \longrightarrow P_{b,a}$*.*

Lets describe our situation in terms of heterogeneous formal contexts. Below is the translation:

- B and A will be the index sets I and J,
- complete lattices U_i or V_j for any $(i,j) \in B \times A = I \times J$ will be the complete lattices $\langle \mathrm{Ext}(\mathcal{C}_i), \leq \rangle$ and $\langle \mathrm{Int}(\mathcal{D}_j), \leq \rangle$,
- $P_{i,j}$ will be a complete lattice of all fuzzy relations from $L^{B_i \times A_j}$,
- any value of relation r will be a binary relation $r(i,j) = r_{i,j} \in L^{B_i \times A_j}$,
- operation $\circ_{i,j} : \mathrm{Ext}(\mathcal{C}_i) \times \mathrm{Int}(\mathcal{D}_j) \longrightarrow L^{B_i \times A_j}$ is defined as

  $$(f \circ_{i,j} g)(b,a) = f(b) \otimes g(a)$$

  for any $f \in \mathrm{Ext}(\mathcal{C}_i)$ and $g \in \mathrm{Int}(\mathcal{D}_j)$ and any $(b,a) \in B_i \times A_j$. The mapping $\circ_{i,j}$ is isotone due to isotonicity of $\otimes$.

Lemma 8. *The mapping* $\circ_{i,j}$ *is left-continuous.*

Proof. Let

$$(f_k \circ g)(b,a) = f_k(b) \otimes g(a) \leq m$$

for all $k \in K$ and for some $(b,a) \in B \times A$ and $m \in L$. It is equivalent to inequality $f_k(b) \leq g(a) \to m$ for all $k \in K$. Hence, $\bigvee_{k \in K} f_k(b) \leq g(a) \to m$ and it is equivalent to

$$\Big(\bigvee_{k \in K} f_k \circ g\Big)(b,a) = \bigvee_{k \in K} f_k(b) \otimes g(a) \leq m.$$

Proof of left-continuity of the second argument is similar. □

Definition 13. *Lets define a pair of derivation operators* $\langle \nwarrow, \searrow \rangle$ *of the following form* $\nwarrow\colon \prod_{i\in I} \mathrm{Ext}(\mathcal{C}_i) \to \prod_{j\in J} \mathrm{Int}(\mathcal{D}_j)$ *and* $\searrow\colon \prod_{j\in J} \mathrm{Int}(\mathcal{D}_j) \to \prod_{i\in I} \mathrm{Ext}(\mathcal{C}_i)$ *defined for heterogeneous formal context mentioned above as follows:*

$$\nwarrow (\Phi)^j = \bigvee \{ g \in \mathrm{Int}(\mathcal{D}_j) : (\forall i \in I) \Phi^i \circ_{i,j} g \leq r_{i,j} \}$$

$$\searrow (\Psi)^i = \bigvee \{ f \in \mathrm{Ext}(\mathcal{C}_i) : (\forall j \in J) f \circ_{i,j} \Psi^j \leq r_{i,j} \}$$

for any $\Phi \in \prod_{i\in I} \mathrm{Ext}(\mathcal{C}_i)$ *and any* $\Psi \in \prod_{j\in J} \mathrm{Int}(\mathcal{D}_j)$*.*

Lemma 9. *Let* $\mathcal{K} = \langle \bigcup_{i\in I} B_i, \{C_i; i \in I\}, \bigcup_{j\in J} A_j, \{D_j; j \in J\}, \bigcup_{(i,j)\in I\times J} r_{i,j} \rangle$ *be a second order formal context. Then*

$$\uparrow_{\widehat{\mathcal{K}}} (\Phi) \leq \nwarrow (\Phi) \text{ and } \downarrow_{\widehat{\mathcal{K}}} (\Psi) \leq \searrow (\Psi)$$

for any $\Phi \in \prod_{i\in I} \mathrm{Ext}(\mathcal{C}_i)$ *and* $\Psi \in \prod_{j\in J} \mathrm{Int}(\mathcal{D}_j)$*.*

Proof. Let $j \in J$ be arbitrary.

$$\begin{aligned} \uparrow_{\widehat{\mathcal{K}}} (\Phi)^j(a) &= \bigwedge_{i\in I} \bigwedge_{o\in B_i} (\Phi^i(o) \to \rho_{ij}(o)(a)) \\ &= \bigvee \{ g \in \mathrm{Int}(\mathcal{D}_j) : (\forall i \in I)(\forall o \in B_i) \Phi^i(o) \otimes g(a) \leq \rho_{ij}(o)(a) \}. \end{aligned}$$

Then

$$\begin{aligned} \uparrow_{\widehat{\mathcal{K}}} (\Phi)^j &= \bigvee \{ g \in \mathrm{Int}(\mathcal{D}_j) : (\forall i \in I)(\forall o \in B_i) \Phi^i \circ_{ij} g \leq \rho_{ij} \} \\ &\quad \text{because of } \rho_{ij} \leq r_{ij} \\ &\quad \leq \bigvee \{ g \in \mathrm{Int}(\mathcal{D}_j) : (\forall i \in I)(\forall o \in B_i) \Phi^i \circ_{ij} g \leq r_{ij} \} \\ &\quad = \nwarrow (\Phi)^j. \end{aligned}$$

Hence $\uparrow_{\widehat{\mathcal{K}}} (\Phi) \leq \nwarrow (\Phi)$. Similarly for $\downarrow_{\widehat{\mathcal{K}}}$ and $\searrow$. □

6 Connections to standard homogenic fuzzy operators

In this part, we focus on an appropriate generalization of the standard homogenic fuzzy formal concept derivation operators in two different ways.

6.1 Singleton connection

Lemma 10. *Lets have an L-fuzzy formal context* $\langle \{x\}, L, \lambda \rangle$*, where for an arbitrary* $k \in L$ *is* $\perp_x = \lambda(x, k) = k$*. Any value* $k \in L$ *is an extent of* $\perp_x$*.*

Proof. Let k be an arbitrary value from L.

$$\downarrow\uparrow (k)(x) = \bigwedge_{m\in L} (\uparrow (k)(m) \to m) = \bigwedge_{m\in L} (\bigwedge_{x\in\{x\}} (k \to m) \to m)$$

$$
\begin{aligned}
&= \bigwedge_{m\in L:m\geq k}(1\to m)\wedge \bigwedge_{m\in L:m<k}((k\to m)\to m)\\
&= \bigwedge_{m\in L:m\geq k} m\wedge \bigwedge_{m\in L:m<k}((k\to m)\to m) = \star\\
&\bigwedge_{m\in L:m\geq k} m = k \text{ (L is a complete lattice)}\\
&(k\to m)\to m\geq k \text{ (closure property)}\\
&\bigwedge_{m\in L:m<k}((k\to m)\to m)\geq k\\
&\star = k
\end{aligned}
$$

So $\downarrow\uparrow(k) = k$ for any $k\in L$. $\square$

Lemma 11. *Lets have an L-fuzzy formal context* $\mathcal{C} = \langle B, A, r\rangle$ *with derivation operators* $\langle\uparrow,\downarrow\rangle$. *Lets have a second order formal context*

$$\mathcal{K} = \Big\langle \bigcup_{b\in B}\{b\}, \{\bot_b : b\in B\}, \bigcup_{a\in A}\{a\}, \{\bot^*_a : a\in A\}, \bigcup_{(b,a)\in B\times A} r(b,a)\Big\rangle.$$

Then concept lattices of $\mathcal{C}$ *and* $\mathcal{K}$ *are isomorphic.*

Proof. Let $\Phi\in\prod_{b\in B}\mathrm{Ext}(\bot_b)$. By previous lemma is easy to see that $\Phi\in L^B$. Moreover $r(b,a)$ for any $(b,a)\in B\times A$ as an arbitrary value from L is an extent of $\bot_b$ and intent of $\bot^*_a = \langle L, \{a\}, {}^{\mathrm{t}}\lambda\rangle$. Hence any $r(b,a)\in L$-Bonds$(\bot_b,\bot^*_a)$.

$$\uparrow_{\widehat{\mathcal{K}}}(\Phi)(a) = \bigwedge_{b\in B}\uparrow_{r(b,a)}(\Phi(o)) = \bigwedge_{b\in B}(\Phi(o)\to r(o,a)) = \uparrow(\Phi)(a).$$

Similarly for $\downarrow_{\widehat{\mathcal{K}}} = \downarrow$. $\square$

6.2 $\neq$ connection

Moreover, an another connection is presented in this subsection. The connection is based on the fact that concept lattice of $\langle X, X, \neq\rangle$ is isomorphic to L^X in the case that L is closed under double negation law.

Lemma 12. *Lets have an L-fuzzy formal context* $\mathcal{X} = \langle X, X, \neq\rangle$, *where* L *is closed under double negation law. Then any L-set from* L^X *is closed in* $\mathcal{X}$.

Proof. Lets have an arbitrary L-set $f\in L^X$.

$$
\begin{aligned}
\uparrow(f)(x) &= \bigwedge_{y\in X}(f(y)\to(y\neq x))\\
&= \bigwedge_{y\in X:y\neq x}(f(y)\to 1)\wedge(f(x)\to(x\neq x))\\
&= 1\wedge\neg f(x) = \neg f(x)
\end{aligned}
$$

Then $\downarrow\uparrow(f)(x) = \neg\neg f(x) = f(x)$. $\square$

Lemma 13. *Lets have an L-fuzzy formal context $\mathcal{C} = \langle B, A, r\rangle$, where L is closed under double negation law. Concept lattice of $\mathcal{C}$ is isomorphic to a concept lattice of second order formal context $\mathcal{K} = \left\langle B, \mathcal{B}, A, \mathcal{A}, r\right\rangle$, where $\mathcal{B} = \langle B, B, \neq\rangle$ and $\mathcal{A} = \langle A, A, \neq\rangle$.*

Proof. Index sets I and J are singletons in this case. So due to previous lemma $\Phi \in \mathrm{Ext}(\mathcal{B}) = L^B$. $\Uparrow(\Phi) = \uparrow_\rho (f)$ where $\rho = \bigvee\{\beta \in L\text{-Bonds}(\mathcal{B}, \mathcal{A}) : \beta \leq r\}$. Because of the previous lemma we know that any row and column or r is closed in $\mathcal{B}$ and $\mathcal{A}$, respectively. Hence $r \in L\text{-Bonds}(\mathcal{B}, \mathcal{A})$ and $r = \rho$.

Finally $\uparrow_{\widehat{\mathcal{K}}} = \uparrow$. Similarly for $\downarrow_{\widehat{\mathcal{K}}} = \downarrow$. □

References

1. L. Antoni, S. Krajči, O. Krídlo, B. Macek, and L. Pisková. Relationship between two FCA approaches on heterogeneous formal contexts. *Proceedings of CLA*, 93–102, 2012.
2. L. Antoni, S. Krajči, O. Krídlo, B. Macek, and L. Pisková. On heterogeneous formal contexts. *Fuzzy Sets and Systems*, In Press.
3. R. Bělohlávek. Fuzzy concepts and conceptual structures: induced similarities. *Proceedings of Joint Conference on Information Sciences*, pp. 179–182, 1998.
4. R. Bělohlávek. Lattices of fixed points of fuzzy Galois connections. *Mathematical Logic Quartely*, 47(1):111–116, 2001.
5. R. Bělohlávek. Concept lattices and order in fuzzy logic. *Annals of Pure and Applied Logic*, 128(1–3):277-298, 2004.
6. R. Bělohlávek. Lattice-type fuzzy order is uniquely given by its 1-cut: proof and consequences. *Fuzzy Sets and Systems*, 143:447–458, 2004.
7. B. Ganter and R. Wille. *Formal concept analysis.* Springer–Verlag, 1999.
8. O. Krídlo and M. Ojeda-Aciego. On the L-fuzzy generalization of Chu correspondences. *International Journal of Computer Mathematics*, 88(9):1808-1818, 2011.
9. 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.
10. O. Krídlo and M. Ojeda-Aciego. Linking L-Chu correspondences and completely lattice L-ordered sets. *Proceedings of CLA*, 233–244, 2012.
11. J. Medina and M. Ojeda-Aciego. Multi-adjoint t-concept lattices. *Information Sciences*, 180:712–725, 2010.
12. J. Medina and M. Ojeda-Aciego. On multi-adjoint concept lattices based on heterogeneous conjunctors. *Fuzzy Sets and Systems*, 208: 95–110, 2012.
13. H. Mori. Chu Correspondences. *Hokkaido Matematical Journal*, 37:147–214, 2008.
14. H. Mori. Functorial properties of Formal Concept Analysis. Proc ICCS, *Lecture Notes in Computer Science*, 4604:505–508, 2007.
15. J. Pócs. Note on generating fuzzy concept lattices via Galois connections. *Information Sciences* 185 (1):128–136, 2012.
16. J. Pócs. On possible generalization of fuzzy concept lattices. *Information Sciences*, 210: 89–98, 2012.

AOC-posets: a scalable alternative to Concept Lattices for Relational Concept Analysis

Xavier Dolques[1], Florence Le Ber[1], and Marianne Huchard[2]

(1) ICube, Université de Strasbourg/ENGEES, CNRS, Strasbourg,
{xavier.dolques , florence.leber}@engees.unistra.fr
(2) LIRMM, Université de Montpellier 2 et CNRS
huchard@lirmm.fr

Abstract. Relational Concept Analysis (RCA) is a useful tool for classification and rule discovery on sets of objects with relations. Based on FCA, it produces more results than the latter but also an increase in complexity. Besides, in numerous applications of FCA, AOC-posets are used rather than lattices in order to reduce combinatorial problems. An AOC-poset is a subset of the concept lattice considering only concepts introducing an object or an attribute. AOC-posets are much smaller and easier to compute than concept lattices and still contain the information needed to rebuild the initial data. This paper introduces a modification of the RCA process based on AOC-posets rather than concept lattices. This work is motivated by a big set of relational data on river streams to be analysed. We show that using AOC-poset on these data provides a reasonable concept number.

1 Introduction

Relational Concept Analysis (RCA) [1] is based on iterative use of the classical Formal Concept Analysis algorithm to handle relational data: formal objects are described with formal attributes as in FCA, and with their relationships with other formal objects. Because RCA groups formal objects using relationships to formal objects at any distance, it often comes with a combinatorial explosion, and patterns of interest are difficult to extract from the huge set of built concepts. Various strategies can be used to cope with this complexity, including separating the initial formal object sets into smallest ones after a first analysis, or introducing queries [2]. Here we propose to adapt the RCA process in order to use only posets of concepts introducing objects or attributes (AOC-poset) rather than to build full concept lattices at each step of the process. Indeed AOC-posets are smaller and easier to compute than concept lattices [3] and their sets of concepts have interesting properties for extracting implication rules.

The context of this research is the FRESQUEAU project[1] which aims at developing new methods for studying, comparing and exploiting all the parameters available concerning streams and water areas. The data we deal with have been

---

[1] `http://engees-fresqueau.unistra.fr/`

© paper author(s), 2013. Published in Manuel Ojeda-Aciego, Jan Outrata (Eds.): CLA 2013, pp. 129–140, ISBN 978–2–7466–6566–8, Laboratory L3i, University of La Rochelle, 2013. Copying permitted only for private and academic purposes.

collected during 3 years over 40 sites in the Alsace plain. Two other data sets are available on the Rhin-Meuse district (North-east of France, 3400 sites, 10 years) and on the Rhône-Méditerranée district (South-east of France, 18000 sites, 40 years). In a previous work, FCA has been used for classifying characteristics of aquatic species [4]. RCA is now used for discovering relationships between those species characteristics and the characteristics of their habitat.

To perform this analysis, we propose to adapt RCA relying on AOC-poset. We show that this approach provides a reasonable concept number in our case.

The paper is organized as follows. Section 2 gives some useful definitions for our presentation. Section 3 details the RCA process based on AOC-poset. Section 5 presents the data set and results that can be obtained with AOC-posets. Related work is presented in section 6 and section 7 concludes the paper opening some perspectives of this work.

2 FCA Basics

Formal Concept Analysis (FCA) [5] aims at extracting an ordered set of concepts from a dataset, called a Formal Context, composed of objects described by attributes. We will denote by $\mathcal{K} = (\mathcal{G}, \mathcal{M}, \mathcal{I})$ a formal context, where $\mathcal{I} \subseteq \mathcal{G} \times \mathcal{M}$. Table 1 is a Formal Context $\mathcal{K}_{Animals} = (\mathcal{G}_{Animals}, \mathcal{M}_{Animals}, \mathcal{I}_{Animals})$ which describes animals by characteristics they may own. The considered animals here are macro-invertebrates that can be found in rivers. We took the examples of the following kinds of animals : *Ancylus (Anc.), Anisus (Ani.), Anodonta (Ano.), Anthomyiidae (Ant.), Aphelocheirus (Aph.), Asellus (Ase.) and Athericidae (Ath.).*

Table 1. Formal Context of animals described by their life traits $\mathcal{K}_{Animals}$

	size (1-2cm) (**att0**)	resistance to dessication (**att1**)	Micro-habitat in sand (**att2**)	reproduction clutches terrestrial (**att3**)	transversal distribution in lakes (**att4**)	transversal distribution on banks (**att5**)
Anc.	×					×
Ani.	×	×				×
Ano.			×			
Ant.	×		×	×		
Aph.	×		×		×	
Ase.	×				×	×
Ath.	×		×	×	×	

Given a $\mathcal{K} = (\mathcal{G}, \mathcal{M}, \mathcal{I})$ formal context, a formal concept is a $C = (Extent(C), Intent(C))$ pair where:

$Extent(C) = \{g \in \mathcal{G} | \forall m \in Intent(C), (g, m) \in \mathcal{I}\}$ is the extent of the concept, $Intent(C) = \{m \in \mathcal{M} | \forall g \in Extent(C), (g, m) \in \mathcal{I}\}$ is the intent of the concept. Given two formal concepts $C_1 = (E_1, I_1)$ and $C_2 = (E_2, I_2)$ of $\mathcal{K}$, the concept specialization order $\leq_s$ is defined by $C_1 = (E_1, I_1) \leq_s C_2 = (E_2, I_2)$ if and only if $E_1 \subseteq E_2$ (and equivalently $I_2 \subseteq I_1$).

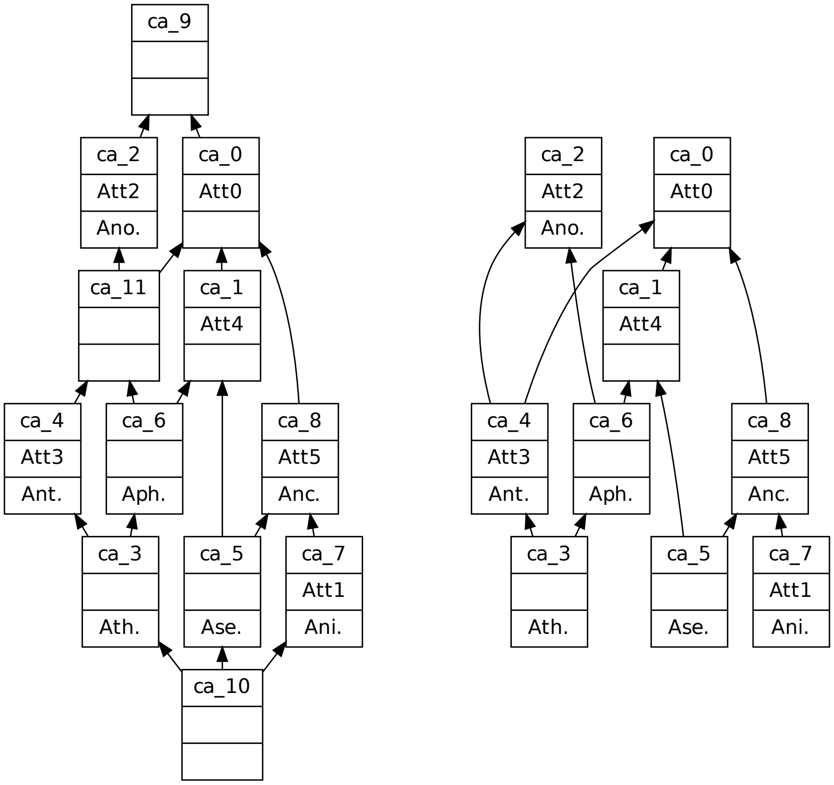


Fig. 1. Concept Lattice of animals (left), AOC poset $AOC_{animals}$ (right)

For simplicity's sake, most lattice representations show attributes (*resp.* objects) solely where they are introduced. They are said to show the simplified intents and simplified extents which will be denoted respectively $Intent_S(C)$ and $Extent_S(C)$ for a given concept C.

Let $\mathcal{C_K}$ be the set of all concepts of a $\mathcal{K}$ formal context. This set of concepts provided with the $(\mathcal{C_K}, \leq_s)$ specialization order has a lattice structure, and is called the concept lattice associated with $\mathcal{K}$.

Left-hand side of Fig. 1 shows the concept lattice associated with the formal context of Table 1. In several FCA applications (*e.g.* in [3,6]; only the *object concepts* (which introduce at least one object) and the *attribute concepts* (which introduce at least one attribute) are used. In Fig. 1 (left-hand side), ca_5 and ca_8 are examples of object concepts; ca_1 and ca_2 are examples of attribute concepts; ca_9, ca_10 and ca_11 do not introduce any object or any attribute.

The so-called AOC-poset (for Attribute-Object-Concept poset) is the suborder of $(\mathcal{C}_{\mathcal{K}}, \leq_s)$ restricted to object-concepts and attribute-concepts. It is also called Galois Sub-Hierarchy, a term we consider less explicit. Right-hand side of Fig. 1 shows the AOC-poset for the context of Table 1. One may have a large difference of complexity between the two structures, because the concept lattice may have $2^{min(|\mathcal{G}|,|\mathcal{M}|)}$ concepts, while the number of concepts in the AOC-poset is bounded by $|\mathcal{G}| + |\mathcal{M}|$.

3 A variant of Relational Concept Analysis with AOC-poset

Relational Concept Analysis aims at extending Formal Concept Analysis to take into account a dataset where objects of several categories are described by attributes and by relations to objects [1,7]. The dataset is called a Relational Context Family.

Definition 1 (Relational Context Family (RCF)). *A Relational Context Family (denoted RCF) is a* $(\mathbf{K}, \mathbf{R})$ *pair where:*

- $\mathbf{K} = \{\mathcal{K}_i\}_{i=1,\dots,n}$ *is a set of* $\mathcal{K}_i = (G_i, M_i, I_i)$ *contexts*
- $\mathbf{R} = \{r_j\}_{j=1,\dots,m}$ *is a set of* r_j *relations where* $r_j \subseteq G_{i_1} \times G_{i_2}$ *for some* $i_1, i_2 \in \{1, \dots, n\}$.

An example of a RCF is composed of an Animal context, denoted $\mathcal{K}_{Animals}$ (Table 1), a Site context, denoted $\mathcal{K}_{Sites}$ (Table at the left-hand side of Fig. 2), and a *contains* relation, denoted $r_{contains}$ (Table 2). Sites are determined by locations on a river where samples are done; two kinds of samples are here considered: physico-chemical samples that measure physical and chemical properties of the water (*e.g.* temperature or presence of a chemical compound) and biological samples that measure the level of life.

Table 2. Relation $r_{contains}$

contains ↗	***Anc.***	***Ani.***	***Ano.***	***Ant.***	***Aph.***	***Ase.***	***Ath.***
Site0				×	×		×
Site1	×			×	×		×
Site2		×		×	×		×
Site3	×	×				×	

When objects of a category (*e.g.* Sites) are connected to objects of another category (*e.g.* Animals) *via* a relation (*e.g.* contains), concepts formed on top of objects of the latter category can be used to form concepts on top of objects

	NH4	**SO4**
Site0	×	×
Site1		×
Site2	×	×
Site3		

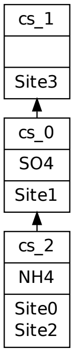


Fig. 2. Formal Context of Sites $\mathcal{K}_{Sites}$ (left), AOC-poset of Sites (right)

of the former category. Let us for example consider the AOC-poset of the right-hand side of Fig. 1 and the `ca_8` concept which groups animals with a transversal distribution on banks (Att5), namely *Ancylus (Anc.)*, *Anisus (Ani.)* and *Asellus (Ase.)*. Now Site1, Site2 and Site3 can be grouped because they contain at least one animal with Att5, that is at least one animal in $Extent($`ca_8`$)$, *e.g. Ancylus (Anc.)* for Site 1 (Table 2). In RCA, this relationship between objects of a category and concepts formed on objects of another category is implemented thanks to *scaling operators*. This results in the creation of special attributes called relational attributes. The most used *scaling operators* are:

- the *existential* scaling operator which encodes the fact that an object o is in relation by $\exists r$ with a concept C if $r(o)$ has a non-empty intersection with $Extent(C)$
- and the *strict universal* scaling operator which encodes the fact that an object o is in relation by $\forall r$ with a concept C if $r(o)$ is non-empty and included in the extent of C.

For a given r relation in a given analysis, its associated scaling operator is denoted by $\rho(r)$. Now, since we rely on AOC-posets, the existential scaling operator is defined slightly differently from [7]. Nevertheless the reference [7] can be read to have more details on the RCA process which is simplified below for space reasons.

Definition 2 (Existential scaling operator). *Let* $\mathcal{K} = (G, M, I)$ *be a context, and* r *a relation, where* G *is the domain of* r, G_{i_r} *is the range of* r, *and* $\mathcal{K}_{i_r} = (G_{i_r}, M_{i_r}, I_{i_r})$ *is another context. Let also* $\mathcal{C}_{i_r}$ *be a set of concepts*[2] *built on* $\mathcal{K}_{i_r}$. *The* $\mathbb{S}_\exists$ *existential scaling operator is applied to the context* $\mathcal{K}$, *denoted by* $\mathbb{S}_\exists(\mathcal{K}, r, \mathcal{C}_{i_r}) = \mathcal{K}^+ = (G^+, M^+, I^+)$, *with:*

- $G^+ = G$
- $M^+ = \{\exists r(C) \mid C \in \mathcal{C}_{i_r}\}$, *where each* $\exists r(C)$ *is a relational attribute*

---

[2] rather than a concept lattice built on $\mathcal{K}_{i_r}$ in [7].

- $I^+ = \{(o, \exists r(C)) \mid o \in G, C \in \mathcal{C}_{i_r}, r(o) \cap Extent(C) \neq \emptyset\}$

Table 3 shows $\mathbb{S}_\exists(K_{Sites}, contains, \mathcal{C}_{Animals})$ where $\mathcal{C}_{Animals}$ is the set of concepts of the AOC-poset of right-hand side of Fig. 1. In this case I^+ contains ($Site3$, $\exists\ contains(ca\_5)$) and ($Site3$, $\exists\ contains(ca\_7)$) because Site3 contains one animal from the extent of these two concepts.

Table 3. Existential Scaling of Formal Context of sites $\mathbb{S}_\exists(K_{Sites}, contains, \mathcal{C}_{Animals})$. **ca_i** is a short expression for $\exists\ contains(ca\_i)$.

$\exists$*contains*	**ca_0**	**ca_1**	**ca_2**	**ca_3**	**ca_4**	**ca_5**	**ca_6**	**ca_7**	**ca_8**
Site0	×	×	×	×	×		×		
Site1	×	×	×	×	×		×		×
Site2	×	×	×	×	×		×	×	×
Site3	×	×				×		×	×

Then, for each $\mathcal{K}$ context of $\mathbf{K}$, the *apposition* of $\mathcal{K}$ (denoted by symbol '|') with the respective results of the scaling upon each r_j of $\mathbf{R}$ with G as domain ($1 \leq j \leq k$) and the chosen $\rho(r_j)$, is used to build a new set of concepts (notations are taken from Def. 2). This apposition is the relational extension of the $\mathcal{K}$ context considering ρ and a set of concepts $\mathbf{C}$ which is a union of concept sets including $\mathcal{C}_{i_{r_j}}$, $1 \leq j \leq k$:

$$\mathbb{E}_{\rho,\mathbf{C}}(\mathcal{K}) = \mathcal{K} \mid \mathbb{S}_{\rho(r_1)}(\mathcal{K}, r_1, \mathcal{C}_{i_{r_1}}) \mid \ldots \mid \mathbb{S}_{\rho(r_k)}(\mathcal{K}, r_k, \mathcal{C}_{i_{r_k}})$$

By extension, $\mathbb{E}^*_{\rho,\mathbf{C}}(\mathbf{K})$ denotes the relational extension of $\mathbf{K}$, which is composed of all the relational extensions of all $\mathcal{K}_i$ in $\mathbf{K}$ (and $\mathbf{C}$ is a union of concept sets associated with all ranges of all relations).

$$\mathbb{E}^*_{\rho,\mathbf{C}}(\mathbf{K}) = \{\mathbb{E}_{\rho,\mathbf{C}}(\mathcal{K}_1), \ldots, \mathbb{E}_{\rho,\mathbf{C}}(\mathcal{K}_n)\}.$$

For example a relational extension of our $\mathbf{K}$ example is composed of Table 1 (no outgoing relation), and the left table of Fig. 2 apposed to Table 3. The AOC-poset built from this extended context is shown on figure 3.

Now a whole construction process consists in building a (possibly infinite) sequence of contexts and AOC-posets associated with $(\mathbf{K}, \mathbf{R})$ and ρ. The first set of contexts (step 0) is $\mathbf{K}^0 = \mathbf{K}$. The contexts of step p are used to build the associated AOC-posets. The $\mathbf{C}_p$ set composed of the sets of the concepts of AOC-posets at step p is used to calculate the relational extension. The set of contexts at step $p+1$ is defined using the relational extension: $\mathbf{K}^{p+1} = \mathbb{E}^*_{\rho,\mathbf{C}_p}(\mathbf{K})$.

Considering independently the contexts: for some $i \in \{1, \ldots, |\mathbf{K}|\}$ the i-th context at the step $p+1$ is the relational extension of the context of the same rank with the concepts from step p: $\mathcal{K}_i^0 = \mathcal{K}_i$, $\mathcal{K}_i^{p+1} = \mathbb{E}_{\rho,\mathbf{C}_p}(\mathcal{K}_i)$. The difference here with classical RCA is that we rely on $\mathcal{K}_i$ and concepts of the AOC-posets of step p to build $\mathcal{K}_i^{p+1}$, rather than on $\mathcal{K}_i^p$ and lattices of step p.

In our example, $\mathbf{K}^0 = (K_{Animals}, K_{Sites})$ and $\mathbf{R} = \{contains\}$ (see Table 2). If $\rho(contains) = \exists$, then

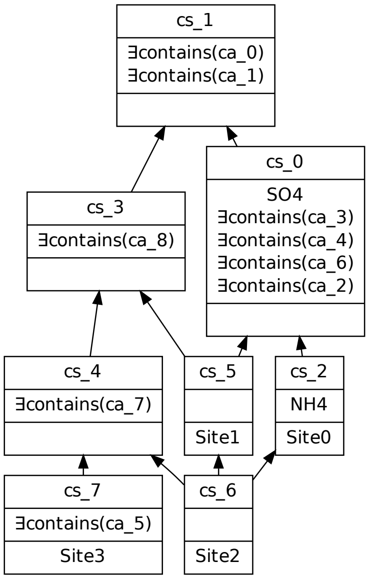


Fig. 3. AOC-posets for the K_{Sites} formal context and the $\mathcal{C}_{Animals}$ set of concepts, existential scaling

- At step 0, we obtain the AOC-posets of Fig. 1 (right) and Fig. 2 (right); $\mathbf{K}^1 = \{K_{Animals}, K_{Sites} \mid \mathbb{S}_{\exists}(K_{Sites}, contains, \mathcal{C}_{Animals})\}$.
- At step 1, we obtain the AOC-posets of Fig. 1 (right) and Fig. 3 (left).

In this simple example, the following steps do not produce any new concept (a fix-point is reached). This process will be denoted by RCA-AOC in the following.

4 Extracting Implication Rules

Implication rules are implications that are verified by the whole data set considered. Some implication rules can be extracted from the AOC-poset concepts by considering their simplified intent.

An attribute a from the simplified intent of a concept C is an attribute that is not contained in the intent of any concept more general than C *i.e.* the set of all the objects sharing a is the extent of C. The objects from the extent also all share the attributes of the full intent of C, but they may not be the only ones. By consequence, for every object o in the data set the presence of a for a given object o implies the presence of all the attributes from the intent of C.

For instance in Fig. 1 we can consider the concept `ca_7`. $Intent(\texttt{ca\_7}) = \{Att1, Att0, Att5\}$ and $Intent_S(\texttt{ca\_7}) = \{Att1\}$ which means that in all the dataset the rule $Att1 \rightarrow Att0 \wedge Att5$ is verified.

All the concepts with no empty simplified intent can be found in the AOC-poset meaning that all the rules having one element at the left hand side can be found with an AOC-poset.

Extracting rules from an existing AOC-poset is straightforward as it consists in reading the simplified intent and the full intent of each concept. The concept order permits to extract rules ordered by support, the rules extracted from the most general concepts having a larger support than more specific rules. The use of RCA permits to use relational attributes in implication rules.

5 A case study

We rely on a large database collecting data on Alsatian streams and water areas (North-east of France) [8], but more data are available through the current FRESQUEAU project, concerning larger areas and periods. The data are either issued from samples (*e.g.* physical, chemical and biological data collected on stream sites), synthetic data (*e.g.* biological indices, land cover) or general information issued from the literature (*e.g.* information about the aquatic species living in the streams). More precisely in this paper we work with three tables. The first one gives values of 27 physico-chemical parameters (*e.g.* temperature, pH, SO4, NH4, organic matters) collected on 49 stream sites. The second table gives the level of population for 197 macro-invertebrates (*e.g. Ancylus*, *Anisus*, *Anodonta*) collected on the same 49 sites. The third one describes the macro-invertebrates with 22 different life traits, *i.e.* their characteristics and functioning, *e.g* life cycle, reproduction mode, etc. each life trait being represented by several modalities (*e.g.* for the life trait *life cycle* there are two possible modalities : *less than a year* or *more than a year*). The sum of the modalities for all life traits is 116. We look for rules combining life traits and physico-chemical parameters, *e.g. "the M modality of the T life trait is associated with a high value of the C physico-chemical parameter"*.

We modeled our data within 4 formal contexts: *stream sites*, *physico-chemical parameters*, *life traits* and *macro-invertebrates* and we considered the three relations between them that are described by our tables: level of physico-chemical parameter, population of macro-invertebrates and life trait of macro-invertebrates. The relations being originally numerical ones we applied a preprocessing to separate values into a few classes. So the *level of physico-chemical parameter* relation has been split into 5 binary relations describing 5 different levels, the *population of macro-invertebrates* relation has also been split into 5 different binary relations and the *life trait of macro-invertebrates* relation has been split into 6 binary relations. This binarisation process has been accomplished under the guidance of a domain expert.

Applying RCA on this data led to a combinatorial explosion that forced us to try other methods of classification as our implementation could not sustain the number of concepts created reaching the limits of time and size. On this same data RCA-AOC gives a result in a few seconds, and can then be used on much bigger data sets. The implementation of RCA-AOC relies on the same algorithm as RCA except for the use of the Hermes algorithm [9] for building AOC-posets instead of an FCA algorithm.

From the obtained AOC-posets we extracted two sets of rules: the rules with a physico-chemical character on the left hand side and the rules with animal traits on the left hand side.

We obtained 49179 rules for the first set and 56 for the second one. The difference is coming from the granularity of data which is different for the physico-chemical characters and the biological characters. It comes directly from the choices of the binarization guided by domain experts in order to support the extraction of rules in one direction. Here the goal is to infer biological characters from physico-chemical characters, the former being more expensive to measure than the latter.

The rule presented below is one rule extracted from the experimental data with a support of 67% of the considered sites.

$$\begin{aligned} \mathit{Site} \sqcap \exists \mathit{lvl1}.\mathit{Lig} \sqsubseteq \exists \mathit{lvl2}.(&\mathit{Animal} \sqcap \exists \mathit{aff0}.(\mathit{Trait} \sqcap \mathit{FR4}) \\ &\sqcap \exists \mathit{aff3}.(\mathit{Trait} \sqcap \mathit{SA1})) \end{aligned}$$

For any *site* of the data set the presence of the component *Lig* at a concentration of level 1 implies the presence of *animals* at a concentration of level 2 that have the *trait* identified by the code *FR4* (modality *diapause or dormancy* of trait *resistance kind*) with an affinity of 0 and the *trait* identified by the code SA1 (modality *fresh water* of the trait *salinity*) with an affinity of 3.

The levels that are referred here are defined specifically to each physico-chemical character, each animal species, and each biological character. For the physico-chemical characters they are usually attached to a level of concentration. The same goes for the animals presence. For the biological characters it refers to an affinity of the species to the modality of a particular biological trait (*e.g.* for its habitat, an animal may have a stronger affinity with *sand* than *gravel*, but may be found in both of them).

Although the modeling can be considered as naive and several approaches could be used to limit the number of concepts, we saw here that RCA has a scale issue as we are only working on a small part of the data. Using AOC-poset allows to handle big data without exploding the number of concepts. Besides, reducing the number of concepts means that the extracted information is also reduced. For the stream site concepts, from which we extract the rules, the AOC-poset keeps the interesting data as we still have the concepts where each attribute is introduced and their order. For the macro-invertebrates however, several concepts are lost that represent combinations of shared life traits. We still have to measure the impact of the missing concepts on the results but it would seem appropriate to combine the RCA-AOC approach with a more traditional lattice building approach that would compute the relevant concepts for the macro-invertebrates while keeping the number of stream site concepts low with AOC-poset.

6 Related Work

Other approaches to integrate relations in FCA have been proposed, including *power context family* [10], and *Logical Concept Analysis* [11]. The originality of RCA is to compute in iterative manner (with a possible stop at each step) several concept lattices from data represented in relational format. The concept lattices are connected by links that abstract the relations between objects. Several operators, borrowed to Description Logics, build the links between concepts.

The original RCA framework, using whole concept lattices, has been used to the analysis and modernization of UML elements [12,13], namely in class diagrams and in use case diagrams. In [14], concept lattices exploit relations between methods and between methods and attributes to detect and fix design defects. Model transformations are learned from transformation examples thanks to several kinds of relations between model elements (*e.g.* between elements inside a model, transformation links between source elements and target elements) [15]. In [16], relations between abstract tasks in an abstract orchestration are used to classify relevant Web services to instantiate the tasks. Other applications can be found in ontology engineering [17,18,19]. In these applications, the datasets are medium-size guarantying the feasibility of the approach. In the FRESQUEAU project, we have larger sets of data where AOC-posets are more suitable than concept lattices. Furthermore, for some issues we want to deal with in the project, like extracting part of the implication rules, AOC-posets are relevant because they contain all the concepts that introduce attributes.

To our best knowledge, the AOC-posets have been introduced by Godin et al [3] in the domain of software engineering (object-oriented programming). The AOC-poset has also been used in applications of FCA to non-monotonic reasoning and domain theory [20] and to produce classifications from linguistic data [21,22]. Specific parts of the AOC-poset (mainly the attribute-concepts part) are used in several works. They include approaches for rebuilding class hierarchy [23] and a recent study for extracting feature tree (in the domain of software product lines) from a set of products [24].

Many approaches exist to extract logical rules from data either in a supervised context for building decision tree [25] or classification rules [26] or in an unsupervised context for association rules learning. Our approach is unsupervised, but AOC-posets only allows the extraction of association rules of confidence 1.

However, the scaling operation permits to consider different kinds of granularity for the relations between the concepts. The relational aspect of the approach permits the extraction of rules from complex relational data which would require to be transformed by propositionalization approaches [27] for many learning approaches. It also could be compared to Inductive Logic Programming [28] in some ways as it will result in first order logic formulas. ILP being a supervised approach, its goal differs as it will try to find the right premise for a given conclusion. The expressivity of the results of our approach are far more restricted than with ILP, even with all the scaling operators that can be imagined, leading to better performances but with a restricted output language.

7 Conclusion

This paper has introduced a RCA process based on AOC-poset in order to deal with computational complexity over big datasets. AOC-posets reduce the number of concepts, with no information lost as the context can still be retrieved from an AOC-poset. In the future we will work on specifying the convergence conditions of AOC-poset based RCA. Indeed, more complex datasets may include cycles between objects. Convergence is ensured with the RCA specification from [7] where the set of concepts used at each step is the set of concepts of the whole lattice. With AOC-poset the convergence is not guaranteed when there are cycles between objects. We also will develop a tool for vizualising the results and assisting their exploration. Finally the approach will be tested on the various datasets of the FRESQUEAU project, and compared with other approaches for relational data mining such as statistical approaches [29] or propositionalization [27].

Acknowledgement. This work was funded by ANR11_MONU14. We warmly acknowledge Corinne Grac (LHYGES) for advices about the dataset, Clémentine Nebut (LIRMM) for the example and Alain Gutierrez (LIRMM) for the implementation of the AOC-poset algorithm (`http://www.lirmm.fr/~gutierre/gsh/`).

References

1. Huchard, M., Rouane-Hacène, M., Roume, C., Valtchev, P.: Relational concept discovery in structured datasets. Ann. Math. Artif. Intell. **49**(1-4) (2007) 39–76
2. Azmeh, Z., Huchard, M., Napoli, A., Rouane-Hacène, M., Valtchev, P.: Querying relational concept lattices. In: CLA'11. (2011) 377–392
3. Godin, R., Mili, H.: Building and Maintaining Analysis-Level Class Hierarchies using Galois Lattices. In: OOPSLA '93. Volume 28. (1993) 394–410
4. Bertaux, A., Le Ber, F., Braud, A., Trémolières, M.: Identifying ecological traits: a concrete FCA-based approach. In: ICFCA 2009. LNAI 5548, Springer-Verlag (2009) 224–236
5. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical Foundations. Springer-Verlag (1999)
6. Loesch, F., Ploedereder, E.: Restructuring variability in software product lines using concept analysis of product configurations. In: CSMR 2007. (2007) 159–170
7. Hacene, M.R., Huchard, M., Napoli, A., Valtchev, P.: Relational concept analysis: mining concept lattices from multi-relational data. Ann. Math. Artif. Intell. **67**(1) (2013) 81–108
8. Grac, C., Le Ber, F., Braud, A., Trémolières, M., Bertaux, A., Herrmann, A., Manné, S., Lafont, M.: Programme de recherche-développement *Indices* – rapport scienfique final. Contrat pluriannuel 1463 de l'Agence de l'Eau Rhin-Meuse, LHYGES – LSIIT – ONEMA – CEMAGREF (2011)
9. Berry, A., Huchard, M., Napoli, A., Sigayret, A.: Hermes: an efficient algorithm for building Galois Sub-hierarchies. In: CLA 2012. (2012) 21–32
10. Prediger, S., Wille, R.: The Lattice of Concept Graphs of a Relationally Scaled Context. In: ICCS'99, Springer (1999) 401–414

11. Ferré, S., Ridoux, O., Sigonneau, B.: Arbitrary Relations in Formal Concept Analysis and Logical Information Systems. In: ICCS'05. Volume 3596 of LNCS., Springer (2005) 166–180
12. Arévalo, G., Falleri, J.R., Huchard, M., Nebut, C.: Building Abstractions in Class Models: Formal Concept Analysis in a Model-Driven Approach. In: MoDELS 2006. (2006) 513–527
13. Dolques, X., Huchard, M., Nebut, C., Reitz, P.: Fixing Generalization Defects in UML Use Case Diagrams. Fundam. Inform. **115**(4) (2012) 327–356
14. Moha, N., Hacene, A.R., Valtchev, P., Guéhéneuc, Y.G.: Refactorings of Design Defects Using Relational Concept Analysis. In: ICFCA 2008. (2008) 289–304
15. Saada, H., Dolques, X., Huchard, M., Nebut, C., Sahraoui, H.A.: Generation of Operational Transformation Rules from Examples of Model Transformations. In: MoDELS 2012. (2012) 546–561
16. Azmeh, Z., Driss, M., Hamoui, F., Huchard, M., Moha, N., Tibermacine, C.: Selection of Composable Web Services Driven by User Requirements. In: ICWS 2011. (2011) 395–402
17. Bendaoud, R., Napoli, A., Toussaint, Y.: Formal Concept Analysis: A unified framework for building and refining ontologies. In: EKAW 2008. LNCS 5268 (2008) 156–171
18. Rouane-Hacene, M., Valtchev, P., Nkambou, R.: Supporting Ontology Design through Large-Scale FCA-Based Ontology Restructuring. In: ICCS 2011. (2011) 257–269
19. Shi, L., Toussaint, Y., Napoli, A., Blansché, A.: Mining for Reengineering: an Application to Semantic Wikis using Formal and Relational Concept Analysis. In: ESWC'11. Volume 6644 of LNCS., Springer (2011) 421–435
20. Hitzler, P.: Default Reasoning over Domains and Concept Hierarchies. In: Proc. of KI 2004. Volume 3238 of LNCS., Springer Verlag (2004) 351–365
21. Osswald, R., Pedersen, W.: Induction of Classifications from Linguistic Data. In: Proc. of ECAI'02 Workshop. (July 2002)
22. Petersen, W.: A Set-Theoretical Approach for the Induction of Inheritance Hierarchies. In: ENTCS. Volume 51., Elsevier (July 2001)
23. Huchard, M., Dicky, H., Leblanc, H.: Galois Lattice as a Framework to specify Algorithms Building Class Hierarchies. Theoretical Informatics and Applications **34** (2000) 521–548
24. Ryssel, U., Ploennigs, J., Kabitzsch, K.: Extraction of feature models from formal contexts. In: SPLC 2011 Workshops. (2011) 4
25. Quinlan, J.: Induction of decision trees. Machine Learning (1986) 81–106
26. Clark, P., Boswell, R.: Rule induction with cn2: Some recent improvements. In: EWSL. (1991) 151–163
27. Lachiche, N.: Propositionalization. In Sammut, C., Webb, G.I., eds.: Encyclopedia of Machine Learning. Springer (2010) 812–817
28. Muggleton, S., de Raedt, L.: Inductive logic programming: Theory and methods. The Journal of Logic Programming **19**(20) (1994) 629–679
29. Doledec, S., Chessel, D., ter Braak, C., Champely, S.: Matching species traits to environmental variables: a new three-table ordination method. Environmental and Ecological Statistics **3** (1996) 143–166

Computing the Concept Lattice using Dendritical Neural Networks

David Ernesto Caro-Contreras, Andres Mendez-Vazquez

Centro de Investigación y Estudios Avanzados del Politécnico Nacional
Av. del Bosque 1145, colonia el Bajío, Zapopan , 45019, Jalisco, México.
`dcaro@gdl.cinvestav.mx,amendez@gdl.cinvestav.mx`

Abstract. Formal Concept Analysis (FCA) is a new and rich emerging discipline, and it provides efficient techniques and methods for efficient data analysis under the idea of "attributes". The main tool used in this area is the Concept Lattice also named Galois Lattice or Maximal Rectangle Lattice. A naive way to generate the Concept Lattice is by enumeration of each cluster of attributes. Unfortunately the numbers of clusters under the inclusion attribute relation has an exponential upper bound. In this work, we present a novel algorithm, PIRA (PIRA is a Recursive Acronym), for computing Concept Lattices in an elegant way. This task is achieved through the relation between maximal height and width rectangles, and maximal anti-chains. Then, using a dendritical neural network is possible to identify the maximal anti-chains in the lattice structure by means of maximal height or width rectangles.

Keywords: formal concept analysis, lattice generation, neural networks, dendrites, maximal rectangles.

1 Introduction

Formal Concept Analysis (FCA) refers to a mathematized and formal human-centered way to analyze data. It can be described as an ontology-based Lattice Theory. This theory describes ways of visualizing patterns, generate implications, and representing generalizations and dependencies [1]. The theory makes use of a binary relation between objects and attributes, which plays a fundamental role in the understanding of human model conceptualization. FCA formalizes all these ideas, and gives a mathematical framework to work on them [2].

The Concept Lattice is mainly used to represent and describe hierarchies between data clusters (formal concepts or classes), which are inherent in the perceived information. A formal concept can be regarded also as a data cluster, in which certain attributes are all shared by a set of objects. The Concept Lattice is the main subject of the theory of FCA, and it was first introduced in the early eighties by Rudolf Wille [2, 3]. Over time, FCA has become a powerful theory for many interesting applications. Examples of these are in the data analysis of Frequent Closed Itemsets [4], association rules and Functional/Conditional hierarchies discovery [5, 6, 7]. For all these reasons, concept lattice generation is an important topic in FCA research [8, 9, 10, 11, 12, 13, 14, 15]. However, the main drawback of concept lattices generation is the exponential size of the concept lattice and its generation complexity [16].

In this work, we present a Morphological Neural Network based method to generate the Concept Lattice. This batch method is capable of generating the Hasse diagram, and it is a bottom-up generation algorithm. With that in mind, this work is organized as follows. Section 2 is an elementary description of FCA Theory. Next, section 3 is a short introduction on how the Single Lattice Layer Perceptron (SLLP) works. In section 4, a description on how to compute the Maximal Rectangles is given. In this section, we also define a link between SLLP and the design of a classifier for maximal anti-chains search. Section 5 shows a comparison between PIRA and other known algorithms.

2 Basic Definitions

2.1 Formal Concept Analysis (FCA)

First, it is necessary to define the concept of a formal context [3].

Definition 1. *A binary formal context* $\mathbb{K}$ *is a triple* (G, M, I)*. In this mathematical structure,* G *and* M *are two finite sets, called objects and attributes respectively, and* $I \subseteq G \times M$ *is a binary relation over* G *and* M*, named the incidence of* $\mathbb{K}$*.*

In order to define formal concepts of the formal context (G,M,I), it is necessary to define two derivation operators, named Galois connectors.

Definition 2. *For an arbitrary subsets* $A \subseteq G$ *and* $B \subseteq M$*:*

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

These two derivation operators satisfy the following three conditions over arbitrary subsets $A_1, A_2 \subseteq G$ and $B_1, B_2 \subseteq M$:

1. $A_1 \subseteq A_2$ then $A_2' \subseteq A_1'$ dually $B_1 \subseteq B_2$ then $B_2' \subseteq B_1'$
2. $A_1 \subseteq A_1''$ and $A_1' = A_1'''$ dually $B_1 \subseteq B_1''$ and $B_1' = B_1'''$
3. $A_1 \subseteq B_1' \iff B_1 \subseteq A_1'$

Next, the definition of a formal concept idea, which represents the building unit of FCA.

Definition 3. *Let* $\mathbb{K}$ *be a formal context,* $\mathbb{K} := (G, M, I)$*. A formal concept* C *of* $\mathbb{K}$ *is defined as a pair* $C = (A, B)$*,* $A \subseteq G$ *and* $B \subseteq M$*, where the following conditions are satisfied,* $A = B'$ *and* $A' = B$*, where* A *is named the extent and* B *is named the intent of the formal concept* (A, B)*.*

Next, we will show some useful notions from Lattice Theory [17, 18] to understand the algebraic structure generated by those derivation operators and the formal concept idea.

Definition 4. *A Partially Ordered Set (Poset) is a set X in which there is a binary relation between elements of X, $\leq$, with the following properties:*

1. $\forall x,\ x \leq x$ (Reflexive).
2. if $x \leq y$ and $y \leq x$, then $x = y$ (Antisymmetric).
3. if $x \leq y$ and $y \leq z$, then $x \leq z$ (Transitive).

Formal Concepts can be partially ordered by inclusion. And for any pair C_i, C_j have a unique greatest lower bound and a unique least upper bound. Then, the set of all formal concept in $\mathbb{K}$, ordered by inclusion, is known as a Concept Lattice. Next definition links some definitions with the Formal Concept notion.

Definition 5. *(Rectangles in $\mathbb{K}$). Let $A \in G$ and $B \in M$, a rectangle in $\mathbb{K}$ is a pair (A, B) such that $A \times B \subseteq I$.*

Given the set of Rectangles in $\mathbb{K}$, a special kind of rectangles are defined as follows:

Definition 6. *(Maximal Rectangles). A rectangle (A_1, B_1) is maximal if and only if there does not exist another valid rectangle (A_2, B_2) in $\mathbb{K}$ such that $A_1 \subseteq A_2$ or $B_1 \subseteq B_2$.*

From here, we have the formal concept as a maximal rectangle.

Theorem 1. *(A, B) is a formal concept of $\mathbb{K}$ if and only if (A, B) is a maximal rectangle in $\mathbb{K}$.*

Now, the following definitions are going to be useful for the proposed bottom-up approach of FCA generation.

Definition 7. *Let $\mathbb{K} := (G, M, I)$ and $A \subseteq G$. A is said to be an object derivative anti-chain set if and only if $A'_1 \nsubseteq A'_2$ and $A'_2 \nsubseteq A'_1$ for any two distinct $A_1, A_2 \in A$.* Dually, for $B \subset M$ nd $B'_1 \nsubseteq B'_2$ for any two distinct B'_1, B'_2 in B.

We will denote as D a derivative anti-chain and as $\mathfrak{A}(\mathbb{K})$ the set of all antichains in $\mathbb{K}$. All sets in which the super/subconcept order is not satisfied for any distinct elements of the set is called anti-chain set.

Definition 8. *D^+ is a maximal derivative anti-chain iff there does not exist other $D \in \mathfrak{A}(\mathbb{K})$ such that $D \supset D^+$. There exists a maximal derivative anti-chain for objects and one for attributes.*

Finally, we use a simple upward closed set definition.

Definition 9. *Upward closed set. Let $(L, \leq)$ be a poset P. A set $S \subseteq L$ is said to be upward closed if $\forall x, y \in S$ with $y \geq x$ implies that $y \in S$.*

Using these definitions of anti-chains, maximal rectangles and upward closed set, it is possible to device a bottom-up approach by means of dendritic neuronal networks.

3 Lattice-Based Neural Networks (LBNN).

Artificial Neural Networks (ANN) are models of learning and automatic processing inspired by the nervous system. The features of ANN make them quite suitable for applications where there is no prior pattern that can be identified, and there is only a basic set of input examples (previously classified or not). They are also highly robust to noise, failure of elements in the ANN, and, finally, they are parallelizable. As we said early, our work is related with LBNN, which is also considered an Artificial Neural Network, and are inspired in recent advances in the neurobiology and biophysics of neural computation **(author?)** [19, 20].

The theory of LBNN is actively used in classification [21, 20, 22], clustering [23], associative memories [24, 25], fuzzy logic [26], among others[27, 28]. Basically, in the LBNN model, an input layer receives external data, and subsequent layers perform the necessary functions to generate the desired outputs. Single Lattice Layer Perceptrons (SLLP), also named Dendritic Single Layer Perceptron, are basically a classifier in which there exists a set of input neurons, a set of output neurons, and a set of dendrites growing from the output neurons. Those dendrites are connected with the input set by some axons from those input neurons. A training set configures those outputs based on the maxima $\bigvee$ and minima $\bigwedge$ operations derived from the morphological algebra $(\mathbb{R}, +, \bigvee, \bigwedge)$. FCA and Mathematical Morphology common algebraic framework: Erosion, dilatation, morphological operators, valuations, an many other functions in concept lattices have been previously studied [29].

In SLLP, a set of n input neurons $N_1, \ldots, N_n$ accepts input $x = (x_1, ..., x_n) \in \mathbb{R}^n$. An input neuron provides information through axonal terminals to the dendritic trees of output neurons. A set of O output neurons is represented by $O_1, ..., O_m$. The weight of an axonal terminal of neuron N_i connected to the k_{th} dendrite of the O_j output neuron is denoted by w^{ℓ}_{ijk}, in which, the superscript $\ell \in \{0, 1\}$ represents an excitatory $\ell = 1$ or a inhibitory $\ell = 0$ input to the dendrite. The k_{th} dendrite of O_j will respond to the total value received from the N input neurons set, and it will accept or reject the given input. Dendrite computation is the most important operation in LBNN. The following equation $\tau^j_k(x)$, from SLLP, corresponds to the computation of the k_{th} dendrite of the j_{st} output neuron[21].

$$\tau^j_k(x) = p_{jk} \bigwedge_{i \in I(k)} \bigwedge_{\ell \in L} (-1)^{1-\ell}(x_i + w^{\ell}_{ijk})$$

Where x is the input value of neurons $N_1, ..., N_n$ and x_i is the the value of the input neuron N_i. $I(k) \subseteq 1, ..., n$ represents the set of all input neurons with synaptic connection on the k_{th} dendrite of O_j. The number of terminal axonal fibers on N_i that synapse on a dendrite of O_j is at most two, since $\ell(i) \subseteq \{0, 1\}$. Finally, the last involved element is $p_{jk} \in \{-1, 1\}$ and it denotes the excitatory ($p_{jk} = 1$) or inhibitory ($p_{jk} = -1$) response of the k_{th} dendrite of O_j to the received input. All the values $\tau^j_k(x)$ are passed to the neuron cell body. The value computation of O_j is a function that computes all its dendrites values. The total value received by O_j is given by[21]:

$$\tau^j(x) = p_j * \bigvee_{k=1}^{K_j} \tau_k^j(x)$$

In this SLLP model, K_j is the set of all dendrites of O_j, $p_j = \pm 1$ represents the response of the cell body to the received input vector. At this point, we know that $p_j = 1$ means that the input is accepted and $p_j = -1$ means that the cell body rejects the received input vector. The last statement related with O_j correspond to an activation function f, namely $y_j = f\left[\tau^j(x)\right]$.

$$f\left[\tau^j(x)\right] = \begin{cases} 1 \iff \tau(x) \geq 0 \\ 0 \iff \tau(x) < 0 \end{cases} \tag{1}$$

As, we mention early, dendrites configurations is computed using a training set. For this, they use merge and elimination methods [17].

4 PIRA Algorithm.

In PIRA-LBNN model for finding upward-closed set elements, a set of binary patterns are represented by G'. Thus, the binary representation of a formal context is itself a set of patterns, in which the derivative of each object is an element $x \in G'$. Then, we can define each element $x = (x_1, ..., x_n) \in G'$ as a binary vector. This allows us to define a simple class classification rule to find maximal rectangles, and in an equivalent way the maximal anti-chains.

In PIRA algorithm we search all the rectangles with maximum width or height from each formal concept founded. There are two ways to achieve our goal. It means that $\ell = 1$ or $\ell = 0$ depending on whether it is calculating, a supremum or an infimum, by using excitatory or inhibitory dendrites. Thus, p_{jk} is also a constant, $p_{jk} = 1$ or $p_{jk} = -1$ and it denotes the excitatory or inhibitory response of the k_{th} dendrite of M_j to the received input, and another remarkable statement is the fact that we only need to connect zeros as axonal branches. The simplest way to compute the value of the k_{th} dendrite derived from the SLLP equations, is:

$$\tau_k^j(x) = \bigvee_{i \in I(k)} (x_i) \tag{2}$$

This equation, where $\tau_k^j(x)$ is the value of the computation of the k_{th} dendrite of the j_{th} output neuron given a input x, $I(k) \subseteq \{1, ..., n\}$ is the set of input neurons with terminal fibers that synapse the k_{th} dendrite of our output neuron. We realize that all weights w_{ijk}^{ℓ} are equal to zero, this is, for our upward closed set classifier, we only need to store zero values from the input patterns in the training step. Our goal is that the output of our classification neuron be $x \in C_1$

Algorithm 1 addDendrite

```
INPUT: NeuralNetwork P, Pattern x
OUTPUT: Updated P
  Dendrite k = addNewDendrite(P)
  FOR EACH element in x
    IF getValue(element) = 0
      i = getPosition(element)
      addAxonalBranch(k, i)
    END
  END
END
```

if the input $x \in D^+$. The training step is to find the set D^+. This is achieved by processing elements from higher to lower cardinality,

Specifically, each dendrite k corresponds with one maximal antichain intent to be tested, $I(k)$ is the incidence set of positions where the value of the maximal rectangle is zero for the pattern represented by the k dendrite. We get the state value of M_j computing the minimum value of all it's dendrites. Again, as the SLLP, each $\tau_k^j(x)$ is computed, and it is passed to the cell body of M_j. Then we can get the total value received by our output neuron as follows:

$$\tau^j(x) = \bigwedge \tau_k^j(x) \tag{3}$$

We realize that the activation function is not required since $f\left[\tau^j(x)\right] = \tau^j(x)$ where $\tau^j(x) = 1$ if $x \nleq y$ for all $y \in C_1$ and $\tau^j(x) = 0$ if $x \leq y$ for some $y \in C_1$. As we mentioned above x is a maximal rectangles if and only if $x \nleq y$ and $y \nleq x$ for all $y \in C_1$. Using the previous statement we can ensure that half of the work is done, and the second test, $y \nleq x$, will be performed by processing data in a particular order. In our case, we use cardinality order ensuring that each new computed row is not a superset of the previously computed rows.

As we said before, the idea is to use our LBNN structure to classify maximal rectangles. When we start computing a formal concept, our structure is empty, this means there are not dendrites or axonal connections. So, the first step is to add each element with the maximal cardinality as a pattern to learn. Algorithm 1 shows how an element is added to our LBNN for maximal rectangles learning. First, algorithm 1 receives as parameter the LBNN which is being trained and a binary vector. As we will see below, this binary vector has been proven as an antichain element. Algorithm 1, first grows a new dendrite k_{th} in our output neuron O_j. Every column in x is checked, if that property is not contained by the object x, then an axonal branch grows from the i position of the input neuron set to the new dendrite. This operation is represented by addAxonalBranch calling. We can assure that we will not misclassify formal concepts in the processed context. The algorithm 2 shows how to compute the Concept Lattice by computing Maximal Antichain Sets recursively.

Algorithm 2 Compute Maximal Rectangles

```
INPUT: A Binary Context K:(G,M,I), K–Supremum, K–Infimum,
    Lattice
OUTPUT: Intent HashSet Maximal_Rectangles
STEP 1:
  Init:
  LBNN Upward Structure
  Maximal_Rectangles
STEP 2:
  Sort G by derivative higher to lower Cardinality
STEP 3:
  IF maximal cardinality is equal to K–Supremum intention
      cardinality
    Add Link from K–Supremum to K–Infimum
    RETURN
  Add, as positive dendrites, all elements with maximal
      cardinality in G' to LBNN.
STEP 4:
  Foreach remaining element in G
    IF Maximal_Rectangles does not contains element' AND
        Upward evaluation element' is 1 then:
      add, as dendrite, element to LBNN
      create Formal Concept with:
        element union K–infimum as extent
        element' as intent
      add this new formal concept to:
        Maximal_Rectangles
    OTHER IF Maximal_Rectangles contains element
      add element to previously Formal Concept created.
STEP 5:
  Foreach Rectangle in Maximal_Rectangles
    IF Lattice does not contains Rectangle
      add Rectangle to Lattice
      add a Link from Rectangle to K–Infimum
      Compute Maximal Rectangles with:
        G = G / Rectangle extent
        M = Rectangle intent
        I = Projection
        K–Supremum
        Rectangle as a K–Infimum
        Lattice
    ELSE
      add a Link from previously created Rectangle in
          Global_Maximal_Rectangles to K–Infimum
```

Algorithm 3 Main Function

```
INPUT: BinaryContext(G,M,I)
OUTPUT: Lattice L
Step 1: Get Maximum and Infimum elements.
  FormalConcept max = getMax(G,M,I)
  FormalConcept min = getMin(G,M,I)
  addConcept(L,max), addConcept(L,min)
STEP 2: Get maximal Rectangles From min
  MaxRectangles = Compute Maximal Rectangles with:
    G/min.extent,
    min.intent,I-Proj,
    max,
    min,
    L
```

In algorithm **2**, the first step creates a new dendritic neural network. It is initialized with one output neuron, $n = |intent|$ and $k = 0$, where the number of input neurons is n, and each input neuron represents one attribute element in M. The second step is used to get the G' set ordered by attribute cardinality. Next, in a third step, if the maximal cardinality of the elements in G' is equal to the Supremum intent cardinality, it adds a link between Supremum and Infimum, and stops. Otherwise, it adds all the elements in G' with the maximal cardinality, to the dendritical neural network structure. Those elements are maximal rectangles in the given binary context. The fourth step is used to check the remaining elements. If an element derivative does not exist already, as an intent, and the evaluation 3 says that it is a maximal rectangle, then, it adds the object and its derivative as a new maximal rectangle. Otherwise, if the element derivative already exists, it is added to the previously formal concept as its extent. In the last step, it processes each maximal rectangle that has been found. If that element is already contained in the lattice, it adds a link between Infimum and the previous element created in the lattice. Otherwise, it adds that link and processes that formal concept recursively. Then, it adds this new element to the lattice structure.

At this point, it computes all the concepts given by the binary context, but $\mathfrak{B}(\mathbb{K})$ cardinality is bounded by an exponential complexity. A simple way to avoid this issue is to use a Binary Tree to store and recover all elements in$\mathfrak{B}(\mathbb{K})$. Basically, this binary tree works as a hash function using the concept of intent in their binary representation. Then, it searches, finds and adds any intent in $|M|$ steps. In addition, the processing order enables us to generate the edges of the Hasse diagram without additional computing steps. Therefore, the process of generating the concept lattice and Hasse diagram presented in this paper has an expected polynomial delay time [16] for maximal anti-chains search.

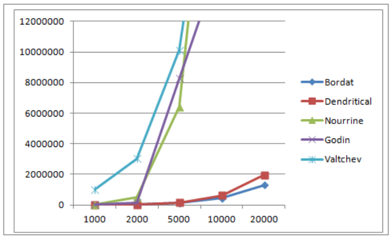
5 Computational Experiments.

As shown in [16], many parameters are involved in the performance and running time of an algorithm. For our tests, we considered the number of objects, the number of attributes, the density of the context and the worst case for contexts of $M \times M$. Where density is the least percentaje of binary relations for each object in $\mathbb{K}$. Those parameters were tested independently. Four algorithms were selected to compare the Dendritical algorithm performance. Implementations were performed in java.

Figure 1 shows the execution time behavior for a formal context with 10% density and 100 attributes. Here, the V-axis represent the running time of each algorithm in ms, and the H-axis the total number of objects under constant number of attributes. The test was performed by increasing the number of objects from one thousand to twenty thousand, with a random distribution and 10% of density.

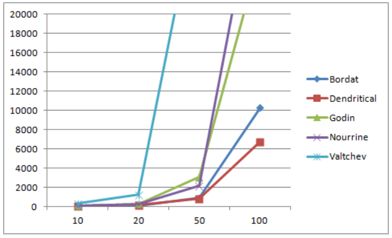
We can verify that Bordat algorithm and our algorithm have the best performance when the number of objects increase with respect to the attributes.

Figure 1. Growing objects number.



Execution time when the number of objects grows from 1000 to 20000, with 100 attributes and 10% of density.

Figure 2. Growing attributes number

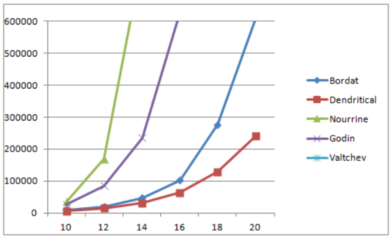


Execution time when the number of attributes from 10 to 100 with 1000 objects and 10% density case.

Figure 2 shows the increase in execution time when the number of attributes increases. All datasets for this test has a 10% density and 1000 objects, and the number of attributes is increasing from ten to one hundred, and, the number of formal concepts is growing too. Here we can notice that Bordat and Dendritical algorithms, are faster than Godin, Nourrine and Valtchev algorithms. We can also notice that Dendritical execution time behavior is faster when the number of objects grows.

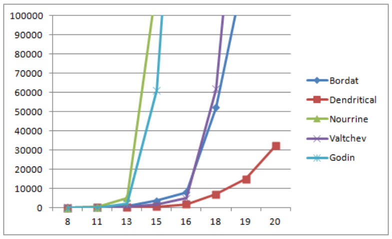
Figure 3 shows the increase in execution time when the density percentage becomes higher. All datasets has 1000 objects and 100 attributes and when the density grows the number of formal concepts grows too. As we mention, density is mainly the percentage of attributes for all the objects in the formal context.

Figure 3. Growing density percentage



Execution time when density increases from 10% to 20%. 1000 objects and 100 attributes case.

Figure 4. MxM worst case contexts



Algorithms running diagonal contexts in which number of attributes are growing from 8 to 20 attributes.

Here, we notice that when the density grows Dendritical is closer and even faster than the other algorithms. Figure 4 shows running time for zero diagonal contexts where |G| = |M| and yields the complete lattice, which means $2^{|M|}$ formal concepts. In this kind of formal contexts, we can see a clear running time superiority of the dendritical algorithm. Finally, table 1 shows some examples of time complexity at each algorithm step.

Table 1. Execution time examples

No. Obj	**No. Att**	**Density**	**Concepts**	**Query Time**	**Ordering**	**Dendritical**	**Total**
1000	*36*	*17*	*8377*	722ms	20ms	49ms	791ms
1000	*49*	*14*	*14190*	924ms	67ms	101ms	1092ms
1000	*81*	*11*	*26065*	1853ms	124ms	201ms	2178ms

Algorithms time when density becomes higher and the number of attributes become lower. Those tests shows the increase in execution when attributes and $I \subseteq G \times M$ are modified.

6 Conclusion

In this paper, we presented an algorithm based on the main idea of maximal rectangles, using the cardinality notion and the dendritical classifier. We have also compared it with some known algorithms for the construction of Concept Lattices.

From the tests presented in this paper, we can see that as the number of objects grows, our algorithm time execution is higher than Bordat or other methods, but it could be a better choice when the density or the number of attributes is high. Also, the results shows a good performance for the $M \times M$ contexts in the worst case scenario, which demonstrates the feasibility of our algorithm on some kinds of datsets.

Bibliography

[1] Belohlavek, R., Beydoun, G.: Formal Concept Analysis With Background Knowledge: Attribute Priorities. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews **5465** (2009) 109–117

[2] Hereth, J., Stumme, G., Wille, R., Wille, U.: Conceptual knowledge discovery: a human-centered approach. Journal of Applied Artificial Intelligence **17** (2003) 281–302

[3] Wille, R.: Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts. In: ICFCA. (2009) 314–339

[4] Valtchev, P., Missaoui, R., Godin, R., Meridji, M.: Generating frequent itemsets incrementally: two novel approaches based on Galois lattice theory. J. Exp. Theor. Artif. Intell. **14**(2-3) (2002) 115–142

[5] Maddouri, M.: A Formal Concept Analysis Approach to Discover Association Rules from Data. R. Belohlavek. V. Snasel CLA **1** (2005) 10–21

[6] Agrawal, R.: Fast algorithms for mining association rules in large databases. Proceedings of the 20th International Conference on Very Large Data Bases, **1** (1994) 487–499

[7] Lakhal, L., S.G.: Efficient Mining of Association Rules Based on Formal Concept Analysis. Ganter et al. Springer. **LNAI 3626** (2005) 180–195

[8] Merwe, D., Obiedkov, S., Kourie, D.: AddIntent: A New Incremental Algorithm for Constructing Concept Lattices. In Eklund, P., ed.: Concept Lattices. Volume 2961 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2004) 372–385

[9] Lv, L., Zhang, L., Jia, P., Zhou, F.: A Bottom-Up Incremental Algorithm of Building Concept Lattice. In Wu, Y., ed.: Software Engineering and Knowledge Engineering: Theory and Practice. Volume 115 of Advances in Intelligent and Soft Computing. Springer Berlin Heidelberg (2012) 91–98

[10] Valtchev, P., Missaoui, R., Lebrun, P.: A partition-based approach towards constructing Galois (concept) lattices. Discrete Math. **256**(3) (2002) 801–829

[11] Nourine, L., Raynaud, O.: A Fast Algorithm for Building Lattices. Inf. Process. Lett. **71**(5-6) (1999) 199–204

[12] Nourine, L., Raynaud, O.: A fast incremental algorithm for building lattices. J. Exp. Theor. Artif. Intell. **14**(2-3) (2002) 217–227

[13] Farach-Colton, M., Huang, Y.: A Linear Delay Algorithm for Building Concept Lattices. In Ferragina, P., Landau, G., eds.: Combinatorial Pattern Matching. Volume 5029 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2008) 204–216

[14] Godin, R., Missaoui, R., Alaoui, H.: Incremental concept formation algorithms based on Galois (Concept) lattices. Computational Intelligence **11**(2) (1995) 246–267

[15] Ganter, B.: Two basic algorithms in concept analysis. In: Proceedings of the 8th international conference on Formal Concept Analysis. ICFCA-10, Berlin, Heidelberg, Springer-Verlag (2010) 312–340

[16] Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental and Theoretical Artificial Intelligence **14**(2-3) (2002) 189–216

[17] Kaburlasos, V.G., Ritter, G.X.: Computational Intelligence Based on Lattice Theory. Volume 67 of Studies in Computational Intelligence. Springer (2007)

[18] B. A. Davey, H.A.P.: Introduction to lattices and orders. 2nd edn. Press Sindicate H. Cambridge University (2002)

[19] Ritter, G.X., Iancu, L., Urcid, G.: Neurons, Dendrites, and Pattern Classification. In: CIARP. (2003) 1–16

[20] Urcid, G., Ritter, G.X., Iancu, L.: Single Layer Morphological Perceptron Solution to the N-Bit Parity Problem. In: CIARP. (2004) 171–178

[21] Ritter, G., Urcid, G.: Learning in Lattice Neural Networks that Employ Dendritic Computing. In Kaburlasos, V., Ritter, G., eds.: Computational Intelligence Based on Lattice Theory. Volume 67 of Studies in Computational Intelligence. Springer Berlin / Heidelberg (2007) 25–44

[22] Barmpoutis, A., Ritter, G.X.: Orthonormal Basis Latice Neural Networks. In Computational Intelligence Based on LatticeTheory, V. Kaburlasos and G. X. Ritter **1** (Springer-Verlag, Heidelberg, Germany, 2007) 43–56

[23] Kaburlasos, V.: Granular Enhancement of Fuzzy ART-SOM Neural Classifiers Based on Lattice Theory. In Kaburlasos, V., Gerhard, R., eds.: Computational Intelligence Based on Lattice Theory. Volume 67 of Studies in Computational Intelligence. Springer Berlin / Heidelberg (2007) 3–23

[24] Aldape-Perez, M., Yanez-Marquez, C., Camacho-Nieto, O., J.Arguelles-Cruz, A.: An associative memory approach to medical decision support systems. Comput. Methods Prog. Biomed. **106**(3) (2012) 287–307

[25] Ritter, G.X., Chyzhyk, D., Urcid, G., Grana, M.: A Novel Lattice Associative Memory Based on Dendritic Computing. In: HAIS. (2012) 491–502

[26] Kaburlasos, V.G., P.V.: Fuzzy lattice neurocomputing (fln) models. Neural Networks **13 (10)** (2000) 1145–1170.

[27] Witte, V., Schulte, S., Nachtegael, M., Malange, T., Kerre, E.: A Lattice-Based Approach to Mathematical Morphology for Greyscale and Colour Images. In Kaburlasos, V., Ritter, G., eds.: Computational Intelligence Based on Lattice Theory. Volume 67. Springer Berlin / Heidelberg (2007) 129–148

[28] Urcid, G., Nieves-Vazquez, J.A., Garcia-A., A., Valdiviezo-N., J.C.: Robust image retrieval from noisy inputs using lattice associative memories. In: Image Processing: Algorithms and Systems. (2009)

[29] Atif, J., B.I.D.F.H.C.: Mathematical morphology operators over concept lattices. . In: Cellier, P., Distel, F., Ganter, B. (Eds.) ICFCA 2013, (Springer-Verlag Berlin Heidelberg) **LNAI 7880** (2013) 28–43.

Isotone L-bonds

Jan Konecny[1] and Manuel Ojeda-Aciego[2]

[1] University Palacky Olomouc, Czech Republic[⋆]
[2] Dept. Matemática Aplicada, Univ. Málaga, Spain[⋆⋆]

Abstract. L-bonds represent relationships between formal contexts. We study properties of these intercontextual structures w.r.t. isotone concept-forming operators in fuzzy setting. We also focus on the direct product of two formal fuzzy contexts and show conditions under which a bond can be obtained as an intent of the product. In addition, we show that the previously studied properties of their antitone counterparts can be easily derived from the present results.

1 Introduction

Formal Concept Analysis (FCA) has become a very active research topic, both theoretical and practical, for formally describing structural and hierarchical properties of data with "object-attribute" character. It is this wide applicability which 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.

A number of different approaches have presented towards a generalization of the framework and scope of FCA and, nowadays, one can find papers which extend the theory by using ideas from rough set theory [21, 15, 14], possibility theory [8], fuzzy set theory [1, 2], the multi-adjoint framework [16, 17, 19] or heterogeneous approaches [6, 18].

Goguen argued in [10] that *concepts* should be studied transversally, transcending the natural boundaries between sciences and humanities, and proposed category theory as a unifying language capable of merging different apparently disparate approaches. Krotzsch [13] suggested a categorical treatment of morphisms, understood as fundamental structuring blocks, in order to model, among other applications, data translation, communication, and distributed computing.

In this paper, we deal with an extremely general form of L-fuzzy Formal Concept Analysis, based on categorical constructs and L-fuzzy sets. Particularly, our approach originated in relation to a previous work [12] on the notion of Chu correspondences between formal contexts, which led to obtaining information about the structure of L-bonds in such a generalized framework.

[⋆] Supported by the ESF project No. CZ.1.07/2.3.00/20.0059, the project is cofinanced by the European Social Fund and the state budget of the Czech Republic.

[⋆⋆] Supported by Spanish Ministry of Science and FEDER funds through project TIN09-14562-C05-01 and Junta de Andalucía project P09-FQM-5233.

In this paper, we study properties of L-bonds w.r.t. isotone concept-forming operators in a fuzzy setting. We also focus on the direct product of two formal fuzzy contexts and show conditions under which a bond can be obtained as an extent of the product. In addition, we show that the previously studied properties of their antitone counterparts can be easily derived from the present results.

2 Preliminaries

We use complete residuated lattices as basic structures of truth-degrees. A complete residuated lattice is a structure $\mathbf{L} = \langle L, \wedge, \vee, \otimes, \rightarrow, 0, 1\rangle$ such that

(i) $\langle L, \wedge, \vee, 0, 1\rangle$ is a complete lattice, i.e. a partially ordered set in which arbitrary infima and suprema exist;
(ii) $\langle L, \otimes, 1\rangle$ is a commutative monoid, i.e. $\otimes$ is a binary operation which is commutative, associative, and $a \otimes 1 = a$ for each $a \in L$;
(iii) $\otimes$ and $\rightarrow$ satisfy adjointness, i.e. $a \otimes b \leqslant c$ iff $a \leqslant b \rightarrow c$.

0 and 1 denote the least and greatest elements. The partial order of $\mathbf{L}$ is denoted by $\leqslant$. Throughout this work, $\mathbf{L}$ denotes an arbitrary complete residuated lattice.

Elements a of L are called truth degrees. Operations $\otimes$ (multiplication) and $\rightarrow$ (residuum) play the role of (truth functions of) "fuzzy conjunction" and "fuzzy implication". Furthermore, we define the complement of $a \in L$ as

$$\neg a = a \rightarrow 0 \tag{1}$$

An $\mathbf{L}$-set (or fuzzy set) A in a universe set X is a mapping assigning to each $x \in X$ some truth degree $A(x) \in L$ where L is a support of a complete residuated lattice. The set of all $\mathbf{L}$-sets in a universe X is denoted L^X, or $\mathbf{L}^X$ if the structure of $\mathbf{L}$ is to be emphasized.

The operations with $\mathbf{L}$-sets are defined componentwise. For instance, the intersection of $\mathbf{L}$-sets $A, B \in L^X$ is an $\mathbf{L}$-set $A \cap B$ in X such that $(A \cap B)(x) = A(x) \wedge B(x)$ for each $x \in X$, etc. An $\mathbf{L}$-set $A \in L^X$ is also denoted $\{{}^{A(x)}\!/x \mid x \in X\}$. If for all $y \in X$ distinct from $x_1, x_2, \ldots, x_n$ we have $A(y) = 0$, we also write

$$\{{}^{A(x_1)}\!/x_1, {}^{A(x_2)}\!/x_1, \ldots, {}^{A(x_n)}\!/x_n\}.$$

An $\mathbf{L}$-set $A \in L^X$ is called crisp if $A(x) \in \{0, 1\}$ for each $x \in X$. Crisp $\mathbf{L}$-sets can be identified with ordinary sets. For a crisp A, we also write $x \in A$ for $A(x) = 1$ and $x \notin A$ for $A(x) = 0$. An $\mathbf{L}$-set $A \in L^X$ is called empty (denoted by $\emptyset$) if $A(x) = 0$ for each $x \in X$. For $a \in L$ and $A \in L^X$, the $\mathbf{L}$-sets $a \otimes A \in L^X$, $a \rightarrow A$,$A \rightarrow a$, and $\neg A$ in X are defined by

$$(a \otimes A)(x) = a \otimes A(x), \tag{2}$$
$$(a \rightarrow A)(x) = a \rightarrow A(x), \tag{3}$$
$$(A \rightarrow a)(x) = A(x) \rightarrow a, \tag{4}$$
$$\neg A(x) = A(x) \rightarrow 0. \tag{5}$$

An **L**-set $A \in L^X$ is called an a-complement if $A = a \rightarrow B$ for some $B \in L^X$.

Binary **L**-relations (binary fuzzy relations) between X and Y can be thought of as **L**-sets in the universe $X \times Y$. That is, a binary **L**-relation $I \in L^{X \times Y}$ between a set X and a set Y is a mapping assigning to each $x \in X$ and each $y \in Y$ a truth degree $I(x, y) \in L$ (a degree to which x and y are related by I).

The composition operators are defined by

$$(A \circ B)(x, y) = \bigvee_{f \in F} A(x, f) \otimes B(f, y), \tag{6}$$

$$(A \triangleleft B)(x, y) = \bigwedge_{f \in F} A(x, f) \rightarrow B(f, y), \tag{7}$$

$$(A \triangleright B)(x, y) = \bigwedge_{f \in F} B(f, y) \rightarrow A(x, f). \tag{8}$$

An **L**-context is a triplet $\langle X, Y, I \rangle$ where X and Y are (ordinary) sets and $I \in L^{X \times Y}$ is an **L**-relation between X and Y. Elements of X are called objects, elements of Y are called attributes, I is called an incidence relation. $I(x, y) = a$ is read: "The object x has the attribute y to degree a."

Consider the following pairs of operators induced by an **L**-context $\langle X, Y, I \rangle$. First, the pair $\langle \uparrow, \downarrow \rangle$ of operators ${}^{\uparrow} : L^X \rightarrow L^Y$ and ${}^{\downarrow} : L^Y \rightarrow L^X$ is defined by

$$C^{\uparrow}(y) = \bigwedge_{x \in X} C(x) \rightarrow I(x, y), \quad D^{\downarrow}(x) = \bigwedge_{y \in Y} D(y) \rightarrow I(x, y). \tag{9}$$

Second, the pair $\langle \cap, \cup \rangle$ of operators ${}^{\cap} : L^X \rightarrow L^Y$ and ${}^{\cup} : L^Y \rightarrow L^X$ is defined by

$$C^{\cap}(y) = \bigvee_{x \in X} C(x) \otimes I(x, y), \quad D^{\cup}(x) = \bigwedge_{y \in Y} I(x, y) \rightarrow D(y), \tag{10}$$

Third, the pair $\langle {}^{\wedge}, {}^{\vee} \rangle$ of operators ${}^{\wedge} : L^X \rightarrow L^Y$ and ${}^{\vee} : L^Y \rightarrow L^X$ is defined by

$$C^{\wedge}(y) = \bigwedge_{x \in X} I(x, y) \rightarrow C(x), \quad D^{\vee}(x) = \bigvee_{y \in Y} D(y) \otimes I(x, y), \tag{11}$$

for $C \in L^X$, $D \in L^Y$.

To emphasize that the operators are induced by I, we also denote the operators by $\langle \uparrow_I, \downarrow_I \rangle$, $\langle \cap_I, \cup_I \rangle$, and $\langle \wedge_I, \vee_I \rangle$. Furthermore, denote the corresponding sets of fixpoints by $\mathcal{B}(X^{\uparrow}, Y^{\downarrow}, I)$, $\mathcal{B}(X^{\cap}, Y^{\cup}, I)$, and $\mathcal{B}(X^{\wedge}, Y^{\vee}, I)$, i.e.

$$\mathcal{B}(X^{\uparrow}, Y^{\downarrow}, I) = \{\langle C, D \rangle \in L^X \times L^Y \mid C^{\uparrow} = D,\, D^{\downarrow} = C\},$$
$$\mathcal{B}(X^{\cap}, Y^{\cup}, I) = \{\langle C, D \rangle \in L^X \times L^Y \mid C^{\cap} = D,\, D^{\cup} = C\},$$
$$\mathcal{B}(X^{\wedge}, Y^{\vee}, I) = \{\langle C, D \rangle \in L^X \times L^Y \mid C^{\wedge} = D,\, D^{\vee} = C\}.$$

The sets of fixpoints are complete lattices, called **L**-concept lattices associated to I, and their elements are called formal concepts.

For a concept lattice $\mathcal{B}(X^{\vartriangle}, Y^{\triangledown}, I)$, where $\langle\vartriangle, \triangledown\rangle$ is either of $\langle\uparrow, \downarrow\rangle$, $\langle\cap, \cup\rangle$, or $\langle\wedge, \vee\rangle$, denote the corresponding sets of extents and intents by $\mathrm{Ext}(X^{\vartriangle}, Y^{\triangledown}, I)$ and $\mathrm{Int}(X^{\vartriangle}, Y^{\triangledown}, I)$. That is,

$$\mathrm{Ext}(X^{\vartriangle}, Y^{\triangledown}, I) = \{C \in L^X \mid \langle C, D\rangle \in \mathcal{B}(X^{\vartriangle}, Y^{\triangledown}, I) \text{ for some } D\},$$
$$\mathrm{Int}(X^{\vartriangle}, Y^{\triangledown}, I) = \{D \in L^Y \mid \langle C, D\rangle \in \mathcal{B}(X^{\vartriangle}, Y^{\triangledown}, I) \text{ for some } C\},$$

A system of **L**-sets $V \subseteq L^X$ is called an **L**-*interior system* if

- V is closed under $\otimes$-multiplication, i.e. for every $a \in L$ and $C \in V$ we have $a \otimes C \in V$;
- V is closed under union, i.e. for $C_j \in V$ $(j \in J)$ we have $\bigcup_{j\in J} C_j \in V$.

$V \subseteq L^X$ is called an **L**-*closure system* if

- V is closed under left $\rightarrow$-multiplication (or $\rightarrow$-shift), i.e. for every $a \in L$ and $C \in V$ we have $a \rightarrow C \in V$ (here, $a \rightarrow C$ is defined by $(a \rightarrow C)(i) = a \rightarrow C(i)$ for $i = 1, \ldots, n$);
- V is closed under intersection, i.e. for $C_j \in V$ $(j \in J)$ we have $\bigcap_{j\in J} C_j \in V$.

3 Weak L-bonds

This section introduces some new notions studied in this work. To begin with, we introduce the notion of weak **L**-bonds as a convenient generalization of bond.

Definition 1. *A weak* **L**-*bond between two* **L**-*contexts* $\mathbb{K}_1 = \langle X_1, Y_1, I_1\rangle$ *and* $\mathbb{K}_2 = \langle X_2, Y_2, I_2\rangle$ *w.r.t.* $\langle\cap, \cup\rangle$ *is an* **L**-*relation* $\beta \in L^{X_1\times Y_2}$ *s.t.*

$$\mathrm{Ext}(X_1^{\cap}, Y_2^{\cup}, \beta) \subseteq \mathrm{Ext}(X_1^{\cap}, Y_1^{\cup}, I_1) \quad \textit{and} \quad \mathrm{Int}(X_1^{\cap}, Y_2^{\cup}, \beta) \subseteq \mathrm{Int}(X_2^{\cap}, Y_2^{\cup}, I_2).$$

This notion can be put in relation with that of i-morphism.

Definition 2. *A mapping* $h : V \rightarrow W$ *from an* **L**-*interior system* $V \subseteq L^X$ *into an* **L**-*interior system* $W \subseteq L^Y$ *is called an* i-morphism *if it is a* $\otimes$- *and* $\bigvee$-*morphism, i.e. if*

- $h(a \otimes C) = a \otimes h(C)$ *for each* $a \in L$ *and* $C \in V$*;*
- $h(\bigvee_{k\in K} C_k) = \bigvee_{k\in K} h(C_k)$ *for every collection of* $C_k \in V$ $(k \in K)$*.*

An i-morphism $h : V \rightarrow W$ *is said to be an* extendable i-morphism *if* h *can be extended to an i-morphism of* L^X *to* L^Y*, i.e. if there exists an i-morphism* $h'\colon L^X \rightarrow L^Y$ *such that for every* $C \in V$ *we have* $h'(C) = h(C)$*;*

The following results will be used hereafter.

Lemma 1 ([5]).

1. *For* $V \subseteq L^X$*, if* $h : V \rightarrow L^Y$ *is an extendable i-morphism then there exists an* **L**-*relation* $A \in L^{X\times Y}$ *such that* $h(C) = C \circ A$ *for every* $C \in L^Y$*.*

2. *Let* $A \in L^{X \times Y}$*, the mapping* $h_A : L^X \to L^Y$ *defined by* $h_A(C) = C \circ A = C^{\cap_A}$ *is an extendable i-morphism.*
3. *Consider two contexts* $\langle X, Y, I\rangle$ *and* $\langle F, Y, B\rangle$*. Then, we have* $\mathrm{Int}(X^{\cap}, Y^{\cup}, I) \subseteq \mathrm{Int}(F^{\cap}, Y^{\cup}, B)$ *if and only if there exists* $A \in L^{X \times F}$ *such that* $I = A \circ B$*,*
4. *Consider two contexts* $\langle X, Y, I\rangle$ *and* $\langle X, F, A\rangle$*. Then, we have* $\mathrm{Ext}(X^{\cap}, Y^{\cup}, I) \subseteq \mathrm{Ext}(X^{\cap}, F^{\cup}, A)$ *if and only if there exists* $B \in L^{F \times Y}$ *such that* $I = A \circ B$*.*

Theorem 1. *The weak* **L***-bonds between* $\mathbb{K}_1 = \langle X_1, Y_1, I_1\rangle$ *and* $\mathbb{K}_2 = \langle X_2, Y_2, I_2\rangle$ *are in one-to-one correspondence with extendable i-morphisms* $\mathrm{Int}(X_1^{\cap}, Y_1^{\cup}, I_1)$ *to* $\mathrm{Int}(X_2^{\cap}, Y_2^{\cup}, I_2)$*.*

Proof. We show procedures to obtain the i-morphism from a weak **L**-bond and vice versa.

"⇒": Let β be a weak **L**-bond between $\mathbb{K}_1 = \langle X_1, Y_1, I_1\rangle$ and $\mathbb{K}_2 = \langle X_2, Y_2, I_2\rangle$. By Definition 1 we have $\mathrm{Int}(X_1^{\cap}, Y_2^{\cup}, \beta) \subseteq \mathrm{Int}(X_2^{\cap}, Y_2^{\cup}, I_2)$; thus by Lemma 1(3) there exists $S_{\mathrm{i}} \in L^{Y_1 \times Y_2}$ such that $\beta = I_1 \circ S_{\mathrm{i}}$. The induced opeator $\cap_{S_{\mathrm{i}}}$ is an extendable i-morphism $\mathrm{Int}(X_1^{\cap}, Y_1^{\cup}, I_1)$ to $\mathrm{Int}(X_2^{\cap}, Y_2^{\cup}, I_2)$ by Lemma 1(2).

"⇐": For an extendable i-morphism $f : \mathrm{Int}(X_1^{\cap}, Y_1^{\cup}, I_1) \to \mathrm{Int}(X_2^{\cap}, Y_2^{\cup}, I_2)$ there is an **L**-relation S_{i} s.t. $f(B) = B^{\cap_{S_{\mathrm{i}}}}$ for each $B \in \mathrm{Int}(X_1^{\cap}, Y_1^{\cup}, I_1)$ by Lemma 1(1). Then $\beta = I_1 \circ S_{\mathrm{i}}$ is a weak **L**-bond by Lemma 1(3) and Lemma 1(4).

One can check that these two procedures are mutually inverse. ◻

Now, consider **L**-bonds w.r.t. $\langle \wedge, \vee\rangle$ defined similarly as in Definition 1, i.e. an **L**-relation $\beta \in L^{X_1 \times Y_2}$ s.t.

$$\mathrm{Ext}(X_1^{\wedge}, Y_2^{\vee}, \beta) \subseteq \mathrm{Ext}(X_1^{\wedge}, Y_1^{\vee}, I_1) \quad \text{and} \quad \mathrm{Int}(X_1^{\wedge}, Y_2^{\vee}, \beta) \subseteq \mathrm{Int}(X_2^{\wedge}, Y_2^{\vee}, I_2).$$

Note that the weak **L**-bonds w.r.t. $\langle \cap, \cup\rangle$ are different from **L**-bonds w.r.t. $\langle \wedge, \vee\rangle$ as the following example shows.

Example 1. Consider L a finite chain containing $a < b$ with $\otimes$ defined as follows:

$$x \otimes y = \begin{cases} x \wedge y & \text{if } x = 1 \text{ or } y = 1, \\ 0 & \text{otherwise,} \end{cases}$$

for each $x, y \in L$. One can easily see that $x \otimes \bigvee_j y_j = \bigvee_j (x \otimes y_j)$ and thus an adjoint operation $\to$ exists such that $\langle L, \wedge, \vee, \otimes, \to, 0, 1\rangle$ is a complete residuated lattice. Namely, $\to$ is given as follows:

$$x \to y = \begin{cases} 1 & \text{if } x \leqslant y, \\ y & \text{if } x = 1, \\ b & \text{otherwise,} \end{cases}$$

for each $x, y \in L$. Consider $I_1 = \big(a\big)$ and $I_2 = \big(b\big)$. One can check that, we have $\mathrm{Ext}(\{x\}^{\cap}, \{y\}^{\cup}, I_1) = \mathrm{Ext}(\{x\}^{\cap}, \{y\}^{\cup}, I_2) = \{\{{}^{b}\!/x\}, x\}$ and, trivially, $\mathrm{Int}(\{x\}^{\cap}, \{y\}^{\cup}, I_2) = \mathrm{Int}(\{x\}^{\cap}, \{y\}^{\cup}, I_2)$. Thus I_2 is a weak **L**-bond between I_1 and I_2 w.r.t. $\langle \cap, \cup\rangle$.

On the other hand, I_2 is not a weak **L**-bond between I_1 and I_2 w.r.t. $\langle \wedge, \vee\rangle$ since $\mathrm{Ext}(\{x\}^{\wedge}, \{y\}^{\vee}, I_1) = \{\varnothing, \{{}^{a}\!/x\}\} \nsupseteq \{\varnothing, \{{}^{b}\!/x\}\} = \mathrm{Ext}(\{x\}^{\wedge}, \{y\}^{\vee}, I_2)$.

Theorem 2. *The system of all weak* **L***-bonds is an* **L***-interior system.*

Proof. From properties of i-morphism. □

4 Strong L-bonds

We provide a stronger version of the previously studied weak **L**-bond, naturally named strong **L**-bond.

Definition 3. *A strong* **L***-bond between two* **L***-contexts* $\mathbb{K}_1 = \langle X_1, Y_1, I_1 \rangle$ *and* $\mathbb{K}_2 = \langle X_2, Y_2, I_2 \rangle$ *is an* **L***-relation* $\beta \in L^{X_1 \times Y_2}$ *s.t.* β *is a weak* **L***-bond w.r.t. both* $\langle \cap, \cup \rangle$ *and* $\langle \wedge, \vee \rangle$*.*

The following lemma introduces equivalent definitions of strong **L**-bonds.

Lemma 2. *The following propositions are equivalent:*

(a) β *is a strong* **L***-bond between* $\mathbb{K}_1 = \langle X_1, Y_1, I_1 \rangle$ *and* $\mathbb{K}_2 = \langle X_2, Y_2, I_2 \rangle$*.*
(b) β *satisfies both* $\mathrm{Ext}(X_1^{\wedge}, Y_2^{\vee}, \beta) \subseteq \mathrm{Ext}(X_1^{\wedge}, Y_1^{\vee}, I_1)$ *and* $\mathrm{Int}(X_1^{\cap}, Y_2^{\cup}, \beta) \subseteq \mathrm{Int}(X_2^{\cap}, Y_2^{\cup}, I_2)$*.*
(c) $\beta = S_{\mathrm{e}} \circ I_2 = I_1 \circ S_{\mathrm{i}}$ *for some* $S_{\mathrm{e}} \in L^{X_1 \times X_2}$ *and* $S_{\mathrm{i}} \in L^{Y_1 \times Y_2}$*.*

Proof. (a) ⇔ (b): By use of the Lemma 1(3) and (4).
(a) ⇔ (c): By definitions. □

In this case, this stronger notion can be related with the c-morphisms, introduced below:

Definition 4. *A mapping* $h : V \to W$ *from a* **L***-closure system* $V \subseteq L^X$ *into an* **L***-closure system* $W \subseteq L^Y$ *is called a* c-morphism *if it is a* $\rightarrow$*- and* $\bigwedge$*-morphism, i.e. if*

- $h(a \rightarrow C) = a \rightarrow h(C)$ *for each* $a \in L$ *and* $C \in V$*;*
- $h(\bigwedge_{k \in K} C_k) = \bigwedge_{k \in K} h(C_k)$ *for every collection of* $C_k \in V$ $(k \in K)$*;*
- *if* C *is an a-complement then* $h(C)$ *is an a-complement.*

For formally establishing the relationship, the two following results are recalled:

Lemma 3 ([4]).

1. *If* $h : V \to L^Y$ *is an extendable c-morphism then there exists an* **L***-relation* $A \in L^{X \times Y}$ *such that* $h(C) = C \triangleright A$ *for every* $C \in L^Y$*.*
2. *Let* $A \in L^{X \times Y}$*, the mapping* $h_A : L^X \to L^Y$ *defined by* $h_A(C) = C \triangleright A \quad (= C^{\wedge_A})$ *is an extendable c-morphism.*

Theorem 3. *The strong* **L***-bonds between* $\mathbb{K}_1 = \langle X_1, Y_1, I_1 \rangle$ *and* $\mathbb{K}_2 = \langle X_2, Y_2, I_2 \rangle$ *are in one-to-one correspondence with extendable c-morphisms* $\mathrm{Ext}(X_2^{\cap}, Y_2^{\cup}, I_2)$ *to* $\mathrm{Ext}(X_1^{\cap}, Y_1^{\cup}, I_1)$*.*

Proof. We show procedures to obtain the c-morphism from a strong **L**-bond and vice versa.

"⇒": Let β be a strong **L**-bond between $\mathbb{K}_1 = \langle X_1, Y_1, I_1\rangle$ and $\mathbb{K}_2 = \langle X_2, Y_2, I_2\rangle$. By Lemma 2 there is $S_\mathrm{e} \in L^{X_1 \times X_2}$ such that $\beta = S_\mathrm{i} \circ I_2$. The induced operator $\cup_{S_\mathrm{i}}$ is an extendable c-morphism $\mathrm{Ext}(X_2^\cap, Y_2^\cup, I_2)$ to $\mathrm{Ext}(X_1^\cap, Y_1^\cup, I_1)$ by Lemma 3(2).

"⇐": For extendable c-morphism $f : \mathrm{Ext}(X_2^\uparrow, Y_2^\downarrow, I_2) \to \mathrm{Ext}(X_1^\cap, Y_1^\cup, I_1)$ there is an **L**-relation S_e s.t. $f(B) = B^{\cup_{S_\mathrm{i}}}$ for each $A \in \mathrm{Ext}(X_2^\cap, Y_2^\cup, I_2)$ by Lemma 3(1). Then $\beta = S_\mathrm{e} \circ I_2$ is a strong **L**-bond by Lemma 2.

One can check that these two procedures are mutually inverse. □

Theorem 4. *The system of all strong* **L***-bonds is an* **L***-interior system.*

Proof. Using Lemma 2 (b), it is an intersection of the **L**-interior systems from Theorem 2. □

Definition 5. *Let* $\mathbb{K}_1 = \langle X_1, Y_1, I_1\rangle, \mathbb{K}_2 = \langle X_2, Y_2, I_2\rangle$ *be* **L***-contexts. The direct product of* $\mathbb{K}_1$ *and* $\mathbb{K}_2$ *is defined as the* **L***-context* $\mathbb{K}_1 \boxplus \mathbb{K}_2 = \langle X_2 \times Y_1, X_1 \times Y_2, \Delta\rangle$ *with* $\Delta(\langle x_2, y_1\rangle, \langle x_1, y_2\rangle) = I_1(x_1, y_1) \otimes I_2(x_2, y_2)$.

Theorem 5. *The intents of* $\mathbb{K}_1 \boxplus \mathbb{K}_2$ *are* **L***-bonds between* $\mathbb{K}_1$ *and* $\mathbb{K}_2$.

Proof. We have

$$\begin{aligned}
\phi^\cap(x_1, y_2) &= \bigvee \phi(x_2, y_1) \otimes \Delta(\langle x_2, y_1\rangle, \langle x_1, y_2\rangle) \\
&= \bigvee_{\langle x_2, y_1\rangle} \phi(x_2, y_1) \otimes I_1(x_1, y_1) \otimes I_2(x_2, y_2) \\
&= \bigvee_{y_1 \in Y_1} \bigvee_{x_2 \in X_2} \phi(x_2, y_1) \otimes I_1(x_1, y_1) \otimes I_2(x_2, y_2) \\
&= \bigvee_{y_1 \in Y_1} I_1(x_1, y_1) \otimes \bigvee_{x_2 \in X_2} \phi(x_2, y_1) \otimes I_2(x_2, y_2) \\
&= \bigvee_{y_1 \in Y_1} I_1(x_1, y_1) \otimes (\phi^T \circ I_2)(y_1, y_2) \\
&= (I_1 \circ \phi^T \circ I_2)(x_1, y_2).
\end{aligned}$$

Now, notice that $(I_1 \circ \phi^T) \circ I_2 = I_1 \circ (\phi^T \circ I_2) = \beta$ is a strong **L**-bond by Lemma 2. □

Remark 1. It is worth mentioning that not every strong **L**-bond is included in $\mathrm{Int}((X_1 \times Y_2)^\cap, (X_2 \times Y_1)^\cup, \Delta)$ since there are isotone **L**-bonds which are not of the form of $I_1 \circ \phi^T \circ I_2$. For instance, using the same structure of truth degrees and I_1 as in Example 1, obviously I_1 is **L**-bond on $\mathbb{K}_1$ (i.e. between $\mathbb{K}_1$ and $\mathbb{K}_1$), but $\mathrm{Int}((X_1 \times Y_2)^\cap, (X_2 \times Y_1)^\cup, \Delta)$ contains only $\emptyset$.

The end of proof of the Theorem 5 also explains which **L**-bonds are intents of $\mathbb{K}_1 \boxplus \mathbb{K}_2$:

Corollary 1. *The intents of* $\mathbb{K}_1 \boxplus \mathbb{K}_2$ *are exactly those* **L***-bonds between* $\mathbb{K}_1$ *and* $\mathbb{K}_2$ *which can be decomposed as* $I_1 \circ \phi^T \circ I_2$ *for some* $\phi \in L^{X_2 \times Y_1}$.

Remark 2 (Relationship to the antitone case in [12]).

Assuming the double negation law, we have the equality $\operatorname{Ext}(X^{\uparrow}, Y^{\downarrow}, I) = \operatorname{Ext}(X^{\cap}, Y^{\cup}, \neg I)$. Thus, for a strong **L**-bond $\beta \in L^{X_1 \times Y_2} = S_{\mathrm{e}} \circ I_2 = I_1 \circ S_{\mathrm{i}}$ between $\mathbb{K}_1$ and $\mathbb{K}_2$ we have $\neg\beta = S_{\mathrm{e}} \triangleleft \neg I_2 = \neg I_1 \triangleright S_{\mathrm{i}}$ being an antitone **L**-bond between $\neg\mathbb{K}_1$ and $\neg\mathbb{K}_2$.

Remark 3. Some papers [9, 12] have considered direct products in the crisp and the fuzzy settings, respectively, for the antitone case. In [12] conditions are specified under which antitone **L**-bonds are present in the concept lattice of the direct product. Corollary 1 and Remark 2 provide a simplification of these conditions. The concept lattice of a direct product $\mathbb{K}_1 \boxtimes \mathbb{K}_2$ defined as in [12] i.e. $\mathbb{K}_1 \boxtimes \mathbb{K}_2 = \langle X_2 \times Y_1, X_1 \times Y_2, \Delta \rangle$ with

$$\Delta(\langle x_2, y_1 \rangle, \langle x_1, y_2 \rangle) = \neg I_1(x_1, y_1) \to I_2(x_2, y_2) \quad (= \neg I_2(x_2, y_2) \to I_1(x_1, y_1))$$

induces concept-forming operator $\phi^{\uparrow_\Delta}$ for which we have

$$\begin{aligned}
\phi^{\uparrow_\Delta}(x_1, y_2) &= \bigwedge_{\langle x_1, y_2 \rangle \in X_1 \times Y_2} \phi(x_2, y_1) \to [\neg I_1(x_1, y_1) \to I_2(x_2, y_2)] \\
&= \bigwedge_{\langle x_2, y_1 \rangle \in X_1 \times Y_2} \neg I_1(x_1, y_1) \to (\phi(x_2, y_1) \to I_2(x_2, y_2)) \\
&= \bigwedge_{x_2 \in X_2} \bigwedge_{y_1 \in Y_1} \neg I_1(x_1, y_1) \to (\phi(x_2, y_1) \to I_2(x_2, y_2)) \\
&= \bigwedge_{y_1 \in Y_1} \neg I_1(x_1, y_1) \to \bigwedge_{x_2 \in X_2} (\phi(x_2, y_1) \to I_2(x_2, y_2)) \\
&= \bigwedge_{y_1 \in Y_1} \neg I_1(x_1, y_1) \to (\phi^T \triangleleft I_2)(y_1, y_2) \\
&= \bigwedge_{y_1 \in Y_1} \neg(\phi^T \triangleleft I_2)(y_1, y_2) \to I_1(x_1, y_1) \\
&= [I_1 \triangleright \neg(\phi^T \triangleleft I_2)](x_1, y_2) \\
&= [\neg I_1 \triangleleft (\phi^T \triangleleft I_2)](x_1, y_2) \\
&= [\neg I_1 \triangleleft (\phi^T \triangleleft I_2)](x_1, y_2) \\
&= [(\neg I_1 \circ \phi^T) \triangleleft I_2)](x_1, y_2) \\
&= [\neg(\neg I_1 \circ \phi^T \circ \neg I_2)](x_1, y_2)
\end{aligned}$$

Whence an antitone **L**-bond is an intent of the concept lattice of $\mathbb{K}_1 \boxtimes \mathbb{K}_2$ iff it is possible to write it as $\neg(\neg I_1 \circ \phi^T \circ \neg I_2)$ i.e. if its complement is an intent of $\neg\mathbb{K}_1 \boxplus \neg\mathbb{K}_2$.

5 Conclusions and future work

We studied **L**-bonds with respect to isotone concept-forming operators, computation of **L**-bonds using dirrect products, and the relationship of these results to the previous results on antitone **L**-bonds.

The present results can be easily generalized to a setting in which extents, intents and the context relation use different structures of truth-degrees. We will bring this generalization in an extended version of the paper.

Our future research in this area includes the the study of yet another type of (extendable) i-morphisms and c-morphisms. In [11], another type of morphism is described: the *a-morphism*. In contrast to the morphisms used in this paper, the a-morphisms are antitone, and their study could shed more light on antitone fuzzy bonds.

References

1. C. Alcalde, A. Burusco, and R. Fuentes-González. Interval-valued linguistic variables: an application to the L-fuzzy contexts with absent values. *Int. J. General Systems*, 39(3):255–270, 2010.
2. 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.
3. R. Bělohlávek. *Fuzzy Relational Systems: Foundations and Principles.* Kluwer Academic Publishers, 2002.
4. R. Belohlavek, J. Konecny. Closure spaces of isotone Galois connections and their morphisms. In Dianhui Wang and Mark Reynolds, editors, *Australasian Conference on Artificial Intelligence*, volume 7106 of *Lecture Notes in Computer Science*, pages 182–191. Springer, 2011.
5. R. Belohlavek, J. Konecny. Row and column spaces of matrices over residuated lattices. *Fundam. Inf.*, 115(4):279–295, December 2012.
6. P. Butka, J. Pócsová, and J. Pócs. On generation of one-sided concept lattices from restricted context. In *IEEE 10th Jubilee Intl Symp on Intelligent Systems and Informatics (SISY)*, pages 111–115, 2012.
7. J. T. Denniston, A. Melton, and S. E. Rodabaugh. Formal concept analysis and lattice-valued Chu systems. *Fuzzy Sets and Systems*, 216:52–90, 2013.
8. D. Dubois and H. Prade. Possibility theory and formal concept analysis: Characterizing independent sub-contexts. *Fuzzy Sets and Systems*, 196:4–16, 2012.
9. B. Ganter. Relational galois connections. In Sergei O. Kuznetsov and Stefan Schmidt, editors, *Formal Concept Analysis*, volume 4390 of *Lecture Notes in Computer Science*, pages 1–17. Springer Berlin Heidelberg, 2007.
10. J. Goguen. What is a concept? *Lecture Notes in Computer Science*, 3596:52–77, 2005.
11. J. Konecny. Closure and Interior Structures in Relational Data Analysis and Their Morphisms. PhD Thesis, UP Olomouc, 2012.
12. 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.
13. M. Krötzsch, P. Hitzler, and G.-Q. Zhang. Morphisms in context. *Lecture Notes in Computer Science*, 3596:223–237, 2005.

14. H. Lai and D. Zhang. Concept lattices of fuzzy contexts: Formal concept analysis vs. rough set theory. *Int. J. Approx. Reasoning*, 50(5):695–707, 2009.
15. Y. Lei and M. Luo. Rough concept lattices and domains. *Annals of Pure and Applied Logic*, 159(3):333–340, 2009.
16. J. Medina. Multi-adjoint property-oriented and object-oriented concept lattices. *Information Sciences*, 190:95–106, 2012.
17. J. Medina and M. Ojeda-Aciego. Multi-adjoint t-concept lattices. *Information Sciences*, 180(5):712–725, 2010.
18. J. Medina and M. Ojeda-Aciego. On multi-adjoint concept lattices based on heterogeneous conjunctors. *Fuzzy Sets and Systems*, 208:95–110, 2012.
19. 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.
20. S. Solovyov. Lattice-valued topological systems as a framework for lattice-valued formal concept analysis. *Journal of Mathematics*, 2013. To appear.
21. Q. Wu and Z. Liu. Real formal concept analysis based on grey-rough set theory. *Knowledge-Based Systems*, 22(1):38–45, 2009.
22. D. Zhang. Galois connections between categories of L-topological spaces. *Fuzzy Sets and Systems*, 152(2):385–394, 2005.

A Conceptual-KDD approach and its application to cultural heritage

Renzo Stanley[1], Hernán Astudillo[1], Víctor Codocedo[2], and Amedeo Napoli[2]

[1]Universidad Técnica Federico Santa María, Av. España 1680. Valparaíso, Chile,
`{rstanley,hernan}@inf.utfsm.cl`
[2]LORIA, BP 70239, F-54506 Vandoeuvre-lès-Nancy, France.
`{victor.codocedo,amedeo.napoli}@loria.fr`

Abstract. Several governmental and non-governmental organizations (NGOs), motivated by the UNESCO have undertaken the task of documenting the intangible cultural heritage of their communities. However, this has proven to be a difficult task. In this work we present a conceptual knowledge discovery in databases (CKDD) approach to aid a particular organization in this task (which has already started). Because of the dynamism of the cultural heritage domain, the design of the database used to store the documentation data has become obsolete. We propose to redesign the database (actually, its schema) to unveil independent modules of information collaboratively created by different domain experts. Finally, we present a straightforward method to convert the redesigned data schema into an ontological model which can be used for integration and publication purposes.

Keywords: Formal concept analysis, Knowledge discovery in databases, Conceptual knowledge discovery, Intangible Cultural Heritage, Documentation

1 Introduction

The Chilean National Council of Culture and Arts[1] (CNCA) has undergone the mission of documenting the intangible cultural heritage (ICH) of different small zones of the country in the context of a world-wide UNESCO[2] crusade to incentivize governments and NGOs to properly maintain our cultural knowledge. The ICH, as drafted in the *Convention for the Safeguarding of the Intangible Cultural Heritage*[3] refers to *"traditions or living expressions inherited from our ancestors and passed on to our descendants, such as oral traditions, performing arts, social practices, rituals, festive events, knowledge and practices concerning nature and the universe or the knowledge and skills to produce traditional crafts"*.

---

[1] `http://www.consejodelacultura.cl`

[2] United nations educational, scientific and cultural organization.

[3] `http://en.wikipedia.org/wiki/Convention_for_the_Safeguarding_of_Intangible_Cultural_Heritage`

To survey the ICH, the CNCA maintains several "documenters" who meet artists, artisans and other actors of the folkloric stage. Each documenter fills a "file" containing several semi-structured and text-free fields which are later registered in a relational database (DB). This database is later consulted by curators (usually domain experts) who fix problems in data definitions or storage. The CNCA has requested help in two specific aspects. Given the difficulty of documenting very different domains in ICH, the design of a data schema to support its documentation is a hard task which leads to inconsistencies in its model. Currently, the data schema used by the CNCA was designed by a computer engineer with some knowledge on the domain. However, as the ICH documentation process expanded to other cultural domains ("music", "folk festivals", "crafts", etc.) the data schema became too general and some modifications were requested to enable a higher level of details. The CNCA requested to analyse the data schema in order to find "modular" partitions on the model corresponding to different domains in ICH. Thus, each module can be maintained by a related domain-expert avoiding the risk of information conflict.

The second request derives from the fact that, since the ICH documentation is a multi-organizational endeavour motivated by UNESCO, it is expected that data collected by CNCA will be integrated with other databases later on. Because of this, CNCA has asked aid to obtain an ontological schema from its current data schema which can be used for linked data publication.

In this work we propose a novel knowledge discovery in databases process guided by formal concept analysis (also, referred to as *conceptual knowledge discovery in databases CKDD* [1]) approach to deal with both requirements. We analyse the current data schema definition to construct a concept lattice through FCA in which we apply different techniques to find the modules representing *ICH domains*. The contributions of this work are firstly, to present a real-world experience using FCA and concept lattices to achieve a task as important as heritage documentation and secondly, the description of a process to elicit an ontological schema from a relational DB schema which can be of benefit in different applications.

The remainder of this article is organised as follows: Section 2 presents the CKDD process while Sections 3, 4 and 5 detail each of its steps. Section 6 describes the case study of applying the CKDD process over the CNCA database for ICH documentation. Finally, Section 7 presents a discussion on related work and concludes the paper.

2 Formal concept analysis and knowledge discovery in databases.

We propose an iterative and human-centred approach to overcome both requirements of the CNCA based on Conceptual Knowledge Discovery in Databases (CKDD) [1]. CKDD is a tool to support humans in the discovery and extraction of knowledge from large collections of data where the conceptual representation of knowledge is a key aspect [12].

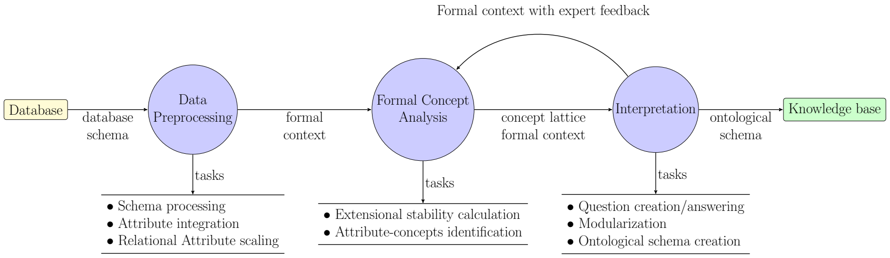

Formal context with expert feedback
Database
database schema
Data Preprocessing
formal context
Formal Concept Analysis
concept lattice formal context
Interpretation
ontological schema
Knowledge base
tasks
• Schema processing
• Attribute integration
• Relational Attribute scaling
tasks
• Extensional stability calculation
• Attribute-concepts identification
tasks
• Question creation/answering
• Modularization
• Ontological schema creation

Fig. 1: FCA-based KDD process for CNCA

Figure 1 presents a 3-step CKDD process designed to take a database schema and convert it into a knowledge base. The process contains a loop between the interpretation and the FCA step in which domain expert knowledge is included. Notice that to avoid conflicts with the CNCA data, we only extract information from the database and never modify it. In the following sections we provide a detailed description of each step and sub-tasks developed.

3 First Step of the CKKD process: Data preprocessing

The first step starts by extracting the database schema and ends when it is converted to a formal context. Currently, most of the relational database management systems are based on the relational model constituted by tables and their relations (there are actually more elements in a relational model which are not of first interest here). We provide an adapted definition of the relational schema model as described in [2].

3.1 Relational data schema model

A relational schema $S = \{R_1, R_2, ..., R_{|S|}\}$ is defined as a set of tables or "relation schemas" $R_i(A_1, A_2, ..., A_{n_i})$ consisting of a table name R_i and a list of n_i fields A_j which define value assignments of the domain $dom(A_j)$ to an *entry* in the table. The notation $R_i.A_j$ stands for the field A_j in table R_i. An entry in a table is defined as an ordered *n-tuple* of values denoted by $t[R_i] =< v_1, v_2, ..., v_{n_i} >$ where each value is denoted as $t[A_j] = v_j \in dom(A_j) \cup \{NULL\}$. $NULL$ indicates that the value is unknown for the field in the entry. The *relation state* $r(R_i) = \{t_1, t_2, ..., t_{r_i}\}$ of table R_i denotes its total set of entries. In a table R_i, the set of fields $SK \subset R_i$ denotes a *superkey* which identifies a tuple as unique. A *primary key* PK is defined as a *superkey* where $|SK| = 1$ and $|.|$ denotes set cardinality, i.e. PK is a single field and we say that $R_i.PK$ is the *primary key* of table R_i the value of which unequivocally identifies an entry in table R_i. Finally, a field $R_1.FK$ is called a *foreign key* iff $R_1.FK = R_2.PK$ which indicates a *relation* between R_1 and R_2. In the particular case of the CNCA database for ICH documentation, a notion of inheritance of tables is supported. We define this as

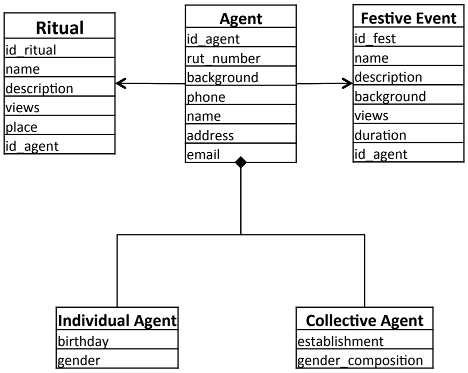


Fig. 2: Data schema example from the ICH domain

follows: table R_1 inherits from table R_2 iff $R_2 \subset R_1$ and $R_1.PK = R_2.PK$, i.e. table R_1 contains all the fields in table R_2 and they share the same primary key.

Figure 2 illustrates a data schema example $S = \{$*Festive Event*, *Ritual*, *Agent*, *Individual Agent*, *Collective Agent*$\}$ where arrows represent relation between two tables and a line with a rhombus at the end represents inheritance. Table *Ritual* can be represented as *Ritual(id_ritual, name, description, views, place, id_agent)* where $dom(name) = string$, *Ritual.id_ritual* is the primary key of table *Ritual* and *Ritual.id_agent* is a *foreign key* relating *Ritual* with *Agent*. The example also shows that *Individual Agent* and *Collective Agent* inherit from table *Agent*, meaning that all the attributes from table *Agent* are also present in tables *Individual Agent* and *Collective Agent*.

3.2 Formal context definition

Considering data schema $S = \{R_1, R_2, ..., R_{|S|}\}$ and the tables of the form $R_i(A_1, A_2, ..., A_{n_i})$ composed by the fields A_j, we define the formal context $\mathcal{K} = (S, A, I)$ where

$$A = \bigcup_{R_i \in S} R_i.A_j$$

$$I = \{(R_i, A_j) \,/\, \forall R_i \in S, \forall A_j \in R_i\}$$

Formal context $\mathcal{K}$ is composed by tables in the set S (all the tables or a sub-set of them), the set of fields A (composed by the set of all fields from all tables considered in S) and the relation set I (the relations between tables in S and their fields in A). Notice that we define a formal context by making the correspondences object-table and attribute-field. To avoid confusions, in this work we differentiate between these correspondences by using different font faces for `objects or attributes` and for *tables or fields*. Along with the formal context, we also define the following rules of **attribute integration** based on the characteristics of the fields:

- The special attribute `id` represents all *primary keys*:
  $id \in A$, $(R_i, id) \in I \; \forall R_i \in S$.
- Fields with the same name are integrated into a single attribute (e.g. attribute `name` represents *Agent.name* and *Ritual.name*)

3.3 Relational attribute scaling

In the context of FCA, *foreign keys* correspond to relational attributes. For example, in Figure 2 we do not say "table *Ritual* contains a *foreign key*" (as in the case of *Ritual* contains a *primary key*), but rather "*Ritual* is related to *Agent* by a *foreign key*". Such kind of attributes cannot be included in a binary formal context. To deal with relational attributes, we scale them and treat them as normal attributes in the lines of [11]. We do so by prepending the prefix `related_to:` to the name of the table where the *foreign key* directs to (e.g. in the formal context in Table 1, we say that the object `Ritual` contains the attribute `related_to:Agent`). This is formalized as

$$R_j.FK = R_i.PK; \; R_i, R_j \in S \Rightarrow \texttt{related\_to:}R_i \in A \wedge \\ (R_j, \texttt{related\_to:}R_i) \in I$$

In the case the table R_i pointed to by a foreign key do not exist in S, then we simply do not take into consideration that table nor we create the scaled relation `related_to:`R_i.

	id	rut_number	email	phone	name	address	background	description	views	place	related_to:Agent	birthday	gender	establishment	gender_composition	duration
Agent	×	×	×	×	×	×	×									
IndividualAgent	×	×	×	×	×	×	×					×	×			
CollectiveAgent	×	×	×	×	×	×	×							×	×	
FestiveEvent	×				×		×	×	×		×					×
Ritual	×				×			×	×	×	×					

Table 1: Formal context created from the data schema example

4 Second Step of the CKDD process: Formal Concept Analysis

This step receives a formal context and ends when a concept lattice is constructed. Figure 3(a) shows the concept lattice calculated for the formal context in Table 1. Along with the concept lattice, the original formal context is sent to step 3 to allow modifications in it for further iterations of the process. We develop two tasks related to the identification of elements which would help in the redesign of the data schema.

4.1 Attribute concept identification

The *attribute concept* of attribute A_j is defined by $\mu A_j = (A'_j, A''_j)$, where $()'$ is the derivation operator in FCA [3]. The identification of attribute concepts is rather simple and we do it by navigating from the top to the bottom of the lattice. Attribute concepts are the most general concepts in the lattice containing a given attribute. In our case, the attribute concept of a given field contains the largest set of tables related to it. This is important since, as we will describe in the next step of interpretation, we process each field separately in order to create the ontological schema. Attribute concepts make this simpler as for each field we only need one concept and not the whole lattice.

4.2 Extensional stability calculation:

Since we are looking to enhance the design of a data schema in the database, within the lattice we would like to have only those concepts which group together in their extents tables of the same domain. This would allow us to address the modularization of information per domain issue which is one of the request of the CNCA, assuming that tables in the same domain are likely to share similar fields. In order to do so, we use the notion of *extensional stability* as firstly described in [6] and later in [10] as a way to measure "the probability of a concept to preserve its extent after leaving out an arbitrary number of attributes".

5 Third Step of the CKDD process: Interpretation

The final step receives a formal context and its associated concept lattice where each attribute concept is identified and each formal concept contains an *extensional stability* value. Since CKDD is an iterative process, this step has two possible outputs. If the expert decides to make another iteration, the process goes back to step 2 sending a modified version of the formal context received including feedback of the expert. If the expert decides to end the process, this will create an "ontological schema" which will be stored in a knowledge base and the process ends. For the iteration, there are two tasks to perform: *Question creation/answering* and *Modularization*. The task which transforms the concept lattice into an *ontological schema* ends the process. A further step of *annotation* which converts the entries in tables into linked data using the ontological schema can be considered, but for the sake of space we have left it out of this paper.

5.1 Question creation and answering

We use *extensional stability* as an indicator of how related are tables within a given concept extent. A lower stability indicates that those tables are grouped more as an accident (for example, because of the misuse of a single (or several) field name(s)) rather than because they belong to the same domain. We look for unstable attribute concepts because they are the most general concepts containing a given attribute, so they are those that relate more tables together. Moreover, we can create questions for the expert to answer in the hope they provide information to "break" the attribute concept and separate domains.

Consider the example in Figure 3(a) in which two domains are illustrated. The first relates events while the second relates people and communities. The most unstable attribute concepts (extensional stability of 0.5) correspond to those labelled with the attributes `background` and `place`. In the case of `place`, it only contains one table meaning that "breaking" this concept does not help in separating domains[4]. The attribute concept of `background` contains four objects, namely `FestiveEvent`, `Agent`, `IndividualAgent` and `CollectiveAgent` for which `FestiveEvent` belongs to a different domain than the others.

Regarding the questions posed to the expert, we have:

1. Would you like to assign `attribute` to all the objects?
2. Would you like to eliminate the `attribute` from a single/a set of objects/s?
3. Would you like to split `attribute` into different attributes for different objects?

If the expert selects option 1, the attribute will be shared by every object and hence, it will be placed as part of the intent of the top of the lattice (like `id` and `name` in Figure 3(a)). In the case of option 2, the attribute is erased from the formal context and hence, its attribute concept is removed. With option 3, we can recommend the partition of the attribute by looking at the sub-concepts of the attribute concept. Then, a new attribute is created for each partition made while the original attribute disappears. Consider A_i to be the attribute for which the expert has to answer a question. For each option we can define a new formal context as follows:

1. Option 1: $\mathcal{K}_A = (S, A, I \cup \{(R, A_i); \forall R \in S\})$
2. Option 2: $\mathcal{K}_A = (S, A\backslash\{A_i\}, I\backslash\{(R, A_i;)\forall R \in S\})$
3. Option 3: Let A^j be the splits of attribute A_i assigned to objects R^j, then $\mathcal{K}_A = (S, (A\backslash\{A_i\}) \cup A^j, I\backslash\{(R, A_i); \forall R \in S\} \cup \{(R^j, A^j)\})$

In the example, the expert selected option 3 splitting `background` into an attribute for `FestiveEvent` (`practice_background`) and another for the `Agents` (`records`). This example is actually an extract of a real case study where the

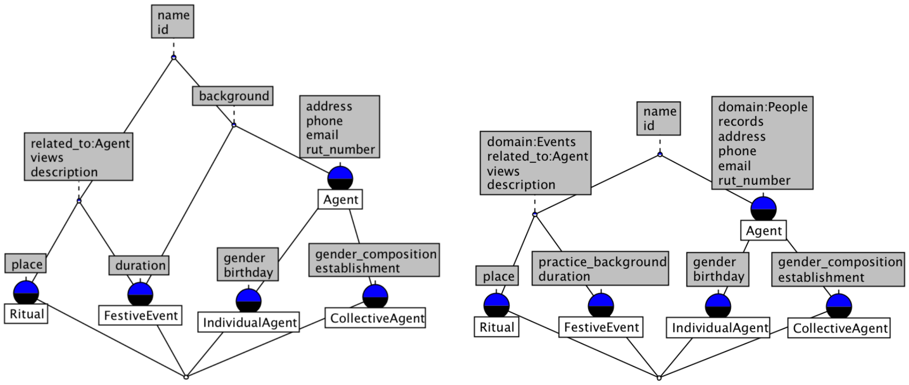
---

[4] A low stability in a singled-object attribute concept may indicate that the table should be split in two or more different tables. We do not address this issue in this work

expert realized that the term “background” was used with multiple meanings and that it should be separated into an attribute registering the history of past events and another for the official records of people. Figure 3(b) presents the concept lattice created from the formal context yielded by this decision.

5.2 Modularization

The expert may choose to perform this task disregarding answering questions. We start from the already processed concept lattice L and find its sublattices such as $L_i \cap L_j = \{\top, \bot\}$ where L_i, L_j are sublattices of L and $\top, \bot$ represent the top and the bottom of L respectively. We achieve this by obtaining the connected graphs from the lattice once $\top$ and $\bot$ are removed. Each sublattice is a candidate to represent a domain which must be labelled by the expert. The label is included into the formal context as a special attribute of the form `domain:Label` with relations to all the objects in the sublattice. The concept lattice is later recalculated. Figure 3(b) depicts the final form of the lattice for the running example. Finally, the expert may also be interested in merging more than one sublattice into a single domain. In that case, the special attribute is added to all objects in the set of sublattices selected by the expert.



(a) Concept lattice created from the data schema example

(b) Concept lattice after the expert’s decision (attribute `background` split into `practice_background` and `records`) including domain labels

Fig. 3: Concept lattices for CNCA (before and after the expert’s decision)

5.3 Ontological schema creation

The final task of the process derives an ontological model from the concept lattice that can be used for data integration and linked data publication. This is done

Concept	Element	Triples created
$\top = (S, \emptyset)$	$R_i \in S$	R_i `rdf:type rdfs:Class` *e.g. cnca:*Agent *rdf:type rdfs:Class*
$\bot = (\emptyset, A)$	$A_j \in A$	A_j `rdf:type rdfs:Property` A_j `rdfs:range rdfs:Literal` *e.g. cnca:establishment rdf:type rdfs:Property* *cnca:establishment rdfs:range rdfs:Literal*
$\bot = (\emptyset, A)$	`related_to:`$R_i \in A$	`related_to:`R_i `rdf:type rdfs:Property` `related_to:`R_i `rdf:range rdfs:`R_i *e.g. cnca:participant rdf:type rdfs:Property* *cnca:participant rdfs:range cnca:Agent*

$\bot = (\emptyset, A)$	`domain:Label` $\in A$	`cnca:Label rdf:type cnca:Domain`
		`cnca:Domain rdf:type rdfs:Class`
		`cnca:in_domain rdf:type rdfs:Property`
		e.g. cnca:People rdf:type cnca:Domain

$\mu A_j = (A'_j, A''_j)$	$R_i \in A'_j$	`cnca:`A_j `rdfs:domain cnca:`R_i *e.g. cnca:participant rdfs:domain cnca:Ritual*
$\mu A_j = (A'_j, A''_j)$	(A_j =`domain:Label` *and* $R_i \in A'_j$)	`cnca:`R_i `cnca:in_domain cnca:Label` *e.g. cnca:Agent cnca:in_domain cnca:People*

Table 2: Formal concept translation into RDF triples. Triples in the third column are created for the elements described in the second column within the formal concepts in the first column

by creating a set of RDF triples[5] for given elements in the formal concepts of the lattice. Table 2 shows this conversion where for example, in the first row it is shown that for the top concept, all the tables in its extent are modelled as RDFS classes[6] [7]. For simplicity purposes, Table 2 only presents a part of the total set of triples created considering the top $\top$, the bottom $\bot$ and attribute μA_j concepts.

6 Case study: CNCA Intangible Cultural Heritage Database

The database schema of the CNCA for ICH documentation consists of nearly 300 tables, however only 27 tables were selected by the experts on the basis of their representative and multi-disciplinary knowledge. Selected tables consider descriptions of agents, collective agents, festive events, culinary manifestations, geolocations and more. The formal context derived from the database contains 27 objects, 56 attributes, and 25 relational attributes. Figure 4 depicts the concept lattice built from this formal context.

---

[5] Resource Description Framework (RDF) is a standard specification for semantic web and linked data based on triples `http://www.w3.org/RDF/`.

[6] Resource Description Framework Schema (RDFS) is an extension of RDF which supports classes, properties and more complex definitions.

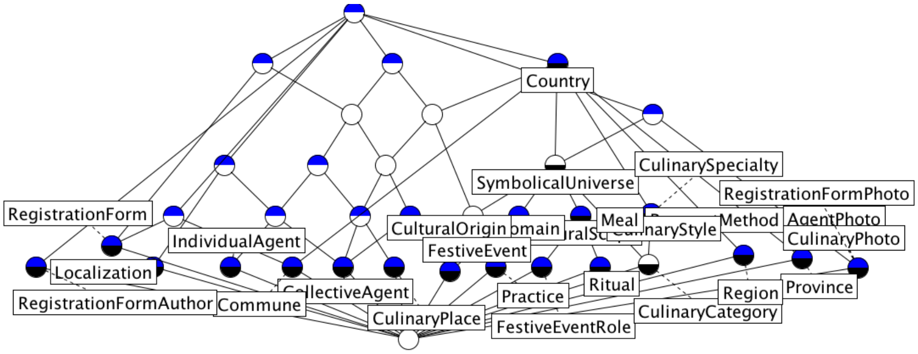


Fig. 4: Concept lattice from the ICH database schema

Iteration	Attribute	Action
1	`name`	Assign to all tables
2	`description`	Assign to all tables
3	`background`	Split the attribute
4	`background`	Split the attribute
5	`related_to:Commune`	Eliminate from some tables
6	`related_to:Localization`	Eliminate from some tables
7	-	Domain labelling
8	-	Domain labelling
9	-	Domain labelling

Table 3: Iterations made for the case study

Table 3 shows the actions taken by the domain expert during 9 iterations of the CKDD process. We distinguished between facts, questions and actions. Facts represent database assertions which are displayed for helping the expert in decision taking. For example, in iteration 1 the expert is presented with the facts: "71% of the objects contain the attribute `name`" and "The attribute `name` has a value in 100% of the entries in the objects where it appears". The first fact helps the expert to understand that `name` is actually very common among the objects and can easily be extended to the whole object set. The second fact combines information from the DB and the concept lattice to tell the expert that in all the entries t which contain the field *name* (represented by the attribute `name`), the value $t[name]$ is different than $NULL$. This indicates him that the attribute should not be removed from any object. Questions and actions were already detailed in Section 5.1.

In iterations 3 and 4 the expert split the attribute `background` ("antecedentes" in Spanish). This attribute appears in several objects (46% of object set) through all the domains, however with different semantics. Finally, the expert created the attributes `historical_background`, `records`, `practice_background` and `culinary_background`. In iteration 5 and 6, the expert decided to eliminate some attributes from the object set. For example, the relation `related_to:Commune` which represented a *foreign key* in the database, was not being used in 3 tables from which it was removed. Finally, in iterations 7, 8 and 9 the expert labelled the modules as subdomains of ICH `FestiveFeatures`, `Geo` and `AgentFeatures`. Figure 5 illustrates the final concept lattice presenting the different subdomains identified from left to right, namely; *Agent descriptors subdomain*, *Festive Event descriptors subdomain*, *Culinary descriptors subdomain*, *Geographical subdomain*, *Content creators subdomain*, *Photo subdomain*, *ICH subdomain*. The last subdomain includes all the manifestations about ICH in this database schema.

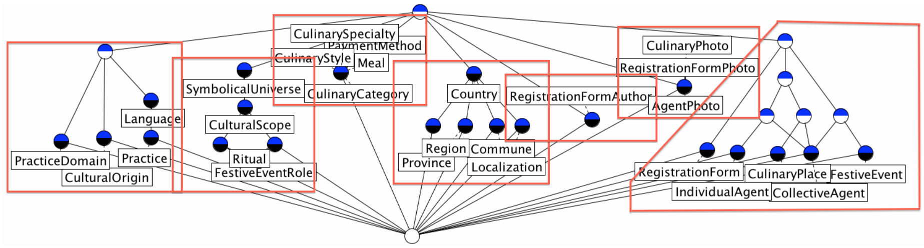


Fig. 5: Concept lattice after 9 iterations

7 Related Work, Discussion and Conclusion

Modelling semantic relations in DBs is proposed in [8] where the author presents a framework to formalize semantics in a lexical database. A lexical database maintains information about concepts and their semantic position w.r.t. each other (*hypernymy* or *meronymy*). In general, relational DBs do not share the same characteristics nor structure as lexical databases, allowing to store more heterogeneous data. In [9] the same author proposes an algebra for relational DBs which can be interpreted also in terms of a formal context in FCA. A similar idea was proposed by [5] where the author uses the algebra to translate a relational database into a family of formal contexts to benefit from both, the simplicity of the relational model description and the power of FCA in analysis. By contrast, our approach is more straightforward since we are not interested in data operations using the concept lattice, but rather in a direct translation of the data schema. Thus, we do not work with the entries in the tables of the database. In our approach we use the concept lattice as a support for guiding the redesign process in which a domain expert is the main provider of knowledge.

To conclude, in this article we have presented an application of a conceptual knowledge discovery in databases process designed to redesign and convert a database schema into an ontological model. The process is heavily *human centred* as it considers a domain expert as the main source of knowledge to guide the process. To support him, we use formal concept analysis with a formal context created from the set of tables and fields extracted from the database schema. The concept lattice calculated from this formal context is used to analyse the schema and create questions which the user should answer. Each question has an associated set of actions aimed at redesigning the database schema model.

The application is implemented over an excerpt of the Chilean National Council of Culture and Arts (CNCA) database for intangible cultural heritage documentation. Currently, we are implementing the process in full scale including more domain experts. Future work include the annotation process of data using the ontological schema created which has already been considered, however not described in this article. As pointed out by one of the reviewers of this work, the approach presented in this article may also be applied in the context of software engineering, specifically in code re-factoring and object-model re-engineering [4].

References

1. Joachim Hereth Correia, Gerd Stumme, Rudolf Wille, and Uta Wille. Conceptual knowledge discovery–a human-centered approach. *Applied Artificial Intelligence*, 17(3):281–302, March 2003.
2. Ramez A Elmasri and Shankrant B Navathe. *Fundamentals of Database Systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3rd edition, 1999.
3. Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, December 1999.
4. Robert Godin and Petko Valtchev. Formal concept analysis-based class hierarchy design in object-oriented software development. In *Formal Concept Analysis*, volume 3626 of *Lecture Notes in Computer Science*. 2005.
5. Joachim Hereth. Relational scaling and databases. In *Conceptual Structures: Integration and Interfaces*, volume 2393 of *Lecture Notes in Computer Science*, pages 62–76. Springer Berlin Heidelberg, 2002.
6. Sergei O Kuznetsov. On stability of a formal concept. *Annals of Mathematics and Artificial Intelligence*, 49(1-4):101–115, 2007.
7. Brian McBride. The resource description framework (rdf) and its vocabulary description language rdfs. In *Handbook on Ontologies*, International Handbooks on Information Systems, pages 51–65. Springer Berlin Heidelberg, 2004.
8. Uta Priss. Efficient implementation of semantic relations in lexical databases. *Computational Intelligence*, 15:79–87, 1999.
9. Uta Priss. Establishing connections between formal concept analysis and relational databases. In *Common Semantics for Sharing Knowledge: Contributions to ICCS 2005*, pages 132–145, 2005.
10. Camille Roth, Sergei Obiedkov, and Derrick Kourie. Towards concise representation for taxonomies of epistemic communities. *Concept Lattices and Their Applications*, 2008.
11. 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*, 2007.
12. Rudolf Wille. Why can concept lattices support knowledge discovery in databases? *Journal of Experimental and Theoretical Artificial Intelligence*, 14(2-3):81–92, 2002.

Analogical proportions and the factorization of information in distributive lattices

Nelly Barbot[1], Laurent Miclet[1], and Henri Prade[2]

[1] IRISA, University Rennes 1
ENSSAT, 6 rue Kerampont, 22305 Lannion, France
[2] IRIT, Université Paul Sabatier
118 route de Narbonne, 31062 Toulouse cedex 9, France

Abstract. Analogical proportions are statements involving four entities, of the form 'A is to B as C is to D'. They play an important role in analogical reasoning. Their formalization has received much attention from different researchers in the last decade, in particular in a propositional logic setting. Analogical proportions have also been algebraically defined in terms of factorization, as a generalization of geometric numerical proportions (that equate ratios). In this paper, we define and study analogical proportions in the general setting of lattices, and more particularly of distributive lattices. The decomposition of analogical proportions in canonical proportions is discussed in details, as well as the resolution of analogical proportion equations, which plays a crucial role in reasoning. The case of Boolean lattices, which reflects the logical modeling, and the case corresponding to entities described in terms of gradual properties, are especially considered for illustration purposes.

Keywords: analogical proportion, lattice, factorization

1 Introduction

Analogical reasoning [1] plays an important role in human reasoning. It enables us to draw plausible conclusions by exploiting parallels between situations, and as such has been studied in AI for a long time, e.g., [2, 3] under various approaches [4]. A key pattern which is associated with the idea of analogical reasoning is the notion of analogical proportions, i. e. statements of the form 'A is to B as C is to D'. However, it is only in the last decade that researchers working in computational linguistics have started to study these proportions in a formal way [5–7]. More recently, analogical proportions have been shown as being of particular interest for classification tasks [8] or for solving IQ tests [9]. Moreover, in the last five years, there has been a number of works, e.g., [10, 11] studying the propositional logic modeling of analogical proportions. The logical view of an analogical proportion amounts to expressing that the difference between A and B (resp. B and A) is the same as the difference between C and D (resp. D and C). Although it can be proved that, beside symmetry, this view agrees with a crucial postulate of analogical proportions, namely that one can exchange B and C in the proportion (as well as A and D), it is not straightforward to see that it holds. In fact, a genuine parallel can be made between analogical proportions and numerical proportions. It suggests that since factorization plays a key role in geometric proportions (which equal two ratios of integers), factorization also makes sense for analogical proportions. This idea is investigated here in the abstract setting of lattices.

In order to do this, we go back to a factorization-based formalization of analogical proportions proposed in [12, 13]. On this basis, these authors proposed a definition of analogical proportions in different settings such as sets, sets of sequences, set of trees, and lattices. As shown in this paper their definition suggested for the lattice setting is incomplete. We then correct and complete this definition. We show that it encompasses the Boolean lattice case that corresponds to the propositional logic encoding of analogical proportions. We then study the more general setting of distributive lattices, identify canonical proportions, and show how analogical proportions can be decomposed into canonical ones, before discussing the solving of analogical proportion equations, a key issue for application to algorithms for analogical reasoning. We also illustrate the approach in the case of a distributive lattice induced by fuzzy sets.

The paper is organized as follows. The next section provides the necessary background on lattices and on analogical proportions. Section 3 establishes the basic form of analogical proportions in distributive lattices, which is illustrated on Boolean and on graded proportions, and then investigates their basic properties. Section 4 introduces the notion of canonical proportions and takes advantage of them for studying the composition and the decomposition of analogical proportions. Section 5 discusses the resolution of analogical equations, and briefly studies the transitivity of analogical proportions.

This paper is a preliminary investigation into the connexions between lattices and analogical proportion. In particular, we are interested in detecting analogical proportions in *concept lattices* (e.g. see [20]). However, since these lattices are generally non distributive, we will have to investigate which of the properties demonstrated here still hold true in concept lattices, and which of them have to be abandoned or weakened. A few hints are given in Sections 3 and 5.

2 Background: Lattices and analogical proportions

Lattices. They are mathematical structures commonly encountered in the semantics of representation and programming languages, in formal concept analysis, machine learning, data mining, and in other areas of computer sciences.

$(L, \vee, \wedge, \leq)$ is a *lattice* when [14]: i) L has at least two elements, ii) $\wedge$ and $\vee$ are two binary internal operations, both idempotent, commutative, associative, and satisfying the absorption laws. A lattice is *distributive* when $u \vee (v \wedge w) = (u \vee v) \wedge (u \vee w)$, or equivalently $u \wedge (v \vee w) = (u \wedge v) \vee (u \wedge w)$ for all u, v and w in L. A *bounded* lattice has a greatest (or maximum) and least (or minimum) element, denoted $\top$ and $\bot$. A bounded lattice is *complemented* if each element x has a complementary y such that $x \wedge y = \bot$ and $x \vee y = \top$. A distributive, bounded and complemented lattice is called a *Boolean* lattice.

Duality theorem. If a theorem T is true in a lattice, then the dual of T is also true. This dual is obtained by replacing all occurrences of $\wedge$ (resp. $\vee$, $\leq$) by $\vee$ (resp. $\wedge$, $\geq$).

Examples. (a) $(2^{\Sigma}, \cap, \cup, \subseteq)$, where Σ is a finite set (alphabet), is a Boolean lattice. (b) $(\mathbb{N}^{+}, \mathrm{gcd}, \mathrm{lcm}, |)$ where $(x \mid y)$ iff x divides y is a distributive lattice, with the minimum element 1 but no maximum element. (c) The set $\mathcal{S}$ of closed intervals on $\mathbb{R}$, including $\emptyset$ and $\mathbb{R}$, is a non-distributive lattice when $\wedge$ is the intersection and $[a,b] \vee [c,d] = [\min(a,c), \max(b,d)]$, where min and max are defined according to the order in $\mathbb{R}$.

Analogical proportions. They are characterized by three axioms. They acknowledge the symmetrical role played by the pairs (A,B) and (C,D) in the proportion 'A is to B as C is to D', and enforce the idea that B and C can be interchanged if the proportion is valid, just as in the equality of two numerical ratios where means can be exchanged. This view dates back to Aristotle [15]. A third, optional, axiom insists on the unicity of the solution $x = B$ for completing the analogical proportion $A : B :: A : x$. These axioms are studied in [16].

Definition 1 (Analogical proportion) *An analogical proportion*[3] *(AP) on a set X is a quaternary relation on X, i.e. a subset of X^4. An element of this subset, written $A : B :: C : D$, which reads 'A is to B as C is to D', must obey the following two axioms:*

1) Symmetry of 'as'*:* $A : B :: C : D \quad \Leftrightarrow \quad C : D :: A : B$

2) Exchange of means*:* $A : B :: C : D \quad \Leftrightarrow \quad A : C :: B : D$

Then, thanks to symmetry, it can be easily seen that $A : B :: C : D \Leftrightarrow D : B :: C : A$ should also hold (exchange of the extremes). According to the first two axioms, five other formulations are equivalent to the canonical form $A : B :: C : D$, $B : A :: D : C$, $D : B :: C : A$, $C : A :: D : B$, $D : C :: B : A$ and $B : D :: A : C$.

Example. Let us take the lattice $(2^{\Sigma}, \cup, \cap, \subseteq)$, where Σ is a finite set $\{a, \ldots, n\}$. Σ may be for example a set of Boolean properties, and a subset of Σ can be used to characterize some object described by the corresponding properties. The four objects described by the subsets $x = \{a,b,e\}$, $y = \{b,c,e\}$, $z = \{a,d,e\}$ and $t = \{c,d,e\}$ are in analogical proportion in this order. Indeed, it suggests an intuitive meaning for 'is to': To transform x into y, one has to remove property a and to include property c; namely $x \setminus y = \{a\}$ and $y \setminus x = \{c\}$. z is transformed into t by exactly the same operations; namely $z \setminus t = \{a\}$ and $t \setminus z = \{c\}$. Such a view of the relation linking x, y, z, t is clearly symmetrical, and satisfies the exchange of the means: namely $x \setminus z = \{b\}$, $z \setminus x = \{d\}$ and $y \setminus t = \{b\}$, $t \setminus y = \{d\}$. This idea that x (resp. y) differs from y (resp. x) in the same way as z (resp. t) differs from t (resp. z) is at the core of the definition of the analogical proportion $x : y :: z : t$ in the Boolean setting [10], as further discussed in the following.

---

[3] When there is no ambiguity, an analogical proportion is also called a *proportion.*

Proportions in commutative semigroups. Stroppa and Yvon [12, 13] have given another definition of the analogical proportion, based on the notion of *factorization*, when the set of objects is a commutative semigroup $(X, \oplus)$.

Definition 2 *A 4-tuple* (x, y, z, t) *in a commutative semigroup* $(X, \oplus)$ *is an AP* $x : y :: z : t$ *when:*
1) either $(y, z) \in \{(x, t), (t, x)\}$,
2) or there exists $(x_1, x_2, t_1, t_2) \in X^4$ *such that* $x = x_1 \oplus x_2$, $y = x_1 \oplus t_2$, $z = t_1 \oplus x_2$ *and* $t = t_1 \oplus t_2$.

This definition satisfies the two basic axioms of the analogical proportion (Definition 1). For example, in $(X, \oplus) = (\mathbb{N}^+, \times)$, with $x_1 = 2$, $x_2 = 3$, $t_1 = 5$ and $t_2 = 7$, one has $(2\times3) : (2\times7) :: (5\times3) :: (5\times7)$, i.e. $6 : 14 :: 15 : 35$, a numerical geometric analogical proportion. Note that this particular proportion corresponds equivalently to the equality: $6 \times 35 = 14 \times 15$.

3 Analogical proportion in lattices

In this section, we are interested in studying how the definition of an analogical proportion by factorization applies to lattices. In particular we are wondering whether the equivalence of the two formulations in the preceding example can be transposed to this algebraic structure.

3.1 Definition

Considering that a lattice $(L, \vee, \wedge)$ is both a commutative semigroup $(L, \vee)$ and $(L, \wedge)$, we define an analogical proportion as follows.

Definition 3 *A 4-tuple* (x, y, z, t) *in* $(L, \vee, \wedge)$ *is an AP* $(x : y :: z : t)$ *when:*
1) there exists $(x_1, x_2, t_1, t_2) \in X^4$ *such that* $x = x_1 \vee x_2$, $y = x_1 \vee t_2$, $z = t_1 \vee x_2$ *and* $t = t_1 \vee t_2$,
2) and there exists $(x'_1, x'_2, t'_1, t'_2) \in X^4$ *such that* $x = x'_1 \wedge x'_2$, $y = x'_1 \wedge t'_2$, $z = t'_1 \wedge x'_2$ *and* $t = t'_1 \wedge t'_2$.

Note that when $x_2 = t_2$ then $y = x$ and $z = t$ and that when $x_1 = t_1$ then $y = t$ and $z = x$. Hence we can have $(y, z) = (x, t)$ or $(y, z) = (t, x)$.

Examples. (a) In $(\mathbb{N}^+, \mathrm{gcd}, \mathrm{lcm}, |)$, we have $(20 : 4 :: 60 : 12)$, with $x_1 = 20$, $x_2 = t_1 = 60$, $t_2 = 12$, $x'_1 = t'_2 = 4$, $x'_2 = 20$ and $t'_1 = 12$. (b) In the lattice $\mathcal{S}$ of closed intervals on $\mathbb{R}$, we have $([0, 3] : \{3\} :: [0, 4] : [3, 4])$ with $x_1 = \{3\}$, $x_2 = \{0\}$, $t_1 = [3, 4]$, $t_2 = \emptyset$, $x'_1 = [0, 3]$, $x'_2 = [0, 4]$, $t'_1 = [0, 4]$ and $t'_2 = [3, 4]$.

Proposition 1 *A 4-tuple* (x, y, z, t) *in* $(L, \vee, \wedge)$ *is an AP* $(x : y :: z : t)$ *iff:*

$$\begin{array}{ll}
x = (x \wedge y) \vee (x \wedge z) & \quad x = (x \vee y) \wedge (x \vee z) \\
y = (x \wedge y) \vee (y \wedge t) & \quad y = (x \vee y) \wedge (y \vee t) \\
z = (z \wedge t) \vee (x \wedge z) & \quad z = (z \vee t) \wedge (x \vee z) \\
t = (z \wedge t) \vee (y \wedge t) & \quad t = (z \vee t) \wedge (y \vee t)
\end{array}$$

Proof. (⇒). Taking $x_1 = x \wedge y$, $x_2 = x \wedge z$, $t_1 = z \wedge t$ and $t_2 = y \wedge t$ show directly that there exist factors satisfying Definition 3.
(⇐). Let us show that $x = (x \wedge y) \vee (x \wedge z)$. Since $x = x_1 \vee x_2$ and $y = x_1 \vee t_2$, we have $x_1 \leq x$ and $x_1 \leq y$. Then $x_1 \leq x \wedge y$. Similarly, factor x_2 satisfies $x_2 \leq x \wedge z$. Hence, $x \leq (x \wedge y) \vee (x \wedge z)$. Besides, x being greater than $(x \wedge y)$ and $(x \wedge z)$, $(x \wedge y) \vee (x \wedge z) \leq x$. The antisymmetry of $\leq$ implies that $x = (x \wedge y) \vee (x \wedge z)$. We show the other equalities in the same manner. □

The above definition applies to general lattices. In this paper, we focus on distributive lattices, since most of the properties to come require this property.

Boolean lattices

Every finite Boolean lattice is isomorphic to the lattice $(X, \cup, \cap, \subseteq)$, where X is a finite set. When considering this lattice, the quantities involved in Definition 1 can be described more precisely (see [16, 17]), as explained below.

Proposition 2 *A 4-tuple* (x, y, z, t) *in the Boolean lattice* $(2^{\Sigma}, \cup, \cap, \subseteq)$ *is in the AP* $(x : y :: z : t)$ *iff there exists a partition of* Σ *composed of six subsets* (a, b, c, d, e, f) *such that* $x = a \cup c \cup e$, $y = b \cup c \cup e$, $z = a \cup d \cup e$ *and* $t = b \cup d \cup e$.

The link with Definition 3 is made by taking[4].: $x_1 = c \cup e$, $x_2 = a \cup e$, $t_1 = d \cup e$ and $t_2 = b \cup e$, and by duality: $x'_1 = \bar{d} \cap \bar{f}$, $x'_2 = \bar{b} \cap \bar{f}$, $t'_1 = \bar{c} \cap \bar{f}$ and $t'_2 = \bar{a} \cap \bar{f}$.

It is also easy to check that this definition is equivalent to Definition 3.

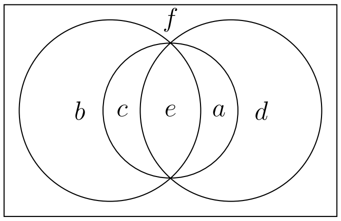


Fig. 1. An *AP* $(x : y :: z : t)$ in a Boolean lattice. $x = a \cup c \cup e$, $y = b \cup c \cup e$, $z = a \cup d \cup e$ and $t = b \cup d \cup e$.

It is worth noticing that the above result has a nice interpretation in practice. Let us view x, y, z, t as subsets of properties that hold true in four different situations. It is then clear that a is the subset of properties that are true in the first situation, but false in the second one, and again true in the third situation and false in the fourth one. Conversely, b is the subset of properties that are false in the first situation, true in the second one, and again false in the third situation and true in the fourth one. Besides, c (resp. d) is the set of properties that are true for both the first and the second situations and false for the third and the fourth ones (resp. false for the first and the second situations and true for the third and the fourth ones. In other words, the disjoint subsets a, b, c, d, e, f have the following interpretations $a = x \setminus y = z \setminus t$, $b = y \setminus x = t \setminus z$, $c \cup e = x \cap y$, $d \cup e = z \cap t$, where $e = x \cap y \cap z \cap t$ is the set of properties that are

---

[4] We denote the complement in 2^{Σ} with an overline

true in all situations (and f the set of properties that are false in all situations); see Figure 1. Thus, one can say that x, y, z, and t are respectively factorized under the form of pairs of disjoint subsets, namely $(a, c \cup e)$ for x, $(b, c \cup e)$ for y, $(a, d \cup e)$ for z, and $(b, d \cup e)$ for x, which perfectly parallels the equality of two numerical ratios of the form $\frac{\alpha\times\gamma}{\beta\times\gamma} = \frac{\alpha\times\delta}{\beta\times\delta}$.

Moreover, the above decomposition using the partition of the referential into six subsets exactly corresponds to the truth table of the analogical proportion in a propositional setting [10, 18] defined equivalently by

$x : y :: z : t \ = (x \wedge \neg y) \equiv (z \wedge \neg t) \ \wedge \ (y \wedge \neg x) \equiv (t \wedge \neg z)$

or $\quad x : y :: z : t \ = (x \wedge t) \equiv (y \wedge z) \ \wedge \ (x \vee t) \equiv (y \vee z)$.

Indeed, in the Boolean lattice associated to the two truth values $0, 1$, $\ x : y :: z : t\ $ is true (i.e., is equal to '1') for the six patterns $(x, y, z, t) = (1, 0, 1, 0)$, $(x, y, z, t) = (0, 1, 0, 1)$, $(x, y, z, t) = (1, 1, 0, 0)$, $(x, y, z, t) = (0, 0, 1, 1)$, $(x, y, z, t) = (1, 1, 1, 1)$ and $(x, y, z, t) = (0, 0, 0, 0)$, and false for the ten other possible patterns which are $(x, y, z, t) = (1, 0, 0, 1)$, $(x, y, z, t) = (0, 1, 1, 0)$ and the eight patterns having an odd number of '1' and '0' (e.g., $(x, y, z, t) = (0, 0, 1, 0)$ or $(x, y, z, t) = (0, 1, 1, 1)$). The six above patterns which make $\ x : y :: z : t\ $ true clearly correspond to the subsets a, b, c, d, e, f.

The case of graded properties

Analogical proportions have been also extended when properties are graded on a chain which is finite, or such as the unit interval $[0, 1]$ [19]. For instance, a property may be half-true. Then, in the case of a finite chain with three elements $\{0, \omega, 1\}$, two views make sense, for which the patterns having truth value '1' are respectively

- the 15 patterns, that includes the 6 of the binary case $(1, 0, 1, 0)$, $(0, 1, 0, 1)$, $(1, 1, 0, 0)$, $(0, 0, 1, 1)$, $(1, 1, 1, 1)$, $(0, 0, 0, 0)$, together with their 9 counterparts $(\omega, 0, \omega, 0)$, $(0, \omega, 0, \omega)$, $(1, \omega, 1, \omega)$, $(\omega, 1, \omega, 1)$, $(\omega, \omega, 0, 0)$, $(0, 0, \omega, \omega)$, $(1, 1, \omega, \omega)$, $(\omega, \omega, 1, 1)$, $(\omega, \omega, \omega, \omega)$
- the 15 above patterns together with the 4 additional ones $(1, \omega, \omega, 0)$, $(0, \omega, \omega, 1)$, $(\omega, 0, 1, \omega)$, $(\omega, 1, 0, \omega)$.

In the second view, we acknowledge the fact that when there is a change from x to y there is *the same* change from z to t, and otherwise there is no change between x and y, and between z and t, but also the fact that the proportion still holds when the change from x to y has the same direction and intensity as the change from z to t (considering that ω is exactly in the "middle" between 0 and 1). It is easy to see that the lattice-based definition proposed here agrees with the first view only, while the 4 additional patterns do not make analogical proportions.

In the case of the unit interval $[0, 1]$, this leads to the following graded view of the analogical proportion:

$x : y :: z : t \ = \min(1 - |\min(x, t) - \min(y, z)|, 1 - |\max(x, t) - \max(y, z)|)$.

It is easy to see that the above definition is a direct counterpart of the second form of the propositional expression of the analogical proportion given above. Moreover, it is equal to 1 only for the 15 patterns mentioned above.

3.2 Basic properties

We show here that in distributive lattices, a 4-tuple in analogical proportion is such that "the product of the means is equal to the product of the extremes".

Proposition 3 *In a distributive lattice,* $(x : y :: z : t)$ *is an AP iff:*

$$y \wedge z \leq x \leq y \vee z,\ x \wedge t \leq y \leq x \vee t,\ x \wedge t \leq z \leq x \vee t \text{ and } y \wedge z \leq t \leq y \vee z \qquad (1)$$

Proof. (⇒). Using the derivations of x given in Proposition 1, we have $x = x \wedge (y \vee z)$ and $x = x \vee (y \wedge z)$ by distributivity and then $y \wedge z \leq x \leq y \vee z$. The other inequalities are similarly derived.
(⇐). By distributivity, $(x \wedge y) \vee (x \wedge z) = x \wedge (y \vee z)$. Moreover, $x \wedge (y \vee z) = x$ since $x \leq y \vee z$. The other equalities are obtained in the same way. □

The next property is a stronger result: the four values of the bounds in the preceding property are actually only two.

Proposition 4 *In a distributive lattice,* $(x : y :: z : t)$ *is an analogical proportion iff* $x \vee t = y \vee z$ *and* $x \wedge t = y \wedge z$*.*

Proof. (⇒). Using the expressions of x, y, z and t given by Proposition 1, we easily check that $x \vee t = y \vee z$ and $x \wedge t = y \wedge z$.

(⇐). By absorption law and distributivity, we have $x = x \wedge (x \vee t) = x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$. The other equations of Proposition 1 can be obtained in a similar way. □

Comment 1. In [13, 7], an incomplete definition of a proportion in a lattice has been given. Actually, only four equalities of Definition 3 were given, and only four equalities of Proposition 1 were demonstrated (in a different manner than here). This definition was flawed, since for example in the lattice $(\{0, 1\}, \vee, \wedge)$ it would have given $(0 : 1 :: 1 : 1)$ as a proportion, although it does not satisfy the basic axioms. In the particular case of Boolean lattices, Proposition 4 has been shown in [10].

Comment 2. If the lattice is not distributive, Proposition 4 is not an equivalence, but an implication. For example, let us consider the elements $x = [2, 3]$, $y = [2, 6]$, $z = [8, 9]$ and t=[6, 9] of the lattice of closed intervals on $\mathbb{R}$. We have $x \vee t = y \vee z$ and $x \wedge t = y \wedge z$ but the conditions of Definition 3 are not satisfied. We are currently studying in general lattices and concept lattices the properties of what can be called a *weak analogical proportion*, namely the fact that four elements are linked by the equalities $x \vee t = y \vee z$ and $x \wedge t = y \wedge z$.

3.3 Determinism

The first and second axioms of Definition 1 are straightforwardly verified by Definition 3. What about the third axiom?

Proposition 5 (Determinism in a distributive lattice) *Let* x *and* y *be two elements of a distributive lattice, the equation in* z*:* $(x : x :: y : z)$ *has the unique solution* $z = y$*. This is also true for the equation* $(x : y :: x : z)$*.*

Proof. Let us consider a solution z of $(x : x :: y : z)$. From Proposition 4, we have

$$x \wedge z = x \wedge y \quad \text{and} \quad x \vee z = x \vee y\,. \tag{2}$$

Besides, using absorption law, $z = (x \vee z) \wedge z$. Consequently, using (2) and distributivity, $z = (x \vee y) \wedge z = (x \wedge z) \vee (y \wedge z)$. Then, using (2), distributivity and absorption, we can conclude: $z = (x \wedge y) \vee (y \wedge z) = (x \vee z) \wedge y = (x \vee y) \wedge y = y$. □

4 Composition and decomposition of analogical equations

We present in this section a particular case of analogical proportion which will be shown later (see section 4) to be a "building block" of the general proportion.

Proposition 6 (Canonical proportions) *Let y and z be two arbitrary elements of a lattice. Then the following analogical proportion is true:*

$$y : y \vee z :: y \wedge z : z\,. \tag{3}$$

Proof. Equations of Proposition 4 are straightforwardly satisfied. □

In the following, we will call this particular analogical proportion a *canonical* analogical proportion (CAP). Note that the previous property holds in general lattices, not only distributive lattices.

In general, analogical proportions in a lattice are not canonical, such as $(14 : 21 :: 10 : 15)$ in $(\mathbb{N}^{+}, \mathrm{gcd}, \mathrm{lcm}, |)$.

Note that a canonical proportion can be written in eight different forms (see Definition 1), by applying the axioms of analogical proportion. We suppose in the following that one of the two particular following forms are used: $y : y \vee z :: y \wedge z : z$ or $z : y \vee z :: y \wedge z : y$. This form is called the $CAP1$ form, as opposed to the $CAP2$ one: $y : y \wedge z :: y \vee z : z$ or $z : y \wedge z :: y \vee z : y$.

We are interested in here in defining primitive proportions, that will be used as "building blocks" of the general proportion. This is done in particular to enlighten primitive chunks in a process of reasoning by analogy.

Definition 4 *Let $\mathtt{a} = (x, y, z, t)$ and $\mathtt{A} = (X, Y, Z, T)$ be two 4-tuples of a distributive lattice $(L, \vee, \wedge)$. We define the $\boxed{\vee}$-composition and the $\boxed{\wedge}$-composition of these two 4-tuples as the 4-tuples:*

$$\mathtt{a} \boxed{\vee} \mathtt{A} = (x \vee X, y \vee Y, z \vee Z, t \vee T) \quad \textit{and} \quad \mathtt{a} \boxed{\wedge} \mathtt{A} = (x \wedge X, y \wedge Y, z \wedge Z, t \wedge T)$$

Note that these operations are commutative and associative.

Definition 5 *A* degenerated *analogical proportion (DAP) is $(x : x :: x : x)$.*
A simple *analogical proportion (SAP) is $(x : y :: x : y)$ (SAP1)*
or $(x : x :: y : y)$ (SAP2).

The next results are all established in a distributive lattice.

Proposition 7 (Composition of an AP and a DAP) *The composition of an AP by a DAP is a AP.*

Proof. Using Proposition 4 and distributivity. □

This property is a generalisation of a property in Boolean lattices, shown in [10]. Analogical proportions are not closed for general composition, as shown below.

Note that the composition of two AP's is not necessarily an AP (nor is the composition of an AP and a SAP) and that the composition of two CAP's is not necessarily an AP.

Proposition 8 *In a distributive lattice, for every AP $\mathtt{a}_1$ there exists a $SAP1$ $\mathtt{a}_2$ and a $SAP2$ $\mathtt{a}_3$ such that $\mathtt{a}_1 = \mathtt{a}_2 \boxed{\vee} \mathtt{a}_3$. There also exists a $SAP1$ $\mathtt{a}_3$ and a $SAP2$ $\mathtt{a}_4$ such that $\mathtt{a}_1 = \mathtt{a}_3 \boxed{\wedge} \mathtt{a}_4$.*

Proof. We check that $(x : y :: z : t)$ is the $\boxed{\wedge}$-composition of $(x \vee y) : (x \vee y) :: (z \vee t) : (z \vee t)$ and $(x \vee z) : (y \vee t) :: (x \vee z) : (y \vee t)$, by proposition 1. It is also the $\boxed{\vee}$-composition of $(x \wedge y) : (x \wedge y) :: (z \wedge t) : (z \wedge t)$ and $(x \wedge z) : (y \wedge t) :: (x \wedge z) : (y \wedge t)$. □

Proposition 9 *In a distributive lattice, for every AP $\mathtt{a}_1$ there exists a $CAP1$ $\mathtt{a}_2$ and a $CAP2$ $\mathtt{a}_3$ such that $\mathtt{a}_1 = \mathtt{a}_2 \boxed{\vee} \mathtt{a}_3$. There exists also a $CAP1$ $\mathtt{a}_3$ and a $CAP2$ $\mathtt{a}_4$ such that $\mathtt{a}_1 = \mathtt{a}_3 \boxed{\wedge} \mathtt{a}_4$.*

Proof. We check that $(x : y :: z : t)$ is the $\boxed{\vee}$-composition of $(x \wedge y) : y :: (x \wedge y \wedge z \wedge t) : (y \wedge t)$ and $(x \wedge z) : (x \wedge y \wedge z \wedge t) :: z : (z \wedge t)$ and the $\boxed{\wedge}$-composition of $(x \vee y) : y :: (x \vee y \vee z \vee t) : (y \vee t)$ and $(x \vee z) : (x \vee y \vee z \vee t) :: z : (z \vee t)$. □

Proposition 10 *In a distributive lattice, for every AP $\mathtt{a}_1$ there exists a $CAP1$ $\mathtt{a}_2$ such that $\mathtt{a}_1 \boxed{\vee} \mathtt{a}_2$ is a $CAP2$.*

Proof. Let $\mathtt{a}_1 = (x : y :: z : t)$, and take $\mathtt{a}_2 = ((z \wedge t) : z :: (x \wedge y \wedge z \wedge t) : (x \wedge z))$.

We have to show that $[x \vee (z \wedge t)] : (y \vee z) :: z : [t \vee (x \wedge z)]$. According to property 1, we show equivalently the two equalities: $[x \vee (z \wedge t)] \vee [t \vee (x \wedge z)] = (y \vee z) \vee z$ and $[x \vee (z \wedge t)] \wedge [t \vee (x \wedge z)] = (y \vee z) \wedge z$. For the second: $x \vee (z \wedge t)] \wedge [t \vee (x \wedge z) = [(x \vee z) \wedge (x \vee t)] \wedge [(t \vee x) \wedge (t \vee z)] = (x \vee z) \wedge (t \vee z) \wedge (x \vee t) = z \wedge (x \vee t) = z \wedge (y \vee z)$.

The first equality has a similar demonstration. □

5 Resolution of analogical equations. Transitivity

In this section, we answer the following question: given three elements of an AP, can we find the fourth one? This is an important issue in analogical reasoning.

Let us suppose that in a distributive lattice we know three elements a, m and M. We are looking for an x satisfying the couple of equations:

$$a \vee x = M \quad \text{and} \quad a \wedge x = m \tag{4}$$

This is a more general question that wondering whether the analogical equation in a distributive lattice $(a : b :: c : x)$ has solutions, since we can take $M = b \vee c$ and $m = b \wedge c$.

Proposition 11 (Unicity of the solution) *When there is a solution to equations 4 in a distributive lattice, then it is unique. Consequently, if there exists a solution to the analogical equation* $(a : b :: c : x)$*, then it is unique.*

Proof. Supposing the equations have two solutions x_1 and x_2 leads to a contradiction with the distributivity between a, x_1 and x_2.

□

Proposition 11 doesn't hold in general lattices: eq. 4 may have several solutions.

Proposition 12 *Let* a*,* m *and* M *be three elements of a distributive lattice such that* $m \leq a \leq M$*. If there exists* $\overset{\diamond}{a}$ *such that:* $(\overset{\diamond}{a} \vee a \geq M)$ *and* $(\overset{\diamond}{a} \wedge a \leq m)$ *then* $x = (M \wedge \overset{\diamond}{a}) \vee m = (m \vee \overset{\diamond}{a}) \wedge M$ *is the unique solution to equations (4).*

Proof. Firstly, we show that $x \wedge a = m$ with the equalities: $x \wedge a = [(M \wedge \overset{\diamond}{a}) \vee m] \wedge a = (M \wedge \overset{\diamond}{a} \wedge a) \vee (m \wedge a) = (\overset{\diamond}{a} \wedge a) \vee m = m$.

Secondly, the equality $x \vee a = M$ is demonstrated in the same manner, using $M \leq (\overset{\diamond}{a} \vee a)$ instead of $(\overset{\diamond}{a} \wedge a) \leq m$. Then $x = (M \wedge \overset{\diamond}{a}) \vee m$ is the solution.

Thirdly, we show in the same manner that $x = (m \vee \overset{\diamond}{a}) \wedge M$ is a solution to (4). Since the solution is unique, the property is demonstrated.

□

When two or three elements are comparable, the solutions of the analogical equation are severely constrained.

Proposition 13 *Let* x*,* y*,* z *and* t *be four elements of a distributive lattice such as* $(x : y :: z : t)$*. If the three first elements are comparable then this AP is a* SAP *or a* CAP*. More precisely,* $(x : y :: z : t)$ *is*

1) $(y \wedge z : y :: z : y \vee z)$ *if* $x \leq y \wedge z$*, and* $(y \vee z : y :: z : y \wedge z)$ *if* $x \geq y \vee z$*. In particular, it is a* $SAP1$ *if* $y \leq z \leq x$ *or* $x \leq z \leq y$*, and a* $SAP2$ *if* $z \leq y \leq x$ *or* $x \leq y \leq z$

2) a $CAP1$ *(resp.* $CAP2$ *)if* $z \leq x \leq y$ *(resp.* $y \leq x \leq z$*).*

Proof.

1) We have from (1) $y \wedge z \leq x \leq y \vee z$. Let us consider the case where $x \leq y \wedge z$. We then have $x = y \wedge z$ and we can easily check that $t = y \vee z$ is solution of equations $x \vee t = y \vee z$ and $x \wedge t = y \wedge z$. Then, using Propositions 4 and 11, $t = y \vee z$ is the unique solution of $(y \wedge z : y :: z : t)$. Moreover, if $x \leq z \leq y$, $t = y \vee z = y$ and then $x = z$. The other cases have a similar demonstration.

2) If $z \leq x \leq y$, $y = x \vee t$ and $z = x \wedge t$ using Proposition 4,

3) The reasoning is similar to the previous one.

□

In the Boolean case, we recall a previous result.

Proposition 14 ([16]) *The analogical equation in t:* $(x : y :: z : t)$ *has a solution in a Boolean lattice if and only if* $y \cap z \subseteq x \subseteq y \cup z$*. In this case, the unique solution is* $t = ((y \cup z) \backslash x) \cup (y \cap z)$*.*

Finally, let us investigate transitivity, which propagates (dis)similarity. In the Boolean case, $(a : b) :: (c : d) :: (e : f)$ holds for general proportions [10]. In the distributive case, $CAP1$ (resp. $CAP2$) are transitive. Proof is omitted due to space limitation.

Proposition 15 (Transitivity of CAP) *If* $(x : (x \vee t) :: (x \wedge t) : t)$ *and* $((x \wedge t) : t :: u : v)$ *are two canonical proportions of form* $CAP1$ *(resp.* $CAP2$*), then* $x : (x \vee t) :: u : v$ *is a canonical proportion of form* $CAP1$ *(resp.* $CAP2$*).*

Conjecture 1 (Non transitivity of proportions) *If* $(x : y :: z : t)$ *and* $(z : t :: u : v)$ *are two analogical proportions in a distributive lattice, it does not necessarily imply that* $(x : y :: u : v)$ *is an analogical proportion.*

We have not found any example to show this property, albeit the transitivity seems impossible to prove. Therefore, the non transitivity in a general distributive lattice is a conjecture. However, we have found an example to show that transitivity doesn't hold in general in a non transitive lattice.

We have not found any counter example to show this property. We conjecture there is no transitivity in distributive lattices. Indeed, in a non distributive lattice, transitivity does not holds, as shown in the following example. In $\mathcal{S}$ (see section 2) we have $[0,3] : \{3\} :: \{0\} : \emptyset$ by considering Definition 3 and $x_1 = \{3\}$, $x_2 = \{0\}$, $t_1 = \emptyset$, $t_2 = \emptyset$, $x'_1 = x'_2 = [0,3]$, $t'_1 = \{0\}$ and $t'_2 = \{3\}$. Similarly, we have $\{0\} : \emptyset :: [0,4] : \{4\}$ using $x_1 = \emptyset$, $x_2 = \{0\}$, $t_1 = \{4\}$, $t_2 = \emptyset$, $x'_1 = \{0\}$, $x'_2 = [0,4]$, $t'_1 = [0,4]$ and $t'_2 = \{4\}$. However, $[0,3] : \{3\} :: [0,4] : \{4\}$ is not true because it is impossible to satisfy the second condition of Definition 3. Indeed, if there exists four elements x'_1, x'_2, t'_1 and t'_2 of $\mathcal{S}$ such that $[0,3] = x'_1 \wedge x'_2$, $\{0\} = x'_1 \wedge t'_2$, $[0,4] = t'_1 \wedge x'_2$ and $\{4\} = t'_1 \wedge t'_2$, the closed interval t'_2 contains 0 and 4 and then $[0,4] \subset t'_2$. Moreover, $[0,4] \subset t'_1$. Consequently, $t'_1 \wedge t'_2 \neq \{4\}$.

6 Conclusion

The results of this paper provide a better understanding of analogical proportions in the general setting of lattices structures. In particular, it relates a factorization-based view of analogical proportions to its propositional logical reading in the case of Boolean lattices. For graded proportions, where the underlying lattice of grades is a chain, it leads to consider that the only fully valid logical proportions are of the form $x : y :: x : y$ (and $x : x :: y : y$) where x and y are elements in the chain. It acknowledges the fact that the change should be exactly the same on both sides of the proportion in order to make it (completely) valid, an idea which is for instance (successfully) at work in [9]. The paper has also introduced canonical forms of analogical proportions that are instrumental in the decomposition of analogical proportions in distributive lattices. The unicity of the solution of an analogical proportion equation when it exists, is a important property that is preserved in distributive lattices, and which enables us to generate accurate conclusions.

Generally speaking, the results presented should be useful to design algorithms helping to propagate information in lattices, especially for purposes of reasoning and learning. Moreover, in [20] a first attempt has been provided for relating analogical proportions to formal concept analysis, and searching for analogical proportions that may hold in a formal context by exploiting the lattice structure of the set of formal concepts. This study of analogical proportions in lattice structures should contribute in the long range to a clearer view of the links between these formalizations of the two key cognitive processes that are conceptual categorization and analogical reasoning.

References

1. Gentner, D., Holyoak, K. J., Kokinov, B. N.: The Analogical Mind: Perspectives from Cognitive Science. MIT Press, Cambridge (2001)
2. Hofstadter, D., Mitchell, M.: The Copycat project: A model of mental fluidity and analogy-making. In: Fluid Concepts and Creative Analogies: Computer Models of the Fundamental Mechanisms of Thought, pp. 205–267, Basic Books (1995)
3. Melis E., Veloso M.: Analogy in problem solving. In: Handbook of Practical Reasoning: Computational and Theoretical Aspects, Oxford Univ. Press (1998)
4. French, R. M.: The computational modeling of analogy-making. Trends in Cognitive Sciences, 6(5), 200–205 (2002)
5. Lepage, Y.: Analogy and formal languages. Elec. Notes Theo. Comp. Sci., 53 (2001)
6. Stroppa, N., Yvon, F.: An analogical learner for morphological analysis. Proc. Conf. Comput. Natural Language Learning, pp. 120–127. (2005)
7. Stroppa, N., Yvon, F.: Du quatrième de proportion comme principe inductif : une proposition et son application à l'apprentissage de la morphologie. Traitement Automatique des Langues, 47(2), 1–27 (2006)
8. Miclet, L., Bayoudh, S., Delhay, A.: Analogical dissimilarity: definition, algorithms and two experiments in machine learning. JAIR, 32, 793–824 (2008)
9. Correa, W., Prade, H., Richard, G.: When intelligence is just a matter of copying. In: Eur. Conf. on Artificial Intelligence, pp. 276–281, IOS Press (2012)
10. Miclet, L., Prade, H.: Handling Analogical Proportions in Classical Logic and Fuzzy Logics Settings. Proc. 10th Eur. Conf. on Symbolic and Quantitative Approaches to Reasoning with Uncertainty, Springer, LNCS 5590, pp. 638–650, (2009)
11. Prade, H., Richard, G.: Homogeneous logical proportions: Their uniqueness and their role in similarity-based prediction. Proc. 13th Int. Conf. on Principles of Knowledge Represent. and Reasoning, pp. 402–412. (2012)
12. Stroppa, N., Yvon, F.: Formal Models of Analogical Proportions. Technical report 2006D008, ENST, Paris (2006)
13. Stroppa, N.: Définitions et caractérisations de modèles à base d'analogies pour l'apprentissage automatique des langues naturelles. ENST Paris (2005)
14. Faure, R., Heurgon, E.: Structures Ordonnées et Algèbres de Boole. Gauthier-Villars (1970)
15. Dorolle, M.: Le Raisonnement par Analogie. PUF, Paris (1949)
16. Lepage, Y.: De l'analogie rendant compte de la commutation en linguistique. Habilitation à diriger les recherches, Université de Grenoble (2003)
17. Miclet, L., Delhay, A.: Relation d'analogie et distance sur un alphabet défini par des traits. Technical report 1632, IRISA, Rennes (2004)
18. Prade, H., Richard, G.: Reasoning with logical proportions. Proc. 12th Int. Conf. on Principles of Knowledge Representation and Reasoning, pp. 545–555. (2010)
19. Prade, H., Richard, G.: Multiple-valued logic interpretations of analogical, reverse analogical, and paralogical proportions. Proc. 40th IEEE Int. Symp. on Multiple-Valued Logic, pp 258–263 (2010)
20. Miclet, L., Prade, H., Guennec, D.: Looking for analogical proportions in a formal concept analysis setting. Int. Conf. on Concept Lattices & App., pp. 295–307 (2011)

Boolean Factor Analysis of Multi-Relational Data

Marketa Krmelova, Martin Trnecka

Data Analysis and Modeling Lab (DAMOL)
Department of Computer Science, Palacky University, Olomouc
`marketa.krmelova@gmail.com`, `martin.trnecka@gmail.com`

Abstract. The Boolean factor analysis is an established method for analysis and preprocessing of Boolean data. In the basic setting, this method is designed for finding factors, new variables, which may explain or describe the original input data. Many real-world data sets are more complex than a simple data table. For example almost every web database is composed from many data tables and relations between them. In this paper we present a new approach to the Boolean factor analysis, which is tailored for multi-relational data. We show our approach on simple examples and also propose future research topics.

1 Introduction

Many data sets are Boolean by nature, that is, they contain only 0s and 1s. For example, any data recording the presence (or absence) of variables in observations are Boolean. Boolean data can be seen as a binary data table (or matrix or formal context) C, where the rows represent objects and the columns represent attributes of these objects. Between objects and attributes exists an incidence relation with meaning that an object i has an attribute j and this fact is represented by one in the Boolean table, i.e. $C_{ij} = 1$. If an object i has not an attribute j, than $C_{ij} = 0$.

Many real-word data sets are more complex that a simple data table. Usually, they are composed from many data tables, which are interconnected by relations. An example of such data can be found in almost every sector of human activity. We call this kind of data *multi-relational data*. In this kind of data, this relations are crucial, because they represent additional information about the relationship between data tables and this information is important for understanding data as a whole.

The Boolean factor analysis (BFA) is used for many data mining purposes. The basic task in the BFA is to find new variables, called factors, which may explain or describe original single input data. Finding factors is obviously an important step for understanding and managing data. Boolean nature of data is in this case beneficial especially from the standpoint of interpretability of the results. On the other hand BFA is suitable for single input Boolean data table with just one relation between objects and attributes. The main aim of this work

Published in Manuel Ojeda-Aciego, Jan Outrata (Eds.): CLA 2013, pp. 187–198, ISBN 978–2–7466–6566–8, Laboratory L3i, University of La Rochelle, 2013.

is to present the BFA of multi-relational data, which takes into account relations between data tables and extract more detailed information from this complex data.

2 Preliminaries and basic notions

We assume familiarity with the basic notions of FCA [3]. In this work, we use the binary matrix terminology, because it is more convenient from our point of view. Consider an $n \times m$ object-attribute matrix C with entries $C_{ij} \in \{0,1\}$ expressing whether an object i has an attribute j or not, i.e. C can be understood as a binary relation between objects and attributes. Because there is no danger of confusion we can consider this matrix as a formal context $\langle X, Y, C\rangle$, where X represents a set of n objects and Y represents a set of m attributes.

A formal concept of $\langle X, Y, C\rangle$ is any pair $\langle E, F\rangle$ consisting of $E \subseteq X$ (so-called extent) and $F \subseteq Y$ (so-called intent) satisfying $E^{\uparrow} = F$ and $F^{\downarrow} = E$ where $E^{\uparrow} = \{y \in Y \mid \text{for each } x \in X : \langle x, y\rangle \in C\}$, and $F^{\downarrow} = \{x \in X \mid \text{ for each } y \in Y : \langle x, y\rangle \in C\}$.

The goal of the BMF (the idea from [1, 6]) is to find decomposition

$$C = A \circ B \tag{1}$$

of I into a product of an $n \times k$ object-factor matrix A over $\{0,1\}$, a $k \times m$ matrix B over $\{0,1\}$, revealing thus k factors, i.e. new, possibly more fundamental attributes (or variables), which explain original m attributes. We want $k < m$ and, in fact, k as small as possible in order to achieve parsimony: The n objects described by m attributes via C may then be described by k factors via A, with B representing a relationship between the original attributes and the factors. This relation can be interpreted in the following way: an object i has an attribute j if and only if there exists a factor l such that i has l (or, l applies to i) and j is one of the particular manifestations of l.

The product $\circ$ in (1) is a Boolean matrix product, defined by

$$(A \circ B)_{ij} = \bigvee_{l=1}^{k} A_{il} \cdot B_{lj}, \tag{2}$$

where $\bigvee$ denotes maximum (truth function of logical disjunction) and $\cdot$ is the usual product (truth function of logical conjunction). For example the following matrix can be decomposed into two Boolean matrices with $k < m$.

$$\begin{pmatrix} 1&1&0 \\ 1&1&1 \\ 1&0&1 \end{pmatrix} = \begin{pmatrix} 0&1 \\ 1&1 \\ 1&0 \end{pmatrix} \circ \begin{pmatrix} 1&0&1 \\ 1&1&0 \end{pmatrix}$$

The least k for which an exact decomposition $C = A \circ B$ exists is in the Boolean matrix theory called the Boolean rank (or Schein rank).

An optimal decomposition of the Boolean matrix can be found via Formal concept analysis. In this approach, the factors are represented by formal concepts, see [2]. The aim is to decompose the matrix C into a product $A_{\mathcal{F}} \circ B_{\mathcal{F}}$ of

Boolean matrices constructed from a set $\mathcal{F}$ of formal concepts associated to C. Let

$$\mathcal{F} = \{\langle A_1, B_1\rangle, \ldots, \langle A_k, B_k\rangle\} \subseteq \mathcal{B}(X, Y, C),$$

where $\mathcal{B}(X, Y, C)$ represents set of all formal concepts of context $\langle X, Y, C\rangle$. Denote by $A_{\mathcal{F}}$ and $B_{\mathcal{F}}$ the $n \times k$ and $k \times m$ binary matrices defined by

$$(A_{\mathcal{F}})_{il} = \begin{cases} 1 \text{ if } i \in A_l \\ 0 \text{ if } i \notin A_l \end{cases} \quad (B_{\mathcal{F}})_{lj} = \begin{cases} 1 \text{ if } j \in B_l \\ 0 \text{ if } j \notin B_l \end{cases}$$

for $l = 1, \ldots, k$. In other words, $A_{\mathcal{F}}$ is composed from characteristic vectors A_l. Similarly for $B_{\mathcal{F}}$. The set of factors is a set $\mathcal{F}$ of formal concepts of $\langle X, Y, C\rangle$, for which holds $C = A_{\mathcal{F}} \circ B_{\mathcal{F}}$. For every C such a set always exists. For details see [2].

Interpretation factors as a formal concepts is very convenient for users and we follow this point of view in our work. Because a factor can be seen as a formal concept, we can consider the intent part (denoted by $intent(F)$) and the extent part (denoted by $extent(F)$) of the factor F.

3 Related work

The Boolean matrix factorization (or decomposition), also known as the Boolean factor analysis, has gained interest in the data mining community during the past few years.

In the literature, we can find a wide range of theoretical and application papers about the Boolean factor analysis. The overview of the Boolean matrix theory can be found in [8]. A good overview from the BMF viewpoint is in e.g. [12]. For our work is the most important [2], where were first used formal concepts as factors.

Several heuristic algorithms for the BMF were proposed. In our work we adopt algorithm GreConD [2] (originally called Algorithm 2), but there exist several different approaches, which use so-called "tiles" in Boolean data [4], hyper-rectangles [15] or which introduce some noise [12, 10] in Boolean data.

From wide range of applications papers let us mentioned only [13] and [14], where the BMF is used for solving the Role mining problem.

In the literature, there can be found several methods for the latent factor analysis of ordinal data and also of multi-relational data [9], but using these methods for Boolean data has proved to be inconvenient many times.

The BMF of multi-relational data is not directly mentioned in any previous work. Indirectly, it is mentioned, in a very specific form, in [11] as Joint Subspace Matrix Factorization, where there are two Boolean matrices, which both share the same rows (or columns). The main aim is to find a set of shared factors (factors common for both matrices) and a set of specific factors (factors which are either in first or second matrix, not in both). This can be viewed as particular, very limited setting of our work.

From our point of view are also relevant works [5, 7]. These introduce the Relational formal concept analysis (RCA), i.e. the Formal concept analysis on multi-relational data. Our approach is different from the RCA. In our approach, we extract factors from each data table and connect these factors into more general factors. In RCA, they iteratively merge data tables into one in the following way: in each step they computed all formal concepts of one data table and these concepts are used as additional attributes for the merged data table. After obtaining a final merged data table, all formal concepts are extracted. Let us mention that our approach delivers more informative results than a simple use of BMF on merged data table from RCA, moreover getting merged data table is computationally hard.

4 Boolean factor analysis of multi-relational data

In this section we describe our basic problem setting. We have two Boolean data tables C_1 and C_2, which are interconnected with relation $\mathcal{R}_{C_1C_2}$. This relation is over the objects of first data table C_1 and the attributes of second data table C_2, i.e. it is an objects-attributes relation. In general, we can also define an objects-objects relation or an attributes-attributes relation. Our goal is to find factors, which explain the original data and which take into account the relation $\mathcal{R}_{C_1C_2}$ between data tables.

Definition 1. *Relation factor (pair factor) on data tables* C_1 *and* C_2 *is a pair* $\left\langle F_1^i, F_2^j \right\rangle$, *where* $F_i^1 \in \mathcal{F}_1$ *and* $F_2^j \in \mathcal{F}_2$ *(*$\mathcal{F}_i$ *denotes set of factors of data table* C_i*) and satisfying relation* $\mathcal{R}_{C_1C_2}$.

There are several ways how to define the meaning of "satisfying relation" from Definition 1. We will define the following three approaches (this definition holds for an object-attribute relation, other types of relations can be defined in similar way):

- F_1^i and F_2^j form pair factor $\langle F_1^i, F_2^j \rangle$ if holds:

  $$\bigcap_{k \in extent(F_1^i)} \mathcal{R}_k \neq \emptyset \text{ and } \bigcap_{k \in extent(F_1^i)} \mathcal{R}_k \subseteq intent(F_2^j),$$

  where $\mathcal{R}_k$ is a set of attributes, which are in relation with an object k. This approach we called *narrow* (it is analogy of the narrow operator in [7]).
- F_1^i and F_2^j form pair factor $\langle F_1^i, F_2^j \rangle$ if holds:

  $$\left(\left(\bigcap_{k \in extent(F_1^i)} \mathcal{R}_k \right) \cap intent(F_1^j) \right) \neq \emptyset.$$

  We called this approach *wide* (it is analogy of the wide operator in [7]).

- for any $\alpha \in [0,1]$, F_1^i and F_2^j form pair factor $\langle F_1^i, F_2^j \rangle$ if holds:

$$\frac{\left|\left(\bigcap_{k \in extent(F_1^i)} \mathcal{R}_k\right) \cap intent(F_2^j)\right|}{\left|\bigcap_{k \in extent(F_1^i)} \mathcal{R}_k\right|} \geq \alpha.$$

We called it an α-*approach.*

Remark 1. It is obvious, that for $\alpha = 0$ and replacing $\geq$ by $>$, we get the wide approach and for $\alpha = 1$, we get the narrow one.

Lemma 1. *For* $\alpha_1 > \alpha_2$ *holds, that a set of relation factors counted by* α_1 *is a subset of a set of relation factors obtained with* α_2*.*

We demonstrate our approach to factorisation of mutli-relational Boolean data by a small illustrative example.

Example 1. Let us have two data tables C_W (Table 1) and C_M (Table 2). C_W represents women and their characteristics and C_M represents men and their characteristics.

Table 1: C_W

	athlete	*undergraduate*	*wants kids*	*is attractive*
Abby		×	×	×
Becky	×		×	
Claire		×		×
Daphne	×	×	×	×

Table 2: C_M

	athlete	*undergraduate*	*wants kids*	*is attractive*
Adam	×			×
Ben		×	×	
Carl	×	×	×	
Dave			×	×

Table 3: $\mathcal{R}_{C_W C_M}$

	athlete	*undergraduate*	*wants kids*	*is attractive*
Abby		×	×	
Becky	×		×	
Claire	×	×		×
Daphne	×	×	×	×

Moreover, we consider relation $\mathcal{R}_{C_W C_M}$ (Table 3) between the objects of first the data table and the attributes of the second data table. In this case, it could be a relation with meaning "woman looking for a man with the characteristics".

Remark 2. Generally, nothing precludes the object-object relation (whose meaning might be "woman likes a man") and the attribute-attribute relation (whose meaning might be "the characteristics of women are compatible with the characteristics of men in the second data table").

Factors of data table C_W are:

- $F_1^W = \langle\{$Abby, Daphne$\}, \{$*undergraduate, wants kids, is attractive*$\}\rangle$
- $F_2^W = \langle\{$Becky, Daphne$\}, \{$*athlete, wants kids*$\}\rangle$
- $F_3^W = \langle\{$Abby, Claire, Daphne$\}, \{$*undergraduate, is attractive*$\}\rangle$

Factors of data table C_M are:

- $F_1^M = \langle\{\text{Ben, Carl}\}, \{\textit{undergraduate, wants kids}\}\rangle$
- $F_2^M = \langle\{\text{Adam}\}, \{\textit{athlete, is attractive}\}\rangle$
- $F_3^M = \langle\{\text{Adam, Carl}\}, \{\textit{athlete}\}\rangle$
- $F_4^M = \langle\{\text{Dave}\}, \{\textit{wants kids, is attractive}\}\rangle$

These factors were obtained via GreConD algorithm from [2]. We have two sets of factors (formal concepts), first set $\mathcal{F}_W = \{F_W^1, F_W^2, F_W^3\}$ factorising data table C_W and $\mathcal{F}_M = \{F_M^1, F_M^2, F_M^3\}$ factorising data table C_M.

Now we use so far unused relation $\mathcal{R}_{C_W C_M}$, between C_W and C_M to joint factors of C_W with factors of C_M into relational factors. For the above defined approaches we get results which are shown below. We write it as binary relations, i.e F_W^i and F_M^j belongs to relational factor $\langle F_W^i, F_M^j\rangle$ iff F_W^i and F_M^j are in relation:

Narrow approach

	F_M^1	F_M^2	F_M^3	F_M^4
F_W^1	×			
F_W^2				
F_W^3	×			

Wide approach

	F_M^1	F_M^2	F_M^3	F_M^4
F_W^1	×			×
F_W^2	×	×	×	×
F_W^3	×			

0.6-approach

	F_M^1	F_M^2	F_M^3	F_M^4
F_W^1	×			
F_W^2		×		
F_W^3	×			

0.5-approach

	F_M^1	F_M^2	F_M^3	F_M^4
F_W^1	×			×
F_W^2		×		
F_W^3	×			

The relational factor in form $\langle F_W^i, F_M^j\rangle$ can be interpreted in the following ways:

- Women, who belong to extent of F_W^i like men who belong to extent of F_M^j. Specifically in this example, we can interpret factor $\langle F_W^1, F_M^1\rangle$, that Abby and Daphne should like Ben and Carl.
- Women, who belong to extent of F_W^i like men with characteristic in intent of F_M^j. Specifically in this example, we can interpret factor $\langle F_W^1, F_M^1\rangle$, that Abby and Daphne should like undergraduate men, who want kids.
- Women, with characteristic from intent F_W^i like men who belong to extent F_M^j. Specifically in this example, we can interpret factor $\langle F_W^1, F_M^1\rangle$, that undergraduate, attractive women, who want kids should like Ben and Carl.
- Women, with characteristic from intent F_W^i like men with characteristic in intent of F_M^j. Specifically in this example, we can interpret factor $\langle F_W^1, F_M^1\rangle$, that undergraduate, attractive women, who want kids should like undergraduate men, who want kids.

Interpretation of the relation between F_W^i and F_M^j is driven by used approach. If we obtain factor $\langle F_W^i, F_M^j\rangle$ by narrow approach, we can interpret relation between F_W^i and F_M^j: “women who belong to F_W^i, like men from F_j^M completely”. For example factor $\langle F_W^1, F_M^1\rangle$ can be interpreted: “All undergraduate attractive women, who want kids, wants undergraduate men, who want kids.”

If we obtain factor $\langle F_W^i, F_M^j \rangle$ by wide approach, we can interpret the relation between F_W^i and F_M^j: "women who belong to F_W^i, like something about the men from F_j^M". For example $\langle F_W^2, F_M^1 \rangle$ can be interpreted: "All athlete woman, who want kids, like undergraduate men or man, who want kids."

If we get $\langle F_W^i, F_M^j \rangle$ by α-approach with value α, we interpret the relation between F_W^i and F_M^j as: "women from F_W^i, like men from F_j^M enough", where α determines measurement of tolerance.

Remark 3. Not all factors from data tables C_W or C_M must be present in any relational factor. It depends on the used relation. For example in Example 1 in narrow approach, the factors F_M^2, F_M^3, F_M^4 are not involved. In this case, we can add these simple factors to the set of relational factors and consider two types of factors. This factors are not pair factors, but classical factors from C_W or C_M. Of course this depends on a particular application.

Remark 4. For one factor F_1^i from the data table C_1, two factors from the data table C_2 (for example $F_2^{j_1}$ and $F_2^{j_2}$) can satisfy the relation. In this case we can add factor $\langle F_1^i, F_2^{j_1} \& F_2^{j_2} \rangle$, where $F_2^{j_1} \& F_2^{j_2}$ means

$$extent(F_2^{j_1} \& F_2^{j_2}) = extent(F_2^{j_1}) \cup extent(F_2^{j_2})$$

and

$$intent(F_2^{j_1} \& F_2^{j_2}) = intent(F_2^{j_1}) \cap intent(F_2^{j_2}),$$

instead of $\langle F_1^i, F_2^{j_1} \rangle$ and $\langle F_1^i, F_2^{j_2} \rangle$ to the relation factor set (in the case, that we consider an object-attribute relation). For example, by using 0.5-approach in Example 1, we get relational factors

$$\Big\langle \langle \{\text{Abby}, \text{Daphne}\}, \{\textit{undergraduate}, \textit{wants kids}, \textit{is attractive}\} \rangle, \\ \langle \{\text{Ben}, \text{Carl}\}, \{\textit{undergraduate}, \textit{wants kids}\} \rangle \Big\rangle$$

and

$$\Big\langle \langle \{\text{Abby}, \text{Daphne}\}, \{\textit{undergraduate}, \textit{wants kids}, \textit{is attractive}\} \rangle, \\ \langle \{\text{Dave}\}, \{\textit{wants kids}, \textit{is attractive}\} \rangle \Big\rangle.$$

This factors can be replaced with factor

$$\Big\langle \langle \{\text{Abby}, \text{Daphne}\}, \{\textit{undergraduate}, \textit{wants kids}, \textit{is attractive}\} \rangle, \\ \langle \{\text{Ben}, \text{Carl}, \text{Dave}\}, \{\textit{wants kids}\} \rangle \Big\rangle.$$

Remark 5. Another, simpler approach to multi-relational data factorization is such, that we do factorization of the relation $\mathcal{R}_{C_1C_2}$. This is correct because we can imagine the relation between data tables C_1 and C_2 as another data table. For each factor, we take the extent of this factor and compute concept in C_1, which contains this extent. Similarly for intents of factors and concepts in C_2. For example one of the factors of $\mathcal{R}_{C_WC_M}$ from Example 1 is:

$$\langle \{\text{Becky, Daphne}\}, \{\textit{athlete, wants kids}\} \rangle.$$

Relational factor computed from this factor will be

$$\langle\langle\{\text{Becky, Daphne}\},\{\textit{athlete, wants kids}\}\rangle,\\ \langle\{\text{Carl}\},\{\textit{athlete, undergraduate, wants kids}\}\rangle\rangle.$$

This approach seems to be better in terms of that we get pair of concepts for every factors, but we do not get an exact decomposition of data tables C_1 and C_2. Moreover this approach can not be extended to n-ary relations.

4.1 n-tuple relational factors, n-ary relations

Above approaches can be generalized for more than two data tables. In this generalization, we do not get factor pairs, but generally factor n-tuples. Now we extend Definition 1 to general definition of relational factor.

Definition 2. *Relation factor on data tables C_1, $C_2, \dots C_n$ is a n-tuple $\langle F_1^{i_1}, F_2^{i_2}, \dots F_n^{i_n}\rangle$, where $F_j^{i_j} \in \mathcal{F}_j$ where $j \in \{1, \dots, n\}$ ($\mathcal{F}_j$ denotes set of factors of data table C_j) and satisfying relations $\mathcal{R}_{C_l C_{l+1}}$ or $\mathcal{R}_{C_{l+1} C_l}$ for $l \in \{1, \dots, n-1\}$.*

We considered only binary relations between data tables, for which holds, that there exists only one relation interconnecting data tables C_i and C_{i+1} for $i \in \{1, \dots, n-1\}$. We left more general relations into the extended version of this paper. Let us mentioned, that this generalization of our approach is possible in the opposite of Remark 5. We show n-tuple relational factors on example.

Example 2. Let data table C_P (Table 4) represents people and their characteristic, C_R (Table 5) represents restaurants and their characteristics and C_C (Table 6) represents which ingredients are included in national cuisines.

Table 4: C_P

	European	*Asian*	*American*	*male*	*female*
Adam			×	×	
Ben	×			×	
Carol	×				×
Dale		×		×	
Emily					×
Frank				×	
Gabby		×			×

Table 5: C_R

	luxury	*ordinal*	*expensive*	*cheap*
Restaurant 1	×		×	
Restaurant 2	×		×	
Restaurant 3	×			×
Restaurant 4		×		×
Restaurant 5		×		×

Table 6: C_C

	vegetable 1	fruit 2	fish 3	sea food 4	legumes 5	mutton 6	lamb 7	olive 8	vine 9	herbs 10	cheese 11	mushroom 12	hot spice 13	rice 14	beef 15	pork 16	poultry 17	bamboo shoot 18	nut 19	lard 20	rabbit 21	venison 22	insides 23	corn 24	pasta/noodle 25	potato 26	pastry 27
Greek	×	×	×	×	×	×	×	×	×	×	×								×								
Chinese	×		×	×	×							×	×	×	×	×	×	×	×	×					×		
French	×		×	×		×	×		×	×	×	×			×	×	×				×	×	×				
Indian	×	×	×	×		×	×						×	×			×										
Czech	×	×			×					×	×	×			×	×	×			×	×	×	×			×	
Spanish	×	×	×	×				×	×	×				×	×	×	×										
Mexican	×	×	×	×	×								×	×	×	×	×							×			
Italian	×	×	×	×				×	×	×	×	×		×											×		×
American	×		×	×							×				×	×	×									×	×
Japanese	×		×	×										×													
German	×	×	×									×			×	×	×					×					

Table 7: $R_{C_P C_C}$

	vegetable 1	fruit 2	fish 3	sea food 4	legumes 5	mutton 6	lamb 7	olive 8	vine 9	herbs 10	cheese 11	mushroom 12	hot spice 13	rice 14	beef 15	pork 16	poultry 17	bamboo shoot 18	nut 19	lard 20	rabbit 21	venison 22	insides 23	corn 24	pasta/noodle 25	potato 26	pastry 27
Adam				×							×		×		×					×						×	
Ben												×			×	×	×			×	×	×					
Carol	×	×						×	×	×	×			×			×	×						×			
Dale			×	×										×			×		×		×	×			×		×
Emily			×				×		×			×					×						×	×			
Frank					×	×							×		×	×											
Gabby	×							×			×			×			×										

Relation $\mathcal{R}_{C_P C_C}$ (Table 7) represents relationship "person likes ingredients" and relation $\mathcal{R}_{C_R C_C}$ (Table 8) represents relationship "restaurant cooks national cuisine". In Tables 9, 10, 11, we can see factors of data tables C_P, C_R and C_C, respectively.

Table 8: $\mathcal{R}_{C_R C_C}$

	Greek	*Chinese*	*French*	*Indian*	*Czech*	*Spanish*	*Mexican*	*Italian*	*American*	*Japanese*	*German*
Restaurant 1	×		×		×	×		×			
Restaurant 2	×	×		×			×	×		×	
Restaurant 3					×			×	×		×
Restaurant 4						×	×	×	×		
Restaurant 5		×		×						×	×

Table 9: Factors of data table $\mathcal{C}_P$

F_P^i	*Extent*	*Intent*
F_P^1	{Adam, Ben, Dale, Frank}	{male}
F_P^2	{Adam, Emily, Frank}	{American}
F_P^3	{Carol, Emily, Gabby}	{female}
F_P^4	{Ben, Carol}	{European}
F_P^5	{Dale, Gabby}	{Asian}

Table 10: Factors of data table $\mathcal{C}_R$

F_R^i	*Extent*	*Intent*
F_R^1	{Restaurant 4, Restaurant 5}	{ordinal, cheap}
F_R^2	{Restaurant 1, Restaurant 2}	{luxury, expensive}
F_R^3	{Restaurant 3}	{luxury, cheap}

Table 11: Factors of data table $\mathcal{C}_C$

F_C^i	*Extent*	*Intent*
F_C^1	{Chinese, French, Spanish, Mexican, American, German}	{1, 3, 15, 16, 17}
F_C^2	{Greek, Spanish, Italian}	{1, 2, 3, 4, 8, 9, 10}
F_C^3	{French, Czech}	{1, 10, 11, 12, 15, 16, 17, 21, 22, 23}
F_C^4	{Chinese, Indian, Spanish, Mexican, Italian, Japanese}	{1, 3, 4, 14}
F_C^5	{Greek, French, Indian}	{1, 3, 4, 6, 7}
F_C^6	{Chinese}	{1, 3, 4, 5, 12, 13, 14, 15, 16, 17, 18, 19, 20, 25}
F_C^7	{Italian, American}	{1, 3, 4, 11, 27}
F_C^8	{Greek, Czech, Mexican}	{1, 2, 5}
F_C^9	{Indian, Mexican}	{1, 2, 3, 4, 13, 14, 17}
F_C^{10}	{Czech, Itelian, German}	{1, 2, 12}
F_C^{11}	{Czech, , American}	{1, 15, 16, 17, 26}
F_C^{12}	{Greek}	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 19}
F_C^{13}	{Greek, French, Spanish, Italian}	{1, 3, 4, 9, 10}
F_C^{14}	{Chinese, Czech}	{1, 5, 12, 15, 16, 17, 20}
F_C^{15}	{French, Czech, German}	{1, 12, 15, 16, 17, 22}
F_C^{16}	{Mexican}	{1, 2, 3, 4, 5, 13, 14, 15, 16, 17, 24}
F_C^{17}	{Chinese, Itelian}	{1, 3, 4, 12, 14, 25}

One of the relational factors, which we get by 0.5-approach, is $\langle F_P^1, F_C^{11}, F_R^3 \rangle$ and could be interpreted as "men would enjoy eating in luxury restaurants where the meals are cheap". Another factor is $\langle F_P^3, F_C^2, F_R^1 \rangle$ and could be interpreted as "women enjoy eating in ordinal cheap restaurants".

4.2 Representation of connection between factors

We can represent the relational factors via graph (n-partite). See Figure 1, which presents the results from the previous example. Each group of nodes (F_P^i, F_C^i, F_R^i) represents factors of a specific data table. Between two nodes, there is an edge iff factors representing nodes satisfy the input relation. Relational factor is path between nodes, which include at most one node from each group. For example, $\langle F_P^2, F_C^3, F_R^1 \rangle$ is a relational factor because there is an edge between nodes F_P^2 and F_C^3 and between F_C^3 and F_R^1.

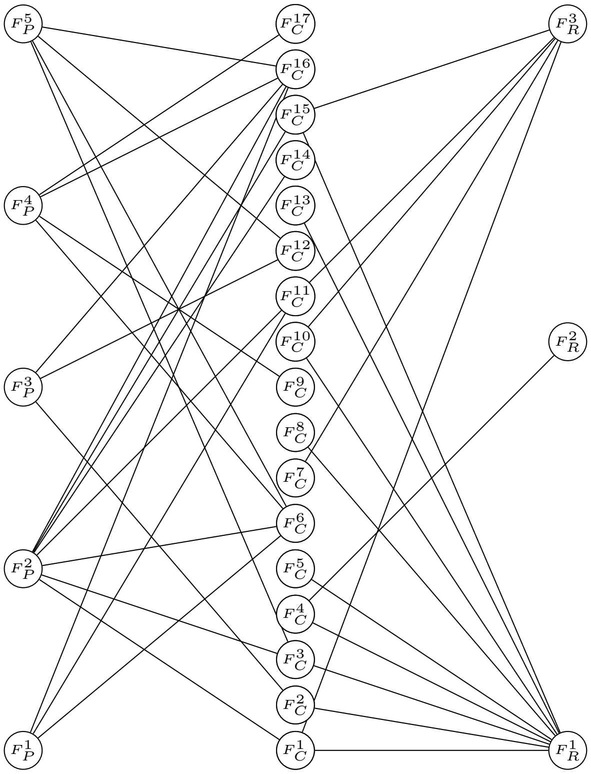


Fig. 1: Representation factors connections via graph.

5 Conclusion and Future Research

In this paper we present the new approach to BMF of multi-relational data, i.e. data which are composed from many data tables and relations between them. This approach, as opposed from to BMF, takes into account the relations and uses these relations to connect factors from individual data tables into one complex factor, which delivers more information than the simple factors.

A future research shall include the following topics: generalization multi-relational Boolean factorization for ordinal data, especially data over residuated lattices. Design an effective algorithm for computing relational factors. Develop new approaches for connecting factors which utilize statistical methods and last but not least drive factor selection in the second data table, using information about factors in the first one and relation between them, for obtaining more relevant data.

Acknowledgment We acknowledge support by the Operational Program Education for Competitiveness Project No. CZ.1.07/2.3.00/20.0060 co-financed by the European Social Fund and Czech Ministry of Education.

References

1. Bartholomew D. J., Knott M.: Latent Variable Models and Factor Analysis, 2nd Ed., London, Arnold, 1999.
2. Belohlavek R., Vychodil V.: Discovery of optimal factors in binary data via a novel method of matrix decomposition. J. Comput. Syst. Sci. 76(1):3–20, 2010.
3. Ganter B., Wille R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin, 1999.
4. Geerts F., Goethals B., Mielikäinen T.: Tiling databases, Proc. Discovery Science 2004, pp. 278–289.
5. Hacene M. R., Huchard M., Napoli A., Valtechev P.: Relational concept analysis: mining concept lattices from multi-relational data. Ann. Math. Artif. Intell. 67(1)(2013), 81–108,.
6. Harman H. H.: Modern Factor Analysis, 2nd Ed. The Univ. Chicago Press, Chicago, 1970.
7. Huchard M., Napoli A., Rouane H. M., Valtchev P.: A proposal for combining formal concept analysis and description logics for mining relational data. ICFCA 2007.
8. Kim K.H.: Boolean Matrix Theory and Applications. Marcel Dekker, New York, 1982.
9. Lippert, C., Weber, S. H., Huang, Y., Tresp, V., Schubert, M., and Kriegel, H.-P.: Relation-prediction in multi- relational domains using matrix-factorization. In NIPS 2008 Workshop on Structured Input - Structured Output, NIPS, 2008.
10. Lucchese C., Orlando S., Perego R.: Mining top-K patterns from binary datasets in presence of noise, SIAM DM 2010, pp. 165–176.
11. Miettinen P.: On Finding Joint Subspace Boolean Matrix Factorizations. Proc. SIAM International Conference on Data Mining (SDM2012), pp. 954-965, 2012.
12. 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.
13. Nau D.S., Markowsky G., Woodbury M.A., Amos D.B.: A mathematical analysis of human leukocyte antigen serology. Math Bioscience 40(1978), 243–270.
14. Vaidya J., Atluri V., Guo Q.: The role mining problem: finding a minimal descriptive set of roles. In: Proc. SACMAT 2007, pp. 175–184, 2007.
15. Xiang Y., Jin R., Fuhry D., Dragan F. F.: Summarizing transactional databases with overlapped hyperrectangles, Data Mining and Knowledge Discovery 23(2011), 215–251.

On Projections of Sequential Pattern Structures (with an application on care trajectories)

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

[1] LORIA (CNRS – Inria NGE – U. de Lorraine), Vandœuvre-lès-Nancy, France
[2] National Research University Higher School of Economics, Moscow, Russia
{aleksey.buzmakov, chedy.raissi}@inria.fr,
{elias.egho, nicolas.jay, amedeo.napoli}@loria.fr, skuznetsov@hse.ru

Abstract. In this paper, we are interested in the analysis of sequential data and we propose an original framework based on FCA. For that, we introduce sequential pattern structures, an original specification of pattern structures for dealing with sequential data. Sequential pattern structures are given by a subsumption operation between set of sequences, based on subsequence matching. To avoid a huge number of resulting concepts, domain knowledge projections can be applied. The original definition of projections is revised in order to operate on sequential pattern structures in a meaningful way. Based on the introduced definition, several projections of sequential pattern structures involving domain or expert knowledge are defined and discussed. This projections are evaluated on a real dataset on care trajectories where every hospitalization is described by a heterogeneous tuple with different fields. The evaluation reveals interesting concepts and justify the usage of introduced projections of sequential pattern structures. This research work provides a new and efficient extension of FCA to deal with complex data, which can be an alternative to the analysis of sequential datasets.

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

Introduction

Analysis of sequential data is a challenging task. In the last two decades, the main emphasis has been on developing efficient mining algorithms with effective pattern representations for sequences of itemsets [1–4]. The traditional sequential pattern mining algorithms generate a large number of frequent sequences while a few of them are truly relevant. Moreover, in some particular cases, only sequential patterns of a certain type are of interest and should be mined first. *Are we able to develop a framework for taking into account only patterns of required types?* Furthermore, in many cases sequential data are described by sequences with complex elements, e.g. a text is a sequence of syntactic trees. To process such kind of data with existing algorithms, elements of sequences can be scaled

Published in Manuel Ojeda-Aciego, Jan Outrata (Eds.): CLA 2013, pp. 199–210, ISBN 978–2–7466–6566–8, Laboratory L3i, University of La Rochelle, 2013.

into itemsets as it is done in the case of multilevel multidimensional data [5]. However, in this case it is rather difficult to introduce expert requirements within a sequence, which leads to even a larger set of resulting patterns.

We approach this problem with FCA and pattern structures [6, 7]. FCA is successfully used for analysis of sequential data [8, 9]. Moreover, it allows one to use different measures of interestingness for the resulting patterns (concepts). Pattern structures allows to directly process sequential data without a scaling step. Furthermore, there are projections of pattern structures, which were introduced in order to simplify the computation of pattern lattices, by simplifying descriptions. Moreover, projections can be efficiently used as special domain knowledge requirements, allowing to reduce the number of irrelevant patterns. We generalize the original definitions of projections, in order to deal with projections respecting domain knowledge. For example, sequences of length 1 are rare useful but they cannot be excluded by the original definition of projections.

The rest of the paper is organized as follows. In Section 1 we remind FCA, pattern structures and measures of concept interestingness. Section 2 states the problem of complex sequences analysis and introduces sequential pattern structures. In Section 3, first, the generalization of projections is defined, and, second, some projections specific to sequential pattern structures are introduced and analyzed. And finally before concluding the paper, we discuss an experimental evaluation in Section 4.

1 FCA and Pattern Structures

FCA [6] is a mathematical formalism having many applications in data analysis. Pattern structures is a generalization of FCA for dealing with complex structures, such as sequences or graphs [7].

Definition 1. *A pattern structure is a triple* $(G, (D, \sqcap), \delta)$*, where* G *is a set of objects,* $(D, \sqcap)$ *is a complete meet-semilattice of descriptions and* $\delta : G \to D$ *maps an object to a description.*

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

The Galois connection for $(G, (D, \sqcap), \delta)$ is defined as follows:

$$A^{\diamond} := \bigsqcap_{g \in A} \delta(g), \qquad \text{for } A \subseteq G$$

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

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

Definition 2. *A pattern concept of a pattern structure* $(G, (D, \sqcap), \delta)$ *is a pair* (A, d) *where* $A \subseteq G$ *and* $d \in D$ *such that* $A^\diamond = d$ *and* $d^\diamond = A$, A *is called the concept extent and* d *is called the concept intent.*

As in standard FCA, a pattern concept corresponds to the maximal set of objects A whose description subsumes the description d, where d is the maximal common description for objects in A. The set of all concepts can be partially ordered w.r.t. partial order on extents (dually, intent patterns, i.e $\sqsubseteq$), within a concept lattice. An example of a pattern structure is given and described in the next sections. It can be noticed that Table 1 defines a pattern structure, while the corresponding lattice is depicted in Figure 1.

It is worth mentioning, that the size of the concept lattice can be exponential w.r.t. to the number of objects, and, thus, we need a special ranking method to select the most interesting concepts for further analysis. Several techniques are considered in [10], where it is shown that stability index [11] is more reliable in noisy data. Thus, we use this index in our current work.

Definition 3. *Given a concept* c*, the concept stability* $Stab(c)$ *is the percent of subsets of the concept extent (denoted* $Ext(c)$*), whose description is equal to the concept intent (denoted* $Int(c)$*).*

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

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

After a brief general description of the analysis with pattern structures, the analysis of sequential data can be specified.

2 Sequential Pattern Structures

2.1 An Example of Sequential Data

Imagine that we have medical trajectories of patients, i.e. sequences of hospitalizations, where every hospitalization is described by a hospital name and a set of procedures. An example of sequential data on medical trajectories with three patients is given in Table 1. There are a set of procedures $P = \{a, b, c, d\}$, a

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

Table 1: Toy sequential data on patient medical trajectories.

set of hospital names $T_H = \{H_1, H_2, H_3, H_4, CL, CH, *\}$, where hospital names are hierarchically organized (by the level of generality), H_1 and H_2 are central hospitals (CH) and H_3 and H_4 are clinics (CL), and $*$ denotes the root of this hierarchy. For the sake of simplicity, we use the $\sqcap$ operator in order to denote the least common ancestor in T_H, i.e. $H_1 \sqcap H_2 = CH$. Every hospitalization is described with one hospital name and may contain several procedures. The procedure order in each hospitalization is not important in our case. For example, the first hospitalization $[H_2, \{c, d\}]$ for the second patient (p^2) was in hospital H_2 and during this hospitalization the patient underwent procedures c and d. An important task is to find the "characteristic" sequences of procedures and associated hospitals in order to improve hospitalization planning, optimize clinical processes or detect anomalies. This sequences can be found by searching for the most stable concepts in the lattice corresponding to a pattern structure.

2.2 Partial Order on Complex Sequences

A sequence is constituted of elements from an alphabet. The classical subsequence matching task requires no special properties of the alphabet. Several generalizations of the classical case were made by introducing a subsequence relation based on itemset alphabet [8] or on multidimensional and multilevel alphabet [5], scaled to itemset alphabet as well. Both these alphabets are certain semilattices, and, thus, we generalize the previous cases, requiring for an alphabet to form a general semilattice $(E, \sqcap_E)$[1]. Thanks to pattern structure formalism we are able to process in a unified way all types of sequential datasets with poset-shaped alphabet. However, some sequential data can have connections between elements, e.g. [12], and, thus, cannot be immediately processed by our approach.

Definition 4. *A sequence is an ordered list of* $e \in (E, \sqcap_E)$, *such that* $e \neq \bot_E$.

Here, $\forall e \in E, \bot_E = \bot_E \sqcap_E e$. The bottom element is required by the lattice structure but provide us with no useful information (it matches to any other element), thus, it is excluded from the sequences. In the same way, in mining of sequences of itemsets the empty itemset cannot be a proper element [2].

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

---

[1] In this paper we consider two semilattices, the first one is related to the characters of the alphabet, $(E, \sqcap_E)$, and the second one is related to pattern structures, $(D, \sqcap)$.

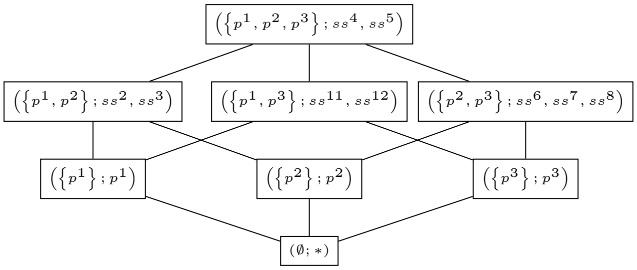

$(\{p^1, p^2, p^3\}; ss^4, ss^5)$
$(\{p^1, p^2\}; ss^2, ss^3)$
$(\{p^1, p^3\}; ss^{11}, ss^{12})$
$(\{p^2, p^3\}; ss^6, ss^7, ss^8)$
$(\{p^1\}; p^1)$
$(\{p^2\}; p^2)$
$(\{p^3\}; p^3)$
$(\emptyset; *)$

Fig. 1: The concept lattice for the pattern structure given by Table 1. Concept intents reference to sequences in Tables 1 and 2.

	Subsequences		Subsequences
ss^1	$\langle[CH, \{c, d\}]; [H_1, \{b\}]; [*, \{d\}]\rangle$	ss^2	$\langle[CH, \{c, d\}]; [*, \{b\}]; [*, \{d\}]\rangle$
ss^3	$\langle[CH, \{\}]; [*, \{d\}]; [*, \{a\}]\rangle$	ss^4	$\langle[*, \{c, d\}]; [*, \{b\}]\rangle$
ss^5	$\langle[*, \{a\}]\rangle$	ss^6	$\langle[*, \{c, d\}]; [CL, \{b\}]; [CL, \{a\}]\rangle$
ss^7	$\langle[CL, \{d\}]; [CL, \{\}]\rangle$	ss^8	$\langle[CL, \{\}]; [CL, \{a, d\}]\rangle$
ss^9	$\langle[CH, \{c, d\}]\rangle$	ss^{10}	$\langle[CL, \{b\}]; [CL, \{a\}]\rangle$
ss^{11}	$\langle[*, \{c, d\}]; [*, \{b\}]\rangle$	ss^{12}	$\langle[*, \{a\}]; [*, \{d\}]\rangle$

Table 2: Subsequences of patient sequences in Table 1.

With complex sequences and such kind of subsequences the computational procedure can be difficult, thus, to simplify the procedure, only "contiguous" subsequences are considered, where only the order of consequent elements is taken into account, i.e. given j_1 in Definition 5, $j_i = j_{i-1}+1$ for all $i \in \{2, 3, ..., k\}$. Such a restriction makes sens for our data, because a hospitalization is a discrete event and it is likely that the next hospitalization has a relation with the previous one, for example, hospitalizations for treating aftereffects of chemotherapy. Below the word "subsequence" refers to "contiguous" subsequence.

Example 1. In the running example (Section 2.1), the alphabet is $E = T_H \times \wp(P)$ with the similarity operation $(h_1, P_1) \sqcap (h_2, P_2) = (h_1 \sqcap h_2, P_1 \cap P_2)$, where $h_1, h_2 \in T_H$ are hospitals and $P_1, P_2 \in \wp(P)$ are sets of procedures. Thus, the sequence ss^1 in Table 2 is a subsequence of p^1 in Table 1 because if we set $j_i = i + 1$ (Definition 5) then $ss^1_1 \sqsubseteq p^1_{j_1}$ ('CH' is an ancestor for H_1 and $\{c, d\} \subseteq \{c, d\}$), $ss^1_2 \sqsubseteq p^1_{j_2}$ (the same hospital and $\{b\} \subseteq \{b, a\}$) and $ss^1_3 \sqsubseteq p^1_{j_3}$ ('*' is an ancestor for anything and $\{d\} \subseteq \{d\}$).

2.3 Meet-semilattice of Sequences

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

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

The set of all possible subsequences for a given sequence can be rather large. Thus, it is more efficient and readable to keep a pattern $d \in D$ as a set of only maximal sequences $\tilde{d}$, $\tilde{d} = \{s \in d \mid \nexists s^* \in d : s^* \geq s\}$. In the rest of the paper, every pattern is given only by the set of its maximal sequences. For example, $\{p^2\} \sqcap \{p^3\} = \{ss^6, ss^7, ss^8\}$ (see Tables 1 and 2), i.e. $\{ss^6, ss^7, ss^8\}$ is the set of all maximal sequences specifying the intersection result of two sets of sequences $\{p^2\}$ and $\{p^3\}$, in the same way $\{ss^6, ss^7, ss^8\} \sqcap \{p^1\} = \{ss^4, ss^5\}$. Note that representing a pattern by the set of all maximal sequences allows for an efficient implementation of the intersection "$\sqcap$" of two patterns. The next proposition is follows from this subsection and Definition 5.

Proposition 1. *Given $(G, (D, \sqcap), \delta)$ and $x, y \in D$, $x \sqsubseteq y$ if and only if $\forall s^x \in x$ there is a sequence $s^y \in y$, such that $s^x \leq s^y$.*

3 Projections of Sequential Pattern Structures

Pattern structures can be hard to process due to the usually large number of concepts in the concept lattice and the complexity of the involved similarity operation (make the parallel with the graph isomorphism problem). Moreover, a given pattern structure can produce a lattice with a lot of patterns which are not interesting for an expert. *Can we save computational time by avoiding the construction of unnecessary patterns?* Projections of pattern structures "simplify" to some degree the computation and allow one to work with a reduced description. In fact, projections can be considered as constraints (or filters) on patterns respecting certain mathematical properties. These mathematical properties ensure that the projection of a lattice is a lattice where projected concepts have certain correspondence to original ones. Moreover, the stability measure of projected concepts never decreases w.r.t the corresponding concepts. We introduce projections on sequential patterns, revising them from [7]. An extended definition of projections w.r.t. the definition in [7] should be provided in order to deal with interesting projections for real-life sequential datasets.

Definition 6. *A projection $\psi : D \to D$ is an interior operator, i.e. it is (1) monotone ($x \sqsubseteq y \Rightarrow \psi(x) \sqsubseteq \psi(y)$), (2) contractive ($\psi(x) \sqsubseteq x$) and (3) idempotent ($\psi(\psi(x)) = \psi(x)$).*

Under a projection ψ, a pattern structure $(G,(D,\sqcap),\delta)$ becomes the projected pattern structure $\psi((G,(D,\sqcap),\delta)) = (G,(D_\psi,\sqcap_\psi),\psi\circ\delta)$, where $D_\psi = \psi(D) = \{d \in D \mid \exists d^* \in D : \psi(d^*) = d\}$ and $\forall x,y \in D, x \sqcap_\psi y := \psi(x \sqcap y)$. Note that in [7] $\psi((G,(D,\sqcap),\delta)) = (G,(D,\sqcap),\psi\circ\delta)$. Now we should show that $(D_\psi,\sqcap_\psi)$ is a semilattice.

Proposition 2. *Given a semilattice* $(D,\sqcap)$ *and a projection* ψ*, for all* $x,y \in D$ $\psi(x \sqcap y) = \psi(\psi(x) \sqcap y)$.

Proof. 1. $\psi(x) \sqsubseteq x$, thus, $x,y \sqsupseteq (x \sqcap y) \sqsupseteq (\psi(x) \sqcap y) \sqsupseteq \psi(\psi(x) \sqcap y)$
2. $x \sqsubseteq y \Rightarrow \psi(x) \sqsubseteq \psi(y)$, thus, $\psi(x \sqcap y) \sqsupseteq \psi(\psi(x) \sqcap y)$
3. $\psi(x \sqcap y) \sqcap \psi(x) \sqcap y \underset{\psi(x\sqcap y)\sqsubseteq\psi(x)}{=} \psi(x \sqcap y) \sqcap y \underset{\psi(x\sqcap y)\sqsubseteq y}{=} \psi(x \sqcap y)$,
   then $(\psi(x) \sqcap y) \sqsupseteq \psi(x \sqcap y)$ and $\psi(\psi(x) \sqcap y) \sqsupseteq \psi(\psi(x \sqcap y)) = \psi(x \sqcap y)$
4. From (2) and (3) follows that $\psi(x \sqcap y) = \psi(\psi(x) \sqcap y)$.

Corollary 1. $X_1 \sqcap_\psi X_2 \sqcap_\psi \cdots \sqcap_\psi X_N = \psi(X_1 \sqcap X_2 \sqcap \cdots \sqcap X_N)$

Corollary 2. *Given a semilattice* $(D,\sqcap)$ *and a projection* ψ*,* $(D_\psi,\sqcap_\psi)$ *is a semilattice, i.e.* $\sqcap_\psi$ *is commutative, associative and idempotent.*

The concepts of a pattern structure and a projected pattern structure are connected with the next proposition, following from Corollary 1:

Proposition 3. *An extent in* $\psi((G,(D,\sqcap),\delta))$ *is an extent in* $(G,(D,\sqcap),\delta)$*. An intent in* $\psi((G,(D,\sqcap),\delta))$ *is of the form* $\psi(d)$*, where d is the intent of the concept with the same extent.*

Moreover, preserving extents of some concepts, projections cannot decrease the stability of the projected concepts, i.e. if the projection preserves a stable concept, then its stability (Definition 3) can only increase.

Proposition 4. *Given a pattern structure* $(G,(D,\sqcap),\delta)$*, its concept* c *and a projected pattern structure* $(G,(D_\psi,\sqcap_\psi),\psi\circ\delta)$*, and the projected concept* $\tilde{c}$*, if the concept extents are equal* ($Ext(c) = Ext(\tilde{c})$) *then* $Stab(c) \leq Stab(\tilde{c})$.

Proof. Concepts c and $\tilde{c}$ have the same extent. Thus, according to Definition 3, in order to prove the proposition statement, it is enough to prove that for any subset $A \subseteq Ext(c)$, if $A^\diamond = Int(c)$ in the original pattern structure, then $A^\diamond = Int(\tilde{c})$ in the projected one. It can be proven from contrary.

Suppose that $\exists A \subset Ext(c)$ such that $A^\diamond = Int(c)$ in the original pattern structure and $A^\diamond \neq Int(\tilde{c})$ in the projected one. Then there is a descendant concept $\tilde{d}$ of $\tilde{c}$ in the projected pattern structure such that $A^\diamond = Int(\tilde{d})$ in the projected lattice. Then there is an original concept d for the projected concept $\tilde{d}$ with the same $Ext(d)$. Then $A^\diamond \sqsupseteq Int(d) \sqsupset Int(c)$ and, so, $A^\diamond$ cannot be equal to $Int(c)$ in the original lattice. Contradiction.

No we are going to present two projections of sequential pattern structures. The first projection comes from the following observation. In many cases it may be more interesting to analyze quite long subsequences rather than short one. This kind of projections is called *Minimal Length Projection* (MLP) and it depends on the minimal allowed length parameter l for the sequences in a pattern. The corresponding function ψ maps a pattern without short sequences to itself, and a sequence with short sequences to the pattern containing only long sequences, $\psi(d) = \{s \in d \mid length(s) > l\}$. Later, propositions 1 and 5 stay that MLP is coherent with Definition 6.

Example 3. If we prefer common subsequences of length ≥ 3, then between p^2 and p^3 in Table 1 there is only one maximal common subsequence, ss^6 in Table 2, while ss^7 and ss^8 are too short to be considered. Figure 2a shows the lattice corresponding the projected pattern structure (Table 1) with patterns of length more or equal to 3.

Proposition 5. *MLP is a monotone, contractive and idempotent function on the semilattice* $(D, \sqcap)$.

Proof. The contractivity and idempotentcy are quite clear from the definition. Remains the proof for monotonicity.

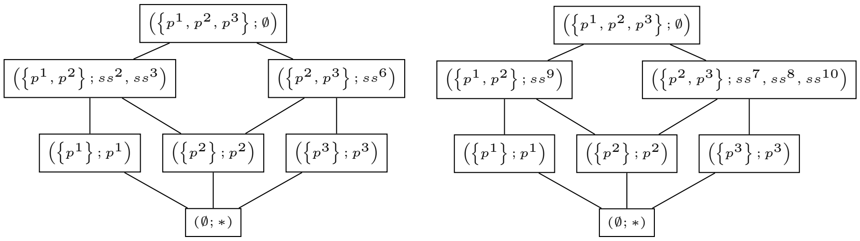
If $X \sqsubseteq Y$ where X and Y are sets of sequences then for every sequence $x \in X$ there is a sequence $y \in Y$ such that $x \leq y$ (Proposition 1). We should show that $\psi(X) \sqsubseteq \psi(Y)$, or in other words for every sequence $x \in \psi(X)$ there is a sequence $y \in \psi(Y)$, such that $x \leq y$. Given $x \in \psi(X)$, since $\psi(X)$ is a subset of X and $X \sqsubseteq Y$, then there is a sequence $y \in Y$ such that $x \leq y$, with $|y| \geq |x| \geq l$ (l is a parameter of MLP), and thus, $y \in \psi(Y)$.

The second projection of a sequential pattern structure is connected to a projection of an alphabet semilattice, $(E, \sqcap_E)$.

Example 4. An expert is interested in finding sequential patterns on how a patient changes hospitals, but he has little interest in procedures. Thus, any element of the alphabet lattice, containing a non-empty set of procedures can be projected to the element with the same hospital but with the empty set of procedures.

Example 5. An expert is interested in finding sequential patterns containing some information about the hospital in every hospitalization, and the corresponding procedures, i.e. hospital field in the patterns cannot be equal to the element "any hospital", denoted $*$, e.g., ss^5 is an invalid pattern, while ss^6 is a valid pattern in Table 2. Thus, any element of the alphabet semilattice with $*$ hospital can be projected to the $\perp_E$. Figure 2b shows the lattice corresponding to the projected pattern structure (Table 1), where projection comes from the projection of the alphabet semilattice.

Below we formally define how the alphabet projection of a sequential pattern structure should be processed. Intuitively, every sequence in a pattern should be substituted with another sequence, by applying the alphabet projection to all its elements. However, the result can be an incorrect sequence, because $\perp_E$ is forbidden to be in a sequence, thus, sequences in a pattern should be "developed" w.r.t. $\perp_E$, as it is explained below.



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

Fig. 2: The projected concept lattices for the pattern structure given by Table 1. Concept intents refer to the sequences in Tables 1 and 2.

Definition 7. *Given an alphabet* $(E, \sqcap_E)$*, an alphabet projection* ψ *and a sequence* s *based on* E*, the projection of the sequence* $\psi(s)$ *is the sequence* $\tilde{s}$ *such that,* $\tilde{s}_i = \psi(s_i)$ *(*s_i *is the* i*-th element of sequence* s*).*

Here, it should be noted that $\tilde{s}$ can be incoherent with Definition 4, since it allows $\bot_E$ to be an element. For simplicity, we allow this incoherence here.

Definition 8. *Given an alphabet* $(E, \sqcap_E)$*, an alphabet projection* ψ*, and a pattern* $d \in D$*, an alphabet-projected pattern* $\tilde{d} = \psi(d)$*, is the set of sequences obtained by the following procedure. For every sequence* $s \in d$*, the projection of* s *is computed (Definition 7) and, then, the projection of the sequence is substituted by the set of its maximal subsequences containing no* $\bot$*. All the resulting sequences constitute the set* $\hat{d}$*, and* $\tilde{d}$ *is the set of maximal sequences in* $\hat{d}$*.*

Example 6. $\{ss^6\}$ is an alphabet-projected pattern for the pattern $\{ss^{10}\}$, where alphabet lattice projection is given in Example 5.

$\{\langle [CH, \{c, d\}] \rangle\}$ is an alphabet-projected pattern for the pattern $\{ss^2\}$, where alphabet lattice projection is given by projecting every element with medical procedure b to the element with the same hospital and with the same set procedures excluding b. The projected sequence of sequence ss^2 is $\langle [CH, \{c, d\}]; [*, \{\}]; [*, \{d\}] \rangle$, but $[*, \{\}] = \bot_E$, and, thus, in order to project the pattern $\{ss^2\}$ the projected sequence is substituted by its maximal subsequences, i.e.
$\psi(\{\langle [CH, \{c, d\}]; [*, \{b\}]; [*, \{d\}] \rangle\}) = \{\langle [CH, \{c, d\}] \rangle\}$.

Proposition 6. *Considering an alphabet* $(E, \sqcap_E)$*, the projection of an alphabet* ψ*, a sequential pattern structure* $(G, (D, \sqcap), \delta)$*, the procedure given by Definition 8 is monotone, contractive and idempotent.*

Proof. This procedure is idempotent, since the projection of the alphabet is idempotent. It is contractive because for a pattern d, for any sequences $s \in d$, the projection of the sequence $\tilde{s} = \psi(s)$ is a subsequence of s. In the Definition 8 the projected sequences should be substituted by its maximal subsequences in order to avoid $\bot_E$, building the sets $\{\tilde{s}^i\}$. Thus, s is a supersequence for any $\tilde{s}^i$, and, so, the projected pattern $\tilde{d} = \psi(d)$ is subsumed by the pattern d.

Finally, we should show monotonicity. Given two patterns $x, y \in D$, such that $x \sqsubseteq y$, i.e. $\forall s^x \in x, \exists s^y \in y : s^x \leq s^y$, consider the projected sequence of s^x, $\psi(s^x)$. As $s^x \leq s^y$ for some s^y then for some $j_0 < j_1 < j_{|s^x|}$ (see Definition 5) $s^x_i \sqsubseteq_E s^y_{j_i}$ $(i \in 1, 2, ..., |s^x|)$, then $\psi(s^x_i) \sqsubseteq_E \psi(s^y_{j_i})$ (by the monotonicity of the alphabet projection), i.e. projected sequence preserve the subsequence relation. Thus, the alphabet projection of the pattern preserve pattern subsumption relation, $\psi(x) \leq \psi(y)$ (Proposition 1), i.e. the alphabet projection is monotone.

4 Sequential Pattern Structure Evaluation

4.1 Implementation

Nearly all state-of-the-art FCA algorithms can be adapted to process pattern structures. We adapted `AddIntent` algorithm [14], as the lattice structure is important for us to calculate stability (see the algorithm for calculating stability in [15]). To compute the semilattice operation ($\sqcap$, $\sqsubseteq$) between two sets of sequences $S = \{s^1, ...s^n\}$ and $T = \{t^1, ..., t^m\}$, $S \sqcap T$ is calculated according to Section 2.3, i.e. maximal sequences among all maximal common subsequences for any pair of s^i and t^j. To find all common subsequences of two sequences, the following observations is useful. If $ss = \langle ss_1; ...; ss_l \rangle$ is a subsequence of $s = \langle s_1; ...; s_n \rangle$ with $j^s_i = k^s + i$ (Definition 5: k^s is the index difference from which ss is a subsequence of s) and a subsequence of $t = \langle t_1; ...; t_m \rangle$ with $j^t_i = k^t + i$ (likewise), then for any index $i \in \{1, 2, ..., l\}$, $ss_i \sqsubseteq_E (s_{j^s_i} \sqcap t_{j^t_i})$. Thus, to find maximal common subsequences between s and t, we, first, align s and t in all possible ways, and then for every alignment we compute the resulting intersection and keep only the maximal ones.

4.2 Experiments and Discussion

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

The dataset considered here comes from a French healthcare system [16]. Each elements of a sequence has a "complex" nature. This dataset contains 2400 patients suffering from *cancer*. Every patient is described as a sequence of hospitalizations without any timestamps. The hospitalization is a tuple with three fields: (i) healthcare institution (e.g. University Hospital of Paris (CHU_{Paris})), (ii) reason of the hospitalization (e.g. a cancer disease), and (iii) set of medical procedures that the patient underwent. An example of a medical trajectory of a patient is $\langle [\mathrm{CHU}_{Paris}, \mathrm{Cancer}, \{P_1, P_2\}]; [\mathrm{CH}_{Nancy}, \mathrm{Chemo}, \{\}]; [\mathrm{CH}_{Nancy}, \mathrm{Chemo}, \{\}] \rangle$. This sequence represents a patient trajectory with three hospitalizations. It expresses that the patient was first admitted to the University Hospital of Paris (CHU_{Paris}) for a cancer problem as the reason, and underwent procedures P_1 and P_2. Then he had two consequent hospitalizations in Central Hospital of Nancy (CH_{Nancy}) in order to do chemotherapy with no additional procedures.

For this dataset the computation of the whole lattice is infeasible. However a medical expert is not interested in all possible patterns, but rather in patterns which answer his analysis question(s). First of all, the patterns of length 1 are unlikely to be of interest for him. Thus, we use the MLP projection of length 2 or 3 taking into account the small average length of the sequences in the dataset.

For the search of patterns containing only information about reasons and medical procedures, we should project every alphabet element on the element with the same reason and the same set of procedures, but substitute hospitalization institution by the most general element in the corresponding taxonomy. Moreover, we do not want to allow reason to be empty, i.e. all such elements should be projected onto $\perp_E$. In this case computation takes 18 seconds producing a lattice with around 34700 concepts. One of the stable concepts has the following intent $\langle[Cancer, \{App.\}]; [Ch.Prep, \{\}]; [Chemo, \{\}]\rangle$, specifying that a cancer was found during the appendix removal surgery, followed by a chemotherapy. This patterns highlight a discovered fact that acute appendicitis has been shown to occur antecedent to cancer within three years because of a carcinoma in colon or rectum [17].

To find patterns revealing dependences between hospitals and reasons all the procedures should be removed from each alphabet element and elements with most general hospital and/or with most general reason should be projected to $\perp_E$. The computation of the corresponding lattice takes 10 seconds, producing around 4200 concepts. $\langle[\mathit{Region\ Lorraine}, Cancer]; [\mathit{Clinic\ in\ Lorraine}, Chemo]\rangle$ is among stable concepts which is rather interesting, because the patients detected cancer somewhere in Region A but then went to exactly the same clinic for chemotherapy. It suggests that the department can lack from clinics for chemotherapy or the quality of the clinic is high.

Conclusion

In this paper, we present sequential pattern structures, an original specification of pattern structures able to deal with complex sequences. Projections of sequential pattern structures allow us to efficiently build concept lattices, by specifying expert demands. To be able to introduce interesting projections, their classical definition is extended. This extension allows us to introduce special projections for sequential pattern structures. The introduced projections are efficiently used for analysis of a dataset on care trajectories.

There are two main directions for future work. First, a study on properties of generalized projections within the overall framework of FCA should be carried out. Second, projections of sequential pattern structures can be deeper analyzed, for producing even more interesting and readable patterns.

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

References

1. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.C.: PrefixSpan Mining Sequential Patterns Efficiently by Prefix Projected Pattern Growth. In: 17th International Conference on Data Engineering. (2001) 215–226
2. Yan, X., Han, J., Afshar, R.: CloSpan: Mining Closed Sequential Patterns in Large Databases. In: Proc. of SIAM Int'l Conf. Data Mining (SDM'03). (2003) 166–177
3. Raïssi, C., Calders, T., Poncelet, P.: Mining conjunctive sequential patterns. Data Min. Knowl. Discov. **17**(1) (2008) 77–93
4. Ding, B., Lo, D., Han, J., Khoo, S.C.: Efficient Mining of Closed Repetitive Gapped Subsequences from a Sequence Database. In: Proc. of IEEE 25th International Conference on Data Engineering, IEEE (March 2009) 1024–1035
5. Plantevit, M., Laurent, A., Laurent, D., Teisseire, M., Choong, Y.W.: Mining multidimensional and multilevel sequential patterns. ACM Transactions on Knowledge Discovery from Data **4**(1) (January 2010) 1–37
6. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. 1st edn. Springer, Secaucus, NJ, USA (1997)
7. Ganter, B., Kuznetsov, S.O.: Pattern Structures and Their Projections. In Delugach, H., Stumme, G., eds.: Conceptual Structures: Broadening the Base SE - 10. Volume 2120 of LNCS. Springer Berlin Heidelberg (2001) 129–142
8. Casas-Garriga, G.: Summarizing Sequential Data with Closed Partial Orders. In: Proc. of the 5th SIAM Int'l Conf. on Data Mining (SDM'05). (2005)
9. Ferré, S.: The Efficient Computation of Complete and Concise Substring Scales with Suffix Trees. In Kuznetsov, S.O., Schmidt, S., eds.: Formal Concept Analysis SE - 7. Volume 4390 of Lecture Notes in Computer Science. Springer (2007) 98–113
10. Klimushkin, M., Obiedkov, S.A., Roth, C.: Approaches to the Selection of Relevant Concepts in the Case of Noisy Data. In: Proc. of the 8th International Conference on Formal Concept Analysis. ICFCA'10, Springer (2010) 255–266
11. Kuznetsov, S.O.: On stability of a formal concept. Annals of Mathematics and Artificial Intelligence **49**(1-4) (2007) 101–115
12. Adda, M., Valtchev, P., Missaoui, R., Djeraba, C.: A framework for mining meaningful usage patterns within a semantically enhanced web portal. In: Proceedings of the 3rd C* Conference on Computer Science and Software Engineering. C3S2E '10, New York, NY, USA, ACM (2010) 138–147
13. Kuznetsov, S.O.: Learning of Simple Conceptual Graphs from Positive and Negative Examples. In Żytkow, J., Rauch, J., eds.: Principles of Data Mining and Knowledge Discovery SE - 47. Volume 1704 of LNCS. Springer Berlin Heidelberg (1999) 384–391
14. Merwe, D.V.D., Obiedkov, S., Kourie, D.: AddIntent: A new incremental algorithm for constructing concept lattices. In Goos, G., Hartmanis, J., Leeuwen, J., Eklund, P., eds.: Concept Lattices. Volume 2961. Springer (2004) 372–385
15. Roth, C., Obiedkov, S., Kourie, D.G.: On succinct representation of knowledge community taxonomies with formal concept analysis A Formal Concept Analysis Approach in Applied Epistemology. International Journal of Foundations of Computer Science **19**(02) (April 2008) 383–404
16. Fetter, R.B., Shin, Y., Freeman, J.L., Averill, R.F., Thompson, J.D.: Case mix definition by diagnosis-related groups. Med Care **18**(2) (February 1980) 1–53
17. Arnbjörnsson, E.: Acute appendicitis as a sign of a colorectal carcinoma. Journal of Surgical Oncology **20**(1) (May 1982) 17–20

Spectral Lattices of reducible matrices over completed idempotent semifields

Francisco J. Valverde-Albacete[1] and Carmen Peláez-Moreno[2]*

[1] Departamento de Lenguajes y Sistemas Informáticos
Univ. Nacional de Educación a Distancia, c/ Juan del Rosal, 16. 28040 Madrid, Spain
`fva@lsi.uned.es`
[2] Departamento de Teoría de la Señal y de las Comunicaciones
Universidad Carlos III de Madrid, 28911 Leganés, Spain
`carmen@tsc.uc3m.es`

Abstract. Previous work has shown a relation between L-valued extensions of FCA and the spectra of some matrices related to L-valued contexts. We investigate the spectra of reducible matrices over completed idempotent semifields in the framework of dioids, naturally-ordered semirings, that encompass several of those extensions. Considering special sets of eigenvectors also brings out complete lattices in the picture and we argue that such structure may be more important than standard eigenspace structure for matrices over completed idempotent semifields.

1 Motivation

The eigenvectors and eigenspaces over certain naturally ordered semirings called *dioids* seem to be intimately related to multi-valued extensions of Formal Concept Analysis [1]. For instance [2, 3] prove that formal concepts are optimal factors for decomposing a matrix with entries in complete residuated semirings. Notice the strong analogy to the SVD, with formal concepts taking the role of pairs of left and right eigenvectors.

Indeed, [4] prove that, at least on a particular kind of dioids, the idempotent semifields, formal concepts are related to the eigenvectors of the unit in the semiring. This result, however, cannot be unified with the former both for theoretical reasons, since idempotent semifields are incomplete (see below), as well as for practical reasons, since the carrier set of idempotent semifields is almost never included in $[0, 1]$.

A possible way forward is to develop these theories under the common framework of the L-fuzzy sets, where L is any complete lattice [5]. Such an endeavour has already been called for [6], although it remains unfulfilled, hence this paper has a two-fold aim:

1. to clarify the spectral theory over *completed* idempotent semifields, and

---

* FJVA is supported by EU FP7 project LiMoSINe (contract 288024). CPM has been partially supported by the Spanish Government-Comisión Interministerial de Ciencia y Tecnología project TEC2011-26807 for this paper.

2. to take steps towards a common framework for the interpretation of *L-Formal Concept Analysis as a spectral construction.*

First steps have been taken in this direction with the development of a spectral theory of irreducible matrices [7] over complete idempotent semifields, whose summary we include below, but the general case, here presented, shows a more intimate relation to lattice theory.

1.1 Dioids and their spectral theory

A *semiring* is an algebra $\mathcal{S} = \langle S, \oplus, \otimes, \epsilon, e\rangle$ whose additive structure, $\langle S, \oplus, \epsilon\rangle$, is a commutative monoid and whose multiplicative structure, $\langle S\backslash\{\epsilon\}, \otimes, e\rangle$, is a monoid with multiplication distributing over addition from right and left and with additive neutral element absorbing for $\otimes$, i.e. $\forall a \in S,\ \epsilon \otimes a = \epsilon$.

Let $\mathcal{M}_n(\mathcal{S})$ be the semiring of square matrices over a semiring $\mathcal{S}$ with the usual operations. Given $A \in \mathcal{M}_n(\mathcal{S})$ the *right (left) eigenproblem* is the task of finding the *right eigenvectors* $v \in S^{n\times 1}$ and *right eigenvalues* $\rho \in S$ (respectively *left eigenvectors* $u \in S^{1\times n}$ and *left eigenvalues* $\lambda \in S$) satisfying:

$$u \otimes A = \lambda \otimes u \qquad\qquad A \otimes v = v \otimes \rho \tag{1}$$

The left and right eigenspaces and spectra are the sets of these solutions:

$$\begin{aligned} \Lambda(A) &= \{\lambda \in S \mid \mathcal{U}_\lambda(A) \neq \{\epsilon^n\}\} & \mathrm{P}(A) &= \{\rho \in S \mid \mathcal{V}_\rho(A) \neq \{\epsilon^n\}\} \\ \mathcal{U}_\lambda(A) &= \{u \in S^{1\times n} \mid u \otimes A = \lambda \otimes u\} & \mathcal{V}_\rho(A) &= \{v \in S^{n\times 1} \mid A \otimes v = v \otimes \rho\} \\ \mathcal{U}(A) &= \bigcup_{\lambda\in\Lambda(A)} \mathcal{U}_\lambda(A) & \mathcal{V}(A) &= \bigcup_{\rho\in\mathrm{P}(A)} \mathcal{V}_\rho(A) \end{aligned} \tag{2}$$

Since $\Lambda(A) = \mathrm{P}(A^{\mathrm{T}})$ and $\mathcal{U}_\lambda(A) = \mathcal{V}_\lambda(A^{\mathrm{T}})$, from now on we will omit references to left eigenvalues, eigenvectors and spectra, unless we want to emphasize differences.

With so little structure it might seem hard to solve (1), but a very generic solution based in the concept of transitive closure A^+ and transitive-reflexive closure A^* of a matrix is given by the following theorem:

Theorem 1 (Gondran and Minoux, [8, Theorem 1]). *Let $A \in \mathcal{S}^{n\times n}$. If A^* exists, the following two conditions are equivalent:*
1. *$A^+_{.i} \otimes \mu = A^*_{.i} \otimes \mu$ for some $i \in \{1 \dots n\}$, and $\mu \in S$.*
2. *$A^+_{.i} \otimes \mu$ (and $A^*_{.i} \otimes \mu$) is an eigenvector of A for e , $A^+_{.i} \otimes \mu \in \mathcal{V}_e(A)$.*

In [7] this result was made more specific in two directions: on the one hand, by focusing on particular types of completed idempotent semirings—semirings with a natural order where infinite additions of elements exist so transitive closures are guaranteed to exist and sets of generators can be found for the eigenspaces—and, on the other hand, by considering more easily visualizable subsemimodules than the whole eigenspace—a better choice for exploratory data analysis.

Specifically, every commutative semiring accepts a canonical preorder, $a \leq b$ if and only if there exists $c \in D$ with $a \oplus c = b$. A *dioid* is a semiring $\mathcal{D}$

where this relation is actually an order. Dioids are zerosumfree and entire, that is they have no non-null additive or multiplicative factors of zero. Commutative complete dioids are already complete residuated lattices. Similarly, semimodules over complete commutative dioids are also complete lattices.

An *idempotent semiring* is a dioid whose addition is idempotent, and a *selective semiring* one where the arguments attaining the value of the additive operation can be identified.

Example 1. Examples of idempotent dioids are

1. The *Boolean lattice* $\mathbb{B} = \langle \{0,1\}, \vee, \wedge, 0, 1 \rangle$
2. All fuzzy semirings, e.g. $\langle [0,1], \max, \min, 0, 1 \rangle$
3. The *min-plus algebra* $\mathbb{R}_{\min,+} = \langle \mathbb{R} \cup \{\infty\}, \min, +, \infty, 0 \rangle$
4. The *max-plus algebra* $\mathbb{R}_{\max,+} = \langle \mathbb{R} \cup \{ -\infty \}, \max, +, -\infty, 0 \rangle$ □

Of the semirings above, only the boolean lattice and the fuzzy semirings are complete dioids, since the rest lack the *top* element $\top$ as an adequate inverse for the bottom in the order.

The second important feature of spectra over complete dioids, as proven in [7], is that the set of eigenvalues on complete dioids is extended with respect to the incomplete case, and it makes sense to distinguish those which are associated to finite eigenvectors, the *proper eigenvalues* $\mathrm{P}^{\mathrm{P}}(A)$, and those associated with non-finite eigenvectors, the *improper eigenvalues* $\mathrm{P}(A) \backslash \mathrm{P}^{\mathrm{P}}(A)$.

Moreover, as said above, the eigenspaces of matrices over complete dioids have the structure of a complete lattice. But since these lattices may be continuous and difficult to represent we introduce the more easily-represented *eigenlattices* $\mathcal{L}_\rho(A)$ and $\mathcal{L}_\lambda(A)$, complete *finite* sublattices of the eigenspaces to be used as scaffolding in visual representations.

1.2 Completed idempotent semifields and their spectral theory

A semiring is a *semifield* if there exists a multiplicative inverse for every element $a \in S$, notated as a^{-1}, and *radicable* if the equation $a^b = c$ can be solved for a. As exemplified above, idempotent semifields are incomplete in their natural order. Luckily, there are procedures for *completing* such structures [9] and we will not differentiate between *complete or completed* structures,

Example 2. The maxplus $\mathbb{R}_{\max,+}$ and minplus $\mathbb{R}_{\min,+}$ semifields can be completed as,

1. the *completed Minplus semifield*, $\overline{\mathbb{R}}_{\min,+} = \langle \mathbb{R} \cup \{-\infty, \infty\}, \min, \dot{+}, -\cdot, \infty, 0 \rangle$.
2. the *completed Maxplus semifield*, $\overline{\mathbb{R}}_{\max,+} = \langle \mathbb{R} \cup \{-\infty, \infty\}, \max, \underset{\cdot}{+}, -\cdot, -\infty, 0 \rangle$,

In this notation we have $\forall c, -\infty \underset{\cdot}{+} c = -\infty$ and $\infty \dot{+} c = \infty$, which solves several issues in dealing with the separately completed dioids. These two completions are inverses $\overline{\mathbb{R}}_{\min,+} = \overline{\mathbb{R}}^{-1}_{\max,+}$, hence order-dual lattices. Indeed they are better jointly called the *max-min-plus* semiring $\overline{\mathbb{R}}^{\min,\dot{+}}_{\max,\underset{\cdot}{+}}$. □

In fact, idempotent semifields $\mathcal{K} = \langle K, \underset{\cdot}{\oplus}, \dot{\oplus}, \underset{\cdot}{\otimes}, \dot{\otimes}, \cdot^{-1}, \bot, e, \top \rangle$, appear as enriched structures, the advantage of working with them being that meets can be expressed by means of joins and inversion as $a \wedge b = (a^{-1} \underset{\cdot}{\oplus} b^{-1})^{-1}$. On a practical note, residuation in complete commutative idempotent semifields can be expressed in terms of inverses, and this extends to eigenspaces.

Given $A \in \mathcal{M}_n(\mathcal{S})$, the *network (weighted digraph) induced by* A, $N_A = (V_A, E_A, w_A)$, consists of a set of vertices $V_A = \bar{n}$, a set of arcs , $E_A = \{(i,j) \mid A_{ij} \neq \epsilon_S\}$, and a weight $w_A : V_A \times V_A \to S,\ (i,j) \mapsto w_A(i,j) = a_{ij}$. This allows us to apply intuitively all notions from networks to matrices and vice versa, like the underlying graph $G_A = (V_A, E_A)$, the set of paths $\Pi_A^+(i,j)$ between nodes i and j or the set of cycles C_A^+ . Matrix A is *irreducible* if every node of V_A is connected to every other node in V_A though a path, otherwise it is *reducible*.

We will use the spectra of irreducible matrices as a basic block for that of reducible matrices: if C_A^+ is the set of cycles of A then $\mu_\oplus(A) = \oplus\{\mu_\oplus(c) \mid c \in C_A^+\}$ is their *aggregate cycle mean*. For finite $\mu_\oplus(A)$, let $\tilde{A}^+ = (A \dot{\otimes} \mu_\oplus(A)^{-1})^+$ be the *normalized transitive closure of* A, and define the set of (right) *fundamental eigenvectors of irreducible* A *for* ρ as $\mathrm{FEV}_\rho(A) = \{\tilde{A}^+_{\cdot i} \mid \tilde{A}^+_{ii} = e\}$, with left fundamental eigenvectors $\mathrm{FEV}_\rho(A^{\mathrm{T}}) = \mathrm{FEV}_\rho(A)^{\mathrm{T}}$. Then,

Theorem 2 ((Right) spectral theory for irreducible matrices, [7]). *Let* $A \in \mathcal{M}_n(\overline{\mathcal{K}})$ *be an irreducible matrix over a complete commutative selective radicable semifield. Then:*

1. $\Lambda(A) = \overline{\mathcal{K}} \backslash \{\bot\} = \mathrm{P}(A)$.
2. $\Lambda^{\mathrm{P}}(A) = \{\mu_\oplus(A)\} = \mathrm{P}^{\mathrm{P}}(A)$.
3. *If* $\rho \in \mathrm{P}(A) \backslash \mathrm{P}^{\mathrm{P}}(A)$*, then* $\mathcal{V}_\rho(A) = \{\bot^n, \top^n\} = \mathcal{L}_\rho(A)$.
4. *If* $\rho = \mu_\oplus(A) < \top$*, then* $\mathcal{V}_\rho(A) = \langle \mathrm{FEV}_\rho(A) \rangle_{\overline{\mathcal{K}}} \supset \mathcal{L}_\rho(A) = \langle \mathrm{FEV}_\rho(A) \rangle_{\mathbb{3}}$.

In this paper we try and find analogue results to Theorem 2 for the reducible case. First, we completely describe the spectra with Theorem 3 in Section 3.1. Then, we provide in Section 3.2 a bottom-up construction of the eigenspaces of certain reducible matrices from that of their irreducible blocks, using from Section 2.2 a recursive scheme to render matrices over idempotent semifields into specialised Upper Frobenius Normal Forms (UFNF). Finally, we discuss our findings in Section 4 and relate them to previous approaches.

2 Preliminaries

2.1 Some partial orders and lattices

In this paper we assume familiarity with basic order notions as described in [1, 10]. We only introduce notation when departing from there.

Recall that every set V with $|V| = n$ elements and a total order $\leq \subseteq V \times V$ is isomorphic to a lattice called the *chain of* n *elements*, $\langle V, \leq \rangle \cong \mathbf{n}$. When the relation is the empty order relation $\varnothing \in V \times V$, it is called an *anti-chain of* n *elements*, $\langle V, \varnothing \rangle \cong \bar{\mathbf{n}}$. Lattice $\mathbb{1} \cong \mathbf{1}$ is the vacuously-ordered singleton. Lattice

$\mathbb{2}$ is the boolean lattice isomorphic to chain **2**. Lattice $\mathbb{3}$ is a lattice lying at the heart of completed semifields, the $\mathbb{3}$-bounded lattice-ordered group $\bot < e < \top$, isomorphic to chain **3**.

If set of order ideals of a poset P is $\mathcal{O}(P)$, then

Proposition 1 ([10, Chap. 1]). *Let* $\langle P, \leq \rangle$ *be a finite poset. Then* $\langle \mathcal{O}(P), \subseteq \rangle$ *is a lattice obtained by the embedding* $\varphi : P \to \mathcal{O}(P), \varphi(x) = \downarrow x$ *, with* $\forall A_1, A_2 \in \mathcal{O}(P), A_1 \vee A_2 = A_1 \cup A_2$ *and* $A_1 \wedge A_2 = A_1 \cap A_2$ *.*

Note that $x \leq y$ in $\mathcal{P}$ if and only if $\downarrow x \subseteq \downarrow y$ in $\mathcal{O}(P)$. Furthermore, $\mathcal{O}(P)$ can be obtained systematically from the ordered set in a number of cases:

Proposition 2 ([10, Chap. 1]). *Let* $\langle P, \leq \rangle$ *be a finite poset. Then*
1. $\mathcal{O}(P \oplus \mathbb{1}) \cong \mathcal{O}(P) \oplus \mathbb{1}$ *and* $\mathcal{O}(\mathbb{1} \oplus P) \cong \mathbb{1} \oplus \mathcal{O}(P)$ *.*
2. $\mathcal{O}(P_1 \uplus P_2) \cong \mathcal{O}(P_1) \times \mathcal{O}(P_2)$ *.*
3. $\mathcal{O}(P^{\mathrm{d}}) \cong \mathcal{F}(P) \cong \mathcal{O}(P)^{\mathrm{d}}$ *.*
4. $\mathcal{O}(n) \cong n \oplus \mathbb{1} \cong \mathbb{1} \oplus n$ *.*
5. $\mathcal{O}(\overline{n}) \cong \mathbb{2}^n$ *.*

2.2 An inductive structure for reducible matrices

Recall that a *digraph (or directed graph)*, is a pair $G = (V, E)$ with V a set of *vertices* and $E \subseteq V \times V$ a set of *arcs (directed edges)*, ordered pairs of vertices, such that for every $i, j \in V$ there is at most one arc $(i, j) \in E$. If $(i, j) \in E$ then we say that *"i is a predecessor of j"* or *"j is a successor of i"*, and $E \in \mathcal{M}_n(\mathbb{B})$ is the *associated relation of* G. If there is a walk from a vertex i to a vertex j in G we say that *"i has access to j"* or *j is reachable from i,* $i \rightsquigarrow j$. Hence, *reachability* is the transitive closure of the associated relation, $\rightsquigarrow = E^+$ [11]. However, vertices $j \in V$ with no incoming or outgoing arcs cannot be part of any cycle, hence $j \not\rightsquigarrow j$ for such nodes, so it is not reflexive, in general. $(\rightsquigarrow \cap I_V)$ is the *reflexive restriction* of $\rightsquigarrow$, that is, the biggest reflexive relation included in it.

If there is a walk from a vertex i to vertex j in G or viceversa we say that *i and j are connected,* $i \rightsquigarrow j \vee j \rightsquigarrow i$. *Connectivity* is the symmetric closure of the reachability relation: its transitive, reflexive restriction is an equivalence relation on V_G whose classes are called the *(dis)connected components of* G . Note that each of the (outwards) disconnected components is actually (inwards) connected. Let $K \geq 1$ be the number of disconnected components of G. Then V and E are partitioned into the subsets of vertices $\{V_k\}_{k=1}^K$ and arcs $\{E_k\}_{k=1}^K$ of each disconnected component $\bigcup_k V_k = V$, $V_k \cap V_l = \varnothing, k \neq l$, $\bigcup_k E_k = E$, $E_k \cap E_l = \varnothing, k \neq l$ and we may write $G = \uplus_{k=1}^K G_k$ is a disjoint union of graphs. G is called *connected* itself if $K = 1$.

On the other hand, since reachability is transitive by construction, its symmetric, reflexive restriction $i \leftrightsquigarrow j \Leftrightarrow i \rightsquigarrow j \wedge j \rightsquigarrow i$ is a refinement of connectivity called *strong connectivity.* Its equivalence classes are the *strongly connected components* of G . For each disconnected component G_k, let R_k be the number of its strongly connected components. If $R_k = 1$ then the k-th component is *strongly* *connected*, otherwise just *connected* and composed of a number of strongly connected components itself. G is *strongly connected* itself if $K = R = 1$.

Given a digraph $G = (V, E)$, the *reduced or condensation digraph,* $\overline{G} = (\overline{V}, \overline{E})$ is induced by the set $\overline{V} = V/ \leftrightsquigarrow$ of strongly connected components, and $C, C' \in \overline{V}$, $(C, C') \in \overline{E}$ iff there exists one arc $(i, j) \in E$ for some $i \in C, j \in C'$ and we say that component C *has access to* C', and write $C \preccurlyeq C'$. It is well known that $\overline{G} = (\overline{V}, \overline{E})$ is a partially ordered set called a *directed acyclic graph (dag).*

Given a matrix $A \in \mathcal{M}_n(\mathcal{S})$, its *condensation digraph* is the partial order of strong connectivity classes $\overline{G}_A = (\overline{V}_A, \overline{E}_A)$, as in Figure 2b in Example 4, of its associated digraph $G_A = (V_A, E_A)$. This can be proven by means of an Upper Frobenius Normal Form (UFNF) [12], a structure for matrices that we intend to specialise and use as a scheme for structural induction over reducible matrices.

In the following, for a set of indices $V_x \subseteq \mathbf{\overline{n}}$ we write $P(V_x)$ for a permutation of it, and $\mathcal{E}_{xy}$ is an empty matrix of conformal dimension most of the times represented on matrix patterns as elliptical dots.

Lemma 1 (Recursive Upper Frobenius Normal Form, UFNF). *Let* $A \in \mathcal{M}_n(S)$ *be a matrix over a semiring and* $\overline{G}_A$ *its condensation digraph. Then,*

1. *(UFNF$_3$) If A has zero lines it can be transformed by a simultaneous row and column permutation of V_A into the following form:*

$$P_3^{\mathrm{T}} \otimes A \otimes P_3 = \begin{bmatrix} \mathcal{E}_{\iota\iota} & \cdot & \cdot & \cdot \\ \cdot & \mathcal{E}_{\alpha\alpha} & A_{\alpha\beta} & A_{\alpha\omega} \\ \cdot & \cdot & A_{\beta\beta} & A_{\beta\omega} \\ \cdot & \cdot & \cdot & \mathcal{E}_{\omega\omega} \end{bmatrix} \tag{3}$$

   where: either $A_{\alpha\beta}$ or $A_{\alpha\omega}$ or both are non-zero, and either $A_{\alpha\omega}$ or $A_{\beta\omega}$ or both are non-zero. Furthermore, P_3 is obtained concatenating permutations for the indices of simultaneously zero columns and rows V_ι, the indices of zero columns but non-zero rows V_α, the indices of zero rows but non-zero columns V_ω and the rest V_β as $P_3 = P(V_\iota)P(V_\alpha)P(V_\beta)P(V_\omega)$.

2. *(UFNF$_2$) If A has no zero lines it can be transformed by a simultaneous row and column permutation $P_2 = P(A_1) \dots P(A_k)$ into block diagonal UFNF*

$$P_2^{\mathrm{T}} \otimes A \otimes P_2 = \begin{bmatrix} A_1 & \cdot & \dots & \cdot \\ \cdot & A_2 & \dots & \cdot \\ \vdots & \vdots & \ddots & \vdots \\ \cdot & \cdot & \dots & A_K \end{bmatrix} \tag{4}$$

   where $\{A_k\}_{k=1}^{K}, K \geq 1$ are the matrices of connected components of $\overline{G}_A$.

3. *(UFNF$_1$) If A is reducible with no zero lines and a single connected component it can be simultaneously row- and column-permuted by P_1 to*

$$P_1^{\mathrm{T}} \otimes A \otimes P_1 = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1R} \\ \cdot & A_{22} & \cdots & A_{2R} \\ \vdots & \vdots & \ddots & \vdots \\ \cdot & \cdot & \cdots & A_{RR} \end{bmatrix} \tag{5}$$

where A_{rr} are the matrices associated to each of its R strongly connected components (sorted in a topological ordering), and $P_1 = P(A_{11}) \dots P(A_{RR})$.

The particular choice of UFNF is clarified by the following Lemma, since the condensation digraph will prove important later on:

Lemma 2 (Scheme for structural induction over reducible matrices). *Let $A \in \mathcal{M}_n(S)$ be a matrix over an entire zerosumfree semiring and $\overline{G}_A$ its condensation digraph. Then:*

1. *If A is irreducible then $\overline{G}_A \cong \mathbb{1}$.*
2. *If A is in UFNF$_2$ then $\overline{G}_A = \biguplus \overline{G}_{A_k}$.*
3. *If A is in UFNF$_3$ then $\overline{G}_A = \overline{G}_{A_{\beta\beta}}$.*
4. *$\overline{G}_{A^{\mathrm{T}}} = (\overline{G}_A)^{\mathrm{d}}$.*

Note that for A in UFNF$_1$, $\overline{G}_A$ may be any connected ordered set.

3 Results

3.1 Generic results for reducible matrices

The following lemma clarifies the order relation between eigenvectors in ordered semimodules,

Lemma 3. *Let $\mathcal{X}$ be a naturally-ordered semimodule.*

1. *Vectors with incomparable supports are incomparable.*
2. *If $\mathcal{X}$ is further complete, vectors with incomparable saturated supports are incomparable.*

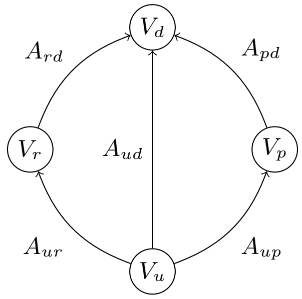
Proof. Let v and w be two vectors in $\mathcal{X}$. Comparability of supports lies in the $\subseteq$ relation: if $\mathrm{supp}(v) \parallel \mathrm{supp}(w)$ then $\mathrm{supp}(v) \nsubseteq \mathrm{supp}(w)$ and $\mathrm{supp}(w) \nsubseteq \mathrm{supp}(v)$. Therefore from $\mathrm{supp}(v) \cap \mathrm{supp}(w)^C \neq \varnothing$ we have $v(\mathrm{supp}(v) \cap \mathrm{supp}(w)^C) \neq \bot$ and $w(\mathrm{supp}(v) \cap \mathrm{supp}(w)^C) = \bot$, hence $v \nleq w$. Similarly, from $\mathrm{supp}(w) \cap \mathrm{supp}(v)^C \neq \varnothing$ we have $w \nleq v$, therefore $v \parallel w$. Claim 2 is likewise argued on the support taking the role of $\overline{\mathbf{n}}$, and the saturated support taking the role of the original support. □

Let $A \in \mathcal{M}_n(\mathcal{S})$ be a matrix and $\overline{G}_A$ be its condensation digraph. Consider a class $C_r \in \overline{V}_A$ and call $V_u = (\bigcup_{C' \in \downarrow C_r} C') \backslash C_r$, $V_d = (\bigcup_{C' \in \uparrow C_r} C') \backslash C_r$ and $V_p = V_A \backslash (V_u \cup C_r \cup V_d)$ the *sets of upstream, downstream and parallel vertices for C_r*, respectively. Using permutation $P_r = P(V_u)P(C_r)P(V_p)P(V_d)$ we may suppose a blocked form of $A(C_r)$ like the one in Fig. 1 without loss of generality. Notice that any of V_u, V_d or V_p may be empty. However, if V_u (resp. V_d) is not of null dimension, then A_{ur} (resp. A_{rd}) cannot be empty.

Proposition 3. *Let $A \in \mathcal{M}_n(\overline{\mathcal{K}})$ be a matrix over a complete commutative selective radicable semifield with $C_A^+ \neq \varnothing$. Then a scalar $\rho > \bot$ is a proper eigenvalue of A if and only if there is at least one class in its condensation digraph $C_r \in \overline{G}_A$ such that $\rho = \mu_\oplus(A_{rr})$.*

$$A(C_r) = \begin{bmatrix} A_{uu} & A_{ur} & A_{up} & A_{ud} \\ \cdot & A_{rr} & \cdot & A_{rd} \\ \cdot & \cdot & A_{pp} & A_{pd} \\ \cdot & \cdot & \cdot & A_{dd} \end{bmatrix}$$

(a) Blocked form of $A(C_R)$



(b) Digraph of $A(C_R)$

Fig. 1: Matrix A focused on C_r, $A(C_r) = {P_r}^{\mathrm{T}} \otimes A \otimes P_r$ and associated digraph. The loops at each node, consisting of (possibly empty) A_{uu}, A_{rr}, A_{pp}, A_{dd} are not shown.

Proof. The proof, for instance, in [8] extends even to $\rho = \top$,. □

The proper spectrum is clarified by:

Lemma 4. *Let $A \in \mathcal{M}_n(\mathcal{S})$ be a reducible matrix over a complete radicable selective semifield. Then, there are no other finite eigenvectors in $\mathcal{V}_\rho(A)$ contributed by $\tilde{A}^\rho$ than those selected by the critical circuits in $C_r \in \overline{V}_A$ such that $\mu_\oplus(A_{rr}) = \rho$,*

$$\mathrm{FEV}^{\mathrm{F}}(A) = \cup_{C_r \in \overline{V}_A} \{ (\tilde{A}^\rho)^{+}_{\cdot i} \mid i \in V_r^c,\ \mu_\oplus(A_{rr}) = \rho \} \ .$$

Proof. If $\rho = \mu_\oplus(A_{rr})$, from Proposition 3 we see that the finite eigenvectors mentioned really belong in $\mathcal{V}_\rho(A)$. If $\rho > \mu_\oplus(A_{rr})$ then $(\tilde{A}^\rho_{rr})^{+}_{ii} < e = (\tilde{A}^\rho_{rr})^{*}_{ii}$ hence the columns selected by C_r do not generate eigenvectors. If $\rho < \mu_\oplus(A_{rr})$ then $(\tilde{A}^\rho_{rr})^{+}_{ij} = \top$ and whether those classes with cycle mean ρ are upstream or downstream of C_r the only value that is propagated is $\top$, hence the eigenvectors are all saturated. □

Theorem 3 (Spectra of generic matrices). *Let $A \in \mathcal{M}_n(\overline{\mathcal{D}})$ be a reducible matrix over an entire zerosumfree semiring. Then,*

1. *If $C_A^+ = \varnothing$ then $\mathrm{P}(A) = \mathrm{P}^{\mathrm{P}}(A) = \{\epsilon\}$.*
2. *If $C_A^+ \neq \varnothing$ and further $\overline{\mathcal{D}}$ is a complete selective radicable semifield,*
   - *(a) If $\overline{\mathrm{zc}}(A) \neq \varnothing$ then $\mathrm{P}(A) = \overline{\mathcal{K}}$ and $\mathrm{P}^{\mathrm{P}}(A) = \{\bot\} \cup \{\mu_\oplus(A_{rr}) \mid C_r \in \overline{V}_A\}$.*
   - *(b) If $\overline{\mathrm{zc}}(A) = \varnothing$ then $\mathrm{P}(A) = \overline{\mathcal{K}} \backslash \{\bot\}$ and $\mathrm{P}^{\mathrm{P}}(A) = \{\mu_\oplus(A_{rr}) \mid C_r \in \overline{V}_A\}$.*

Proof. First, call $\overline{\mathrm{zc}}(A)$ the set of empty columns of A. If G_A has no cycles $C_A^+ = \varnothing$, claim 1 follows from a result in [7] . But if $C_A^+ \neq \varnothing$ then by Proposition 3, $\mathrm{P}^{\mathrm{P}}(A) \supseteq \{\mu_\oplus(A_{rr}) \mid C_r \in \overline{V}_A\}$ and no other non-null proper eigenvalues may exist by Lemma 4. $\bot$ is only proper when $\overline{\mathrm{zc}}(A) \neq \varnothing$ hence claims 2a and 2b follow. □

Translating to UFNF terms:

Corollary 1. *Let* $A \in \mathcal{M}_n(\overline{\mathcal{K}})$ *be a matrix over a complete selective radicable semifield with* $C_A^+ \neq \varnothing$*. Then* $\mathrm{P}(A) = \overline{\mathcal{K}} \backslash \{\bot\}$ *and* $\mathrm{P}^{\mathrm{p}}(A) = \{\mu_\oplus(A_{rr}) \mid C_r \in \overline{V}_A\}$*, unless* A *is in UFNF*$_3$ *and* $\overline{zc}(A) \neq \varnothing$ *whence* $\bot \in \mathrm{P}^{\mathrm{p}}(A) \subseteq \mathrm{P}(A)$ *too.*

Proof. If A is irreducible or in UFNF$_1$ or UFNF$_2$ it has no empty columns or rows. This can only happen in UFNF$_3$ in which case the result follows from Theorem 3. □

Since this solves entirely the description of the spectrum, only the description of the eigenspaces is left pending.

3.2 Eigenspaces of matrices in UFNF$_1$

In this section we offer an instance of how the UFNF can be used to obtain, inductively, the spectrum of reducible matrices.

If for every parallel condensation class $C_p \subseteq V_A$ in $A(C_r)$ illustrated in C_r of Fig. 1 $A_{up} \neq \mathcal{E}_{up}$ or $A_{pd} \neq \mathcal{E}_{pd}$ or both, then A is in UFNF$_1$ with a single connected block. In this case, we can relate the order of the eigenvectors to the condensation digraph: define the *support of a class* $\mathrm{supp}(C)$ as the support of any of the non-null eigenvectors it induces in A.

Lemma 5. *Let* $A \in \mathcal{M}_n(\mathcal{S})$ *be a matrix in UFNF*$_1$ *over a complete zerosumfree semiring. Then, for any class* $C_r \in \overline{V}_A$*,* $\mathrm{supp}(C_r) = \bigcup\{C_{l_r} \mid C_{l_r} \in \downarrow C_r\}$.

Proof. Since A_{rr} is irreducible, if $\rho = \mu_\oplus(A_{rr})$ then for any $v_r \in \mathcal{V}_\rho(A_{rr})$ we have that $\mathrm{supp}(v_r) = V_r$, hence $V_r \subseteq \mathrm{supp}(C_r)$. Also, since $\mathcal{S}$ is complete and zerosumfree $(\tilde{A}^\rho)^+_{rr}$ exists and is full [7]. Since $(\tilde{A}^\rho)^+_{uu}\tilde{A}^\rho_{ur}$ must have a full column for any $C_{l_r} \in \downarrow C_r$ meaning precisely that C_r is downstream from C_{l_r}, the product $(\tilde{A}^\rho)^+_{uu}\tilde{A}^\rho_{ur}(\tilde{A}^\rho)^+_{rr}$ for the nodes in C_{l_r} is non-null and $V_{l_r} \subseteq \mathrm{supp}(C_r)$. □

Lemma 5 establishes a bijection between the supports of condensation classes and downsets in $\overline{G}_A$ which is actually an isomorphism of orders,

Proposition 4. *Let* $A \in \mathcal{M}_n(\overline{\mathcal{K}})$ *be a matrix over a commutative complete selective radicable semifield admitting an UFNF*$_1$*. Then*

1. *Each class* $C_r \in \overline{V}_A$ *generates a distinct saturated eigenvector,* $v_r^\top$*.*
2. $\{v_r^\top \mid C_r \in \overline{V}_A\} \cong \overline{G}_A$.

Proof. Let $v \in \mathcal{V}_\rho(A)$ where $\rho = \mu_\oplus(A_{rr})$ then by Lemma 5 $\mathrm{supp}(v) = \downarrow C_r$, hence $v_r^\top = \top v \in \mathcal{V}_\rho(A)$ is the unique saturated eigenvector, since sat-supp $(\top v) = \mathrm{supp}(\top v) = \mathrm{supp}(C)$, and the bijection follows. By Lemma 3, claim 2 the order relation between classes is maintained between eigenvectors, whence the order isomorphism in claim 2. □

We call $\mathrm{FEV}^{1,\top}(A) = \{v_r^\top \mid C_r \in \overline{V}_A\}$ the *set of of saturated fundamental eigenvectors of* A. Notice that $\overline{V}_{A^{\mathrm{T}}} = \overline{V}_A$ but $\overline{E}_{A^{\mathrm{T}}} = \overline{E}_A^{\mathrm{d}}$, so $\mathrm{FEV}^{1,\top}(A^{\mathrm{T}}) \cong \overline{G}_A^{\mathrm{d}}$.

For every *finite* $\rho \in \mathrm{P}^{\mathrm{P}}(A)$ we define the *critical nodes* $V_\rho^c = \{i \in \overline{\mathbf{n}} \mid (\tilde{A}^\rho)_{ii}^+ = e\}$ and $\mathrm{FEV}_\rho^{1,\mathrm{F}}(A) = \{(\tilde{A}^\rho)_{\cdot i}^+ \mid i \in V_\rho^c\}$ the *(maybe partially) finite fundamental eigenvectors of* ρ. Next, let $\delta_\rho^{-1}(\rho') = e$ if $\rho' = \rho$ and $\delta_\rho^{-1}(\rho') = \top$ otherwise. for $\rho \in \mathrm{P}(A)$ the set of *(right) fundamental eigenvectors of* A *in UFNF*$_1$ *for* ρ as

$$\mathrm{FEV}_\rho^1(A) = \cup_{\rho' \in \mathrm{P}(A)} \{\delta_\rho^{-1}(\rho') \underset{\cdot}{\otimes} \mathrm{FEV}_{\rho'}^{1,\mathrm{F}}(A)\} \ . \tag{6}$$

This definition absorbs two cases, actually,

Lemma 6. *Let* $A \in \mathcal{M}_n(\overline{\mathcal{K}})$ *be a matrix over a commutative complete selective radicable semifield admitting an UFNF*$_1$*. Then,*

1. *for* $\rho \in \mathrm{P}(A) \backslash \mathrm{P}^{\mathrm{P}}(A)$, $\mathrm{FEV}_\rho^1(A) = \mathrm{FEV}^{1,\top}(A)$.
2. *for* $\rho \in \mathrm{P}^{\mathrm{P}}(A)$, $\rho \neq \top$, $\mathrm{FEV}_\rho^1(A) = \mathrm{FEV}_\rho^{1,\mathrm{F}}(A) \cup \mathrm{FEV}^{1,\top}(A) \backslash (\top \underset{\cdot}{\otimes} \mathrm{FEV}_\rho^{1,\mathrm{F}}(A))$.
3. *for* $\rho \in \mathrm{P}(A)$, $\rho \neq \top$, $\mathrm{FEV}^{1,\top}(A) = \top \underset{\cdot}{\otimes} \mathrm{FEV}_\rho^1(A)$.

Proof. If $\rho \in \mathrm{P}(A) \backslash \mathrm{P}^{\mathrm{P}}(A)$, then for all $\rho' \in \overline{\mathcal{K}}$, $\delta_\rho^{-1}(\rho') = \top$. By Proposition 4 claim 1 follows as we range $\rho' \in \mathrm{P}^{\mathrm{P}}(A)$. Similarly, when $\rho \in \mathrm{P}^{\mathrm{P}}(A)$, those classes whose $\rho' \neq \rho$ supply a single saturated eigenvector. However, if $\rho' = \rho$, then $\delta_\rho^{-1}(\rho') = e$ obtains the (partially) finite fundamental eigenvectors $\mathrm{FEV}_\rho^{1,\mathrm{F}}(A)$, the saturated eigenvectors of which cannot be considered fundamental, since they can be obtained from $\mathrm{FEV}_{\rho'}^{1,\mathrm{F}}(A)$ linearly, and will not appear in $\mathrm{FEV}_\rho^1(A)$. Claim 3 is a corollary of the other two. ☐

Call $\mathcal{V}^\top(A) = \langle \mathrm{FEV}^{1,\top}(A) \rangle_{\overline{K}}$ the *saturated eigenspace of* A , then

Corollary 2. *Let* $A \in \mathcal{M}_n(\overline{\mathcal{K}})$ *be a matrix over a commutative complete selective radicable semifield admitting an UFNF*$_1$*. Then,*

1. *For* $\rho \in \mathrm{P}(A)$, $\mathcal{V}^\top(A) \subseteq \mathcal{V}_\rho(A)$.
2. *For* $\rho \in \mathrm{P}(A) \backslash \mathrm{P}^{\mathrm{P}}(A)$*, furthermore,*$\mathcal{V}^\top(A) = \mathcal{V}_\rho(A)$.

Proof. Since we have $\mathrm{FEV}^{1,\top}(A) \subseteq \mathcal{V}_\rho(A)$, then $\mathcal{V}^\top(A) \subseteq \mathcal{V}_\rho(A)$. For $\rho \in \mathrm{P}(A) \backslash \mathrm{P}^{\mathrm{P}}(A)$, $\mathrm{FEV}_\rho^1(A) = \mathrm{FEV}^{1,\top}(A)$ by Lemma 6, so claim 2 follows. ☐

Hence, $\mathcal{V}^\top(A)$ provides a common "scaffolding" for every eigenspace, while the peculiarities for proper eigenvalues are due to the finite eigenvectors. Also, since $\mathcal{V}^\top(A)$ is a complete lattice, $\mathrm{FEV}^{1,\top}(A) \subseteq \mathcal{V}^\top(A)$ is actually a lattice embedding,

Proposition 5. *Let* $A \in \mathcal{M}_n(\overline{\mathcal{K}})$ *be a matrix over a commutative complete selective radicable semifield admitting an UFNF*$_1$*. Then*

1. *For* $\rho \in \mathrm{P}(A) \backslash \mathrm{P}^{\mathrm{P}}(A)$,

$$\mathcal{U}^\top(A) = \langle \mathrm{FEV}^{1,\top}(A^{\mathrm{T}})^{\mathrm{T}} \rangle_{\mathfrak{Z}} \cong \mathcal{F}(\overline{G}_A) \quad \mathcal{V}^\top(A) = \langle \mathrm{FEV}^{1,\top}(A) \rangle_{\mathfrak{Z}} \cong \mathcal{O}(\overline{G}_A) \ . \tag{7}$$

2. for all $\rho \in \mathrm{P}^{\mathrm{P}}(A)$*,* $\rho < \top$

$$\mathcal{U}_{\lambda}(A) = \langle \mathrm{FEV}^{1}_{\lambda}(A^{\mathrm{T}})^{\mathrm{T}} \rangle_{\overline{K}} \qquad\qquad \mathcal{V}_{\rho}(A) = \langle \mathrm{FEV}^{1}_{\rho}(A) \rangle_{\overline{K}} \ .$$

Proof. If $v_r^{\top} \in \mathrm{FEV}^{1,\top}(A)$ then $\lambda v_r^{\top} = \lambda(\top v_r^{\top}) = v_r^{\top}$, whence $\mathcal{V}^{\top}(A) = \langle \mathrm{FEV}^{1,\top}(A) \rangle_{\mathbb{3}}$. In fact, the generation process may proceed on only a subsemiring of $\overline{\mathcal{K}}$ which need not even be complete. For instance, we may use any of the isomorphic copies of $\mathbb{2}$ embedded in $\overline{\mathcal{K}}$, for instance $\{\bot, k\} \cong \mathbb{2}$, with $k \neq \bot$.

Since the number of saturated eigenvectors is finite, being identical to the number of condensation classes, we only have to worry about binary joins and meets. Recall that $v_r^{\top} \vee v_k^{\top} = v_r^{\top} \underset{\cdot}{\oplus} v_k^{\top}$ and $v_r^{\top} \wedge v_k^{\top} = v_r^{\top} \dot{\oplus} v_k^{\top} = \left((v_r^{\top})^{-1} \underset{\cdot}{\oplus} (v_k^{\top})^{-1} \right)^{-1}$. Then $\mathrm{supp}(v_r^{\top} \underset{\cdot}{\oplus} v_k^{\top}) = \mathrm{supp}(v_r^{\top}) \cup \mathrm{supp}(v_k^{\top})$ and also

$$\mathrm{supp}(v_r^{\top} \dot{\oplus} v_k^{\top}) = \left(\mathrm{supp}^{\mathbf{c}}\left(v_r^{\top}\right) \cup \mathrm{supp}^{\mathbf{c}}\left(v_k^{\top}\right)\right)^{\mathbf{c}} = \mathrm{supp}(v_r^{\top}) \cap \mathrm{supp}(v_k^{\top})$$

for $C_r, C_k \in \overline{V}_A$ and Proposition 1 gives the result. For $\rho \in \mathrm{P}^{\mathrm{P}}(A)$, $\mathrm{FEV}^{1}_{\rho}(A) \subseteq \mathcal{V}_{\rho}(A)$ implies that $\langle \mathrm{FEV}^{1}_{\rho}(A) \rangle_{\overline{\mathcal{K}}} \subseteq \mathcal{V}_{\rho}(A)$, and Corollary 4 ensures that no finite vectors are missing. And dually for left eigenspaces. □

This actually proves that $\mathrm{FEV}^{1}_{\rho}(A)$ is join-dense in $\mathcal{V}_{\rho}(A)$.

As already mentioned, $\mathcal{V}_{\rho}(A)$ is a hard-to-visualize semimodule. An *eigenspace schematics* is a modified order diagram where the saturated eigenspace is represented in full but the rays generated by finite eigenvalues $\{\kappa \underset{\cdot}{\otimes} (\tilde{A}^{\rho})^{+}_{\cdot i} \mid i \in V_r^{c}, \rho = \mu_{\oplus}(A_{rr})\}$ are drawn with discontinuous lines, as in the examples below.

We are now introducing another representation inspired by (7): we define the *(left) right eigenlattices of* A *for (*$\lambda \in \Lambda(A)$*)* $\rho \in \mathrm{P}(A)$ as

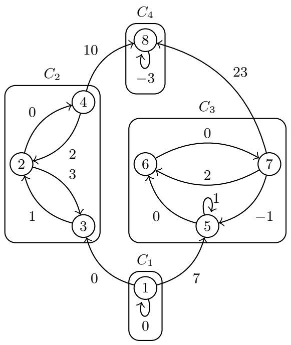
$$\mathcal{L}_{\lambda}(A) = \langle \mathrm{FEV}^{1}_{\rho}(A^{\mathrm{T}})^{\mathrm{T}} \rangle_{\mathbb{3}} \qquad\qquad \mathcal{L}_{\rho}(A) = \langle \mathrm{FEV}^{1}_{\rho}(A) \rangle_{\mathbb{3}} \ .$$

Example 3 (Spectral lattices of irreducible matrices). Since irreducible matrices are in UFNF_1 with a single class, $\mathrm{FEV}^{0}_{\mu_{\oplus}(A)}(A) = \mathrm{FEV}^{1}_{\mu_{\oplus}(A)}(A)$. For $\rho \in \mathrm{P}(A) \backslash \mathrm{P}^{\mathrm{P}}(A)$ we have $\mathrm{FEV}^{0,\top}(A) = \{\top^{n}\}$, whence $\overline{G}_A \cong \mathbb{1}$ and $\mathcal{V}^{\top}(A) = \{\bot^{n}, \top^{n}\} \cong \mathbb{2}$. For $\rho \in \mathrm{P}^{\mathrm{P}}(A)$, $\rho < \top$, as proven in [7], $\mathcal{V}_{\rho}(A)$ is finitely generable from $\mathrm{FEV}^{0}_{\rho}(A)$, but the form of the eigenspace and eigenlattice for $\Lambda^{\mathrm{P}}(A) = \{\mu_{\oplus}(A)\} = \mathrm{P}^{\mathrm{P}}(A)$ depends on the critical cycles and the eigenvectors they induce. □

Example 4. Consider the matrix $A \in \mathcal{M}_n(\overline{\mathbb{R}}_{\max,+})$ from [13, p. 25.7, example 2] in UFNF_1 depicted in Fig. 2.(a). The condensed graph $\overline{G}_A$ in Fig. 2.(b) has for vertex set $\overline{V}_A = \{C_1 = \{1\}, C_2 = \{2,3,4\}, C_3 = \{5,6,7\}, C_4 = \{8\}\}$, so consider the strongly connected components $G_{A_{kk}} = (C_k, E \cap C_k \times C_k), 1 \leq k \leq 4$. Their maximal cycle means are $\mu_k = \mu_{\oplus}(A_{kk})$: $\mu_1 = 0$, $\mu_2 = 2$, $\mu_3 = 1$ and $\mu_4 = -3$, respectively, corresponding to critical circuits: $C^{c}(G_{A_{11}}) = \{1 \circlearrowleft\}$, $C^{c}(G_{A_{22}}) = \{2 \to 3 \to 2\}$, $C^{c}(G_{A_{33}}) = \{5 \circlearrowleft, 6 \to 7 \to 6\}$, $C^{c}(G_{A_{44}}) = \{8 \circlearrowleft\}$.

$$A_3 = \begin{bmatrix} 0 & \cdot & 0 & \cdot & 7 & \cdot & \cdot & \cdot \\ \cdot & \cdot & 3 & 0 & \cdot & \cdot & \cdot & \cdot \\ \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & 2 & \cdot & \cdot & \cdot & \cdot & \cdot & 10 \\ \cdot & \cdot & \cdot & \cdot & 1 & 0 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 & \cdot \\ \cdot & \cdot & \cdot & \cdot & -1 & 2 & \cdot & 23 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -3 \end{bmatrix}$$

(a) A reducible matrix in UFNF_1



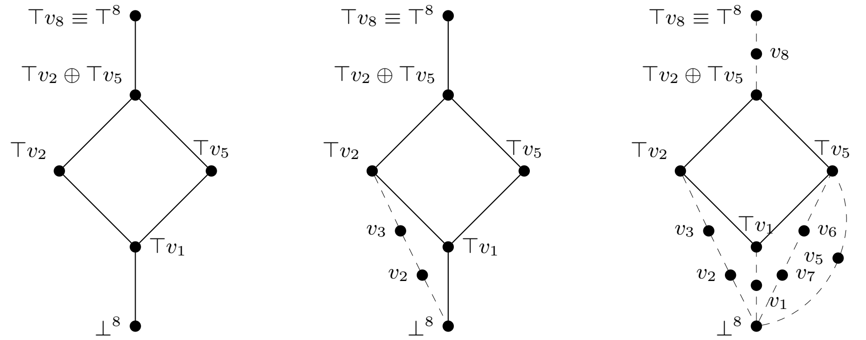
(b) Class diagram (rectangles) overlaid on G_{A_3}

$$\begin{array}{c|cccccccc} 1 & 0 & \top & \top & \top & \top & \top & \top & \top \\ 2 & \cdot & 0 & 1 & -2 & \cdot & \cdot & \cdot & 6 \\ 3 & \cdot & -1 & 0 & -3 & \cdot & \cdot & \cdot & 5 \\ 5 & \cdot & \cdot & \cdot & \cdot & 0 & -1 & -2 & 20 \\ 6 & \cdot & \cdot & \cdot & \cdot & -3 & 0 & -1 & 21 \\ 7 & \cdot & \cdot & \cdot & \cdot & -2 & 1 & 0 & 22 \\ 8 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \end{array}$$

(c) Left fundamental eigenvectors

$$\begin{array}{ccccccc} 1 & 2 & 3 & 5 & 6 & 7 & 8 \\ \hline 0 & -3 & -2 & 6 & 5 & 4 & \top \\ \cdot & 0 & 1 & \cdot & \cdot & \cdot & \top \\ \cdot & -1 & 0 & \cdot & \cdot & \cdot & \top \\ \cdot & 0 & 1 & \cdot & \cdot & \cdot & \top \\ \cdot & \cdot & \cdot & 0 & -1 & -2 & \top \\ \cdot & \cdot & \cdot & -3 & 0 & -1 & \top \\ \cdot & \cdot & \cdot & -2 & 1 & 0 & \top \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 0 \end{array}$$

(d) Right fundamental eigenvectors



(e) $\mathcal{V}^{\top}(A_3)$ (f) Schematics of $\mathcal{V}_2(A_3)$ (g) Schematics of $\mathcal{V}(A_3)$

Fig. 2: Matrix A_3 (a), its associated digraph and class diagram (b), its left (c) and right (d) fundamental eigenvectors annotated with their eigennodes to the left and above, respectively; the eigenspace of improper eigenvectors $\mathcal{V}^{\top}(A_3)$ in (e), a schematic of the right eigenspace of proper eigenvalue $\rho = 2$, $\mathcal{V}_2(A_3)$ in (f) and the schematics of the whole right eigenspace $\mathcal{V}(A_3)$ in (g).

Note that node 4 does not generate an eigenvector in either spectrum, since it does not belong to a critical cycle.

Therefore $\Lambda^{\mathrm{P}}(A_3) = \mathrm{P}^{\mathrm{P}}(A_3) = \{2, 1, 0, -3\}$ each left eigenspace is the span of the set of eigenvectors chosen from distinct critical cycles for each class of A: $\mathcal{U}_{\mu_1}(A) = \langle (\tilde{A}_3{}^{\mu_1})^{\overset{+}{}}_{1\cdot} \rangle$, $\mathcal{U}_{\mu_2}(A) = \langle (\tilde{A}_3{}^{\mu_2})^{\overset{+}{}}_{2\cdot} \rangle$, $\mathcal{U}_{\mu_3}(A) = \langle (\tilde{A}_3{}^{\mu_3})^{\overset{+}{}}_{\{5,6\}\cdot} \rangle$, and $\mathcal{U}_{\mu_4}(A) = \langle (\tilde{A}_3{}^{\mu_4})^{\overset{+}{}}_{8\cdot} \rangle$ –as described by the row vectors of Fig. 2.(c)–and the right eigenspaces are $\mathcal{V}_{\mu_1}(A) = \langle (\tilde{A}_3{}^{\mu_1})^{\overset{+}{}}_{\cdot 1} \rangle$, $\mathcal{V}_{\mu_2}(A) = \langle (\tilde{A}_3{}^{\mu_2})^{\overset{+}{}}_{\cdot 2} \rangle$, $\mathcal{V}_{\mu_3}(A) = \langle (\tilde{A}_3{}^{\mu_3})^{\overset{+}{}}_{\cdot\{5,6\}} \rangle$, and $\mathcal{V}_{\mu_4}(A) = \langle (\tilde{A}_3{}^{\mu_4})^{\overset{+}{}}_{\cdot 8} \rangle$ –as described by the column vectors of Fig. 2.(d).

The saturated eigenspace is easily represented by means of an order diagram–Hasse diagram–as that of Fig. 2.(e). Note how it is embedded in that of any proper eigenvalue like $\rho = 2$ in Fig. 2.(f). Since the representation of continuous eigenspaces is problematic, we draw s*chematics* of them, as in Fig. 2.(f). Fig. 2.(g) shows a schematic view of the union of the eigenspaces for proper eigenvalues $\mathcal{V}(A_3) = \cup_{\rho \in \mathrm{P}^{\mathrm{P}}(A)} \mathcal{V}_\rho(A_3)$. ☐

4 Discussion

In this paper, we have discussed the spectrum of reducible matrices with entries in completed idempotent semifields. To the extent of our knowledge, this was pioneered in [14] and both [7] and this paper can be understood as systematic explorations to try and understand what was stated in there. For this purpose, the consideration of particular UFNF forms for the matrices has been crucial: while the description in [14] is combinatorial ours is constructive.

The usual notion of spectrum as the set of eigenvectors with more than one (null) eigenvector appears in this context as too weak: when a matrix has at least one cycle then all the values in the semifield (except the bottom $\perp$) belong to the spectrum. If the matrix has at least one empty column (resp. empty row) and a cycle then *all* of the semifield is the spectrum. Rather than redefine the notion of spectrum we have decided to introduce the *proper spectrum* as the set of eigenvalues with at least one vector with finite support.

Regarding the eigenspaces, we found not only that they are complete continuous lattices for proper eigenvalues, but also that they are *finite (complete) lattices* for *improper* eigenvalues. Looking for a device to represent the information within each proper eigenspace we focus on the fundamental eigenvectors of an irreducible matrix for each eigenvalue: those with unit values in certain of their coordinates. The span of those eigenvectors by the action of the $\mathbb{3}$-bounded lattice-ordered group generates the finite eigenlattices. Interestingly, since improper eigenvectors only have non-finite coordinates, their span by the $\mathbb{3}$-blog is exactly the same finite lattice as their span by the whole semifield itself.

With these building blocks it is easy to build finite lattices for reducible matrices of any UFNF description, as exemplified above. We believe this will be a useful technique to understand and visualize the concept lattices of formal contexts with entries in an idempotent semifield and other dioids.

Bibliography

[1] Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin, Heidelberg (1999)

[2] 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

[3] Belohlavek, R.: Optimal decompositions of matrices with entries from residuated lattices. Journal Of Logic And Computation **22**(6) (November 2012) 1405–1425

[4] Valverde-Albacete, F.J., Pelaez-Moreno, C.: Spectral lattices of $(R_{max,+})$-formal contexts. In: Formal Concept Analysis. Volume LCNS 4933. (2008) 124–139

[5] Goguen, J.A.: L-fuzzy sets. J. Math. Anal. Appl **18**(1) (1967) 145–174

[6] Gondran, M., Minoux, M.: Dioids and semirings: links to fuzzy set and other applications. Fuzzy Sets And Systems **158**(1273–1294) (April 2007)

[7] Valverde-Albacete, F.J., Peláez-Moreno, C.: The spectra of irreducible matrices over completed commutative idempotent semifields. In preparation (2013)

[8] Gondran, M., Minoux, M.: Valeurs propres et vecteurs propres dans les dioïdes et leur interprétation en théorie des graphes. EDF, Bulletin de la Direction des Etudes et Recherches, Serie C, Mathématiques Informatique **2** (1977) 25–41

[9] Valverde-Albacete, F.J., Pelaez-Moreno, C.: Extending conceptualisation modes for generalised Formal Concept Analysis. Information Sciences **181** (May 2011) 1888–1909

[10] Davey, B., Priestley, H.: Introduction to lattices and order. 2nd edn. Cambridge University Press, Cambridge, UK (2002)

[11] Schmidt, G., Ströhlein, T.: Relations and Graphs. Discrete Mathematics for Computer Scientists. Springer (1993)

[12] Brualdi, R.A., Ryser, H.J.: Combinatorial Matrix Theory. Volume 39 of Encyclopedia of Mathematics and its Applications. Cambridge University Press (1991)

[13] Akian, M., Bapat, R., Gaubert, S.: 25. [15] 25–1–25–17

[14] Jun, M., Yan, G., zhi Yong, D.: The eigen-problem in the completely max-algebra. In: Proceedings of the Sixth International Conference on Parallel and Distributed Computing. (2005)

[15] Hogben, L., ed.: Handbook of Linear Algebra. Discrete Mathematics and Its Applications. Chapman & Hall/CRC (2007)

On information retrieval in morphological image and signal processing

C. Alcalde[1], A. Burusco[2], J.C. Díaz[3], R. Fuentes-González[2], and J. Medina[3]*

[1] Dpt. Matemática Aplicada. Universidad del País Vasco UPV/EHU. Spain
Email: `c.alcalde@ehu.es`
[2] Dpt. Automática y Computación, Univ. Pública de Navarra. Spain
Email: {`burusco,rfuentes`}`@unavarra.es`
[3] Dpt. of Mathematics, University of Cádiz. Spain†
Email: {`juancarlos.diaz,jesus.medina`}`@uca.es`

Abstract. Mathematical Morphology is a theory concerned with the processing and analysis of images or signals using filters and other operators that modify them. This paper studies how the original images and signals can be retrieved using fuzzy property-oriented concept lattices and fuzzy relation equations.

1 Introduction

The fundamentals of mathematical morphology (initiated by G. Matheron [18] and J. Serra [22]), are in the set theory, the integral geometry and the lattice algebra. This methodology is used in the recent years in general contexts related to activities as the information extraction in digital images, the noise elimination or the pattern recognition.

The generalization of this theory, in order to consider fuzzy subsets as objects, was given in [5–9, 16, 17] using L-fuzzy sets as images and structuring elements, which was called *fuzzy morphological image processing.*

On the other hand, the fuzzy property-oriented concept lattice framework [15] arises as a fuzzy generalization of Rough Set Theory and in which a set of objects and a set of attributes are assumed, following the view point of Formal Concept Analysis. This theory has been considered to solve fuzzy relation equations [12] and important results, in order to obtain the whole set of solutions, are given.

Recently, both theories, fuzzy mathematical morphology and fuzzy property-oriented concept lattice, have been related [1], extending the initial relation given in [2].

In mathematical morphology, the usual procedure is, given a structuring image, to obtain the dilation and the erosion from an initial image. But what happen if we lose the original image or, simply, we have not got it because we

---

[*] Corresponding author.

[†] Partially supported by the Spanish Science Ministry projects TIN2009-14562-C05 and TIN2012-39353-C04, and by Junta de Andalucía project P09-FQM-5233.

only know its corresponding dilation or erosion, how can the original image be obtained?

This paper studies the problem of objects retrieval in the framework of mathematical morphology. It is usual that there is noise in the transmission of information or, in several cases, it is easier to send a kind of image than another one. Hence, the received object is not equal to the original one. Note that this problem is also related to other settings, such as object recognition.

Specifically, we focus on solving the problem of obtaining the original object A from another one received B, assuming a structuring image and that B is the dilation or the erosion of the original image A. For that, this problem will be written as a fuzzy relation equation and the relationship introduced in [1] and the results given in [12] will be used to solve it.

2 Preliminaries

This section recalls the fuzzy property-oriented concept lattice framework [15], fuzzy mathematical morphology [5–9, 16, 17], the relationship beetween them, introduced in [1], and fuzzy relation equations [21].

2.1 L-fuzzy property-oriented concept lattice

In this framework a complete residuated lattice $(L, \vee, \wedge, *, I, 0, 1, \leq_?)$ is considered as algebraic structure. Moreover, a fuzzy (formal) context is assumed, (X, Y, R), where $R\colon X \times Y \to L$ is a fuzzy relation between the sets X and Y, where X can be interpreted as a set of objects and Y as a set of properties (attributes).

Given a fuzzy context (X, Y, R), two mappings $R_\exists\colon L^X \to L^Y$ and $R^\forall\colon L^Y \to L^X$ can be defined as:

$$R_\exists(A)(y) = \sup\{A(x) * R(x, y) \mid x \in X\} \tag{1}$$

$$R^\forall(B)(x) = \inf\{I(R(x, y), B(y)) \mid y \in Y\} \tag{2}$$

for all $A\colon X \to L$, $B\colon Y \to L$, $x \in X$ and $y \in Y$, where I is the residuated implication associated with the conjunctor $*$. Examples of these operators are given in [3, 19].

As a first result, the pair $(R_\exists, R^\forall)$ forms an isotone Galois connection [13]. Therefore, a *property-oriented concept* (or, a *concept based on rough set theory*) of (X, Y, R) is a pair $(A, B) \in L^X \times L^Y$ such that $B = R_\exists(A)$ and $A = R^\forall(B)$.

The set of all property-oriented concepts of (X, Y, R) is denoted by $\mathcal{P}(X, Y, R)$ and it is a complete lattice [15], which is called the *property-oriented concept lattice of* (X, Y, R) (or, the *concept lattice of* (X, Y, R) *based on rough set theory*) [15]. For that isotone Galois connection $(R_\exists, R^\forall)$ and lattice $\mathcal{P}(X, Y, R)$ interesting properties have been proven, e.g., in [3, 4, 13, 15].

2.2 Fuzzy mathematical morphology

Fuzzy morphological image processing has been developed using L-fuzzy sets A and S (with $X = \mathbb{R}^2$ or $X = \mathbb{Z}^2$) as images and structuring elements in [5–9, 16, 17]. The structuring image S represents the effect that we want to produce over the initial image A.

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. In the literature (see [6, 9, 14, 17]) erosion and dilation operators are introduced associated with the residuated pair $(*, I)$ as follows:

If $S \colon X \to L$ 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:*

$$\varepsilon_S(A)(x) = \inf\{I(S(y-x), A(y)) \mid y \in X\} \qquad \textit{for all } x \in X$$

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

$$\delta_S(A)(x) = \sup\{S(x-y) * A(y) \mid y \in X\} \qquad \textit{for all } x \in X$$

From these definitions arise two mappings which will be called the fuzzy erosion and dilation operators $\varepsilon_S, \delta_S \colon L^X \to L^X$.

2.3 Relationship between both theories

In [1] these previous theories were related. For that, first of all, any fuzzy image $S \in L^X$ was associated with the fuzzy relation $R_S \in L^{X \times X}$, defined as:

$$R_S(x, y) = S(y - x)$$

for all $x, y \in X$. Hence, the fuzzy erosion and dilation of an L-fuzzy subset A of X are written as follows:

$$\varepsilon_S(A)(x) = \inf\{I(R_S(x, y), A(y)) \mid y \in X\}$$
$$\delta_S(A)(x) = \sup\{R_S(y, x) * A(y) \mid y \in X\}$$

and the following results were proved.

Proposition 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 property-oriented context whose incidence relation is the relation R_S associated with S. Then the erosion ε_S and dilation δ_S operators in (L, X, S) are related to the derivation operators $(R_S)^\forall$ and $(R_S)_\exists$ in the L-fuzzy property-oriented context (L, X, X, R_S) by:*

$$\varepsilon_S(A) = (R_S)^\forall(A)$$
$$\delta_S(A) = (R_S)_\exists(A)$$

for all $A \in L^X$.

This relation provides that the dilation and erosion have the properties of the isotone Galois connection $(R_\exists, R^\forall)$. The following result shows the connection between the outstanding morphological elements and the L-fuzzy property-oriented concepts.

Theorem 1. *Let $S \in L^X$ and its associated relation $R_S \in L^{X\times X}$, the following statements are equivalent:*

1. *The pair $(A, B) \in L^X \times L^X$ is an L-fuzzy property-oriented concept of the context (L, X, X, R_S).*
2. *A is S-closed (i.e. $\varepsilon_S \circ \delta_S(A) = A$) and B is the S-dilation of A.*
3. *B is S-open (i.e. $\delta_S \circ \varepsilon_S(B) = B$) and A is the S-erosion of B.*

2.4 Systems of fuzzy relation equations

Systems of fuzzy relation equations have been widely studied, for instance in [10, 11, 20]. This section recalls these kind of systems for which a residuated lattice $(L, \vee, \wedge, *, I, 0, 1, \leq)$ is fixed.

A *system of fuzzy relation equations with* sup-$*$-*composition* (SFRE$*$), is the following system of equations

$$A_i \circ R = B_i, \quad \text{or} \quad \bigvee_{u\in U} (A_i(u) * R(u, v)) = B_i(v), \quad i \in \{1, \dots, n\} \tag{3}$$

where $R \in L^{U\times V}$ is an unknown fuzzy relation, $A_1, \dots, A_n \in L^U$ and $B_1, \dots, B_n \in L^V$. Its counterpart is a *system of fuzzy relation equations with* inf-I-*composition* (SFREI), that is,

$$A_j^* \lhd R = D_j \quad \text{or} \quad \bigwedge_{v\in V} I(A_j(v), R(u, v)) = D_j(u), \quad j \in \{1, \dots, m\} \tag{4}$$

considered with respect to an unknown fuzzy relation $R \in L^{U\times V}$ and the fuzzy subsets $A_1^*, \dots, A_m^* \in L^V$ and $D_1, \dots, D_m \in L^U$.

Given the universes $U = \{u_1, \dots, u_m\}$ and $V = \{v_1, \dots, v_n\}$, an unknown fuzzy relation $R \in L^{U\times V}$ and two arbitrarily chosen fuzzy subsets of respective universes $A_1 \dots, A_n \in L^U$, $B_1, \dots, B_n \in L^V$. If an element of V is fixed, $v \in V$, $A_i(u_j) = a_{ij}$ for $i \in \{1, \dots, n\}$, $j \in \{1, \dots, m\}$, $R(u_j, v) = x_j$, $B_i(v) = b_i$, then System (3) can be written as

$$\begin{array}{ccccc} a_{11} * x_1 \vee \cdots \vee a_{1m} * x_m & = & b_1 \\ \vdots \quad\; \vdots \qquad\quad\; \vdots & \vdots & \vdots \\ a_{n1} * x_1 \vee \cdots \vee a_{nm} * x_m & = & b_n \end{array} \tag{5}$$

Consequently, for each $v \in V$, we obtain a column of R, thus, solving n SFRE$*$ systems, we will obtain the unknown relation. In order to obtain a SFREI system where the considered universes U, V and the unknown fuzzy relation $R \in L^{U\times V}$ are as in the previous system, an element of U is fixed, $u \in U$, fuzzy subsets $A_1^*, \dots, A_m^* \in L^V$, $D_1, \dots, D_m \in L^U$ are assumed, such that $A_j^*(v_i) = a_{ij}$ for $i \in \{1, \dots, n\}$, $j \in \{1, \dots, m\}$, $R(u, v_i) = y_i$ and $D_j(u) = d_j$. Therefore, System (4) can be written as

$$\begin{array}{rcl} I(a_{11}, y_1) \wedge \cdots \wedge I(a_{n1}, y_n) & = & d_1 \\ \vdots \qquad \vdots \qquad \vdots & \vdots & \vdots \\ I(a_{1m}, y_1) \wedge \cdots \wedge I(a_{nm}, y_n) & = & d_m \end{array} \tag{6}$$

Therefore, for each $u \in U$, we obtain a row of R, thus, solving m SFRE→ systems, the unknown relation will be obtained.

3 Images and signals retrieval

This section introduces an application of fuzzy relation equation to objects retrieval in the fuzzy mathematical morphology setting. From the relationship between fuzzy mathematical morphology and fuzzy property-oriented concept lattice, recalled in Section 2.3, and the relationship between fuzzy property-oriented concept lattice and fuzzy relation equations [12], we will solve the problem of obtaining the original object $A \colon X \to L$ from another one received $B \colon X \to L$ and a fixed structuring image $S \colon X \to L$.

Specifically, given an image $B \colon X \to L$, we can consider a structuring image $S \colon X \to L$ and ask if there exists $A \colon X \to L$ such that $\delta_S(A) = B$ and, if there exists, how to obtain it. Analogously, for each image $A \colon X \to L$, we can ask if there exists $B \colon X \to L$ such that $\varepsilon_S(B) = A$, for a structured image S, and, if there exists, how to obtain it.

First of all, we will write this mathematical morphology problem in terms of fuzzy relation equations.

Given a finite subset $X_0 = \{x_1, \dots, x_n\}$ of X, an image $B \colon X_0 \to L$ and a structuring image $S \colon X_0 \to L$, if we want to obtain an image $A \colon X_0 \to L$ such that $\delta_S(A) = B$, then we need to solve the following system:

$$\begin{array}{rcl} R_S(x_1, x_1) * A(x_1) \vee \cdots \vee R_S(x_n, x_1) * A(x_n) & = & b_1 \\ \vdots \qquad \vdots \qquad \vdots & \vdots & \vdots \\ R_S(x_1, x_n) * A(x_1) \vee \cdots \vee R_S(x_n, x_n) * A(x_n) & = & b_n \end{array} \tag{7}$$

in which $B(x_i) = b_i$, for all $i \in \{1, \dots, n\}$.

Analogously, given an image $A \colon X_0 \to L$ and a structuring image $S \colon X_0 \to L$, obtaining an image $B \colon X_0 \to L$, such that $\varepsilon_S(B) = A$, is equivalent to solve the system:

$$\begin{array}{rcl} I(R_S(x_1, x_1), B(x_1)) \wedge \cdots \wedge I(R_S(x_1, x_n), B(x_n)) & = & a_1 \\ \vdots \qquad \vdots \qquad \vdots & \vdots & \vdots \\ I(R_S(x_n, x_1), B(x_1)) \wedge \cdots \wedge I(R_S(x_n, x_n), B(x_n)) & = & a_n \end{array} \tag{8}$$

Next, several results will be presented in the fuzzy mathematical morphology framework, based on the properties introduced in [12]. The first one is about the solvability of Systems (7) and (8).

Theorem 2. *Given a finite subset $X_0 \subseteq X$ and $B \in L^{X_0}$, defined by $B(x_i) = b_i$, for each $i \in \{1, \dots, n\}$. System (7) can be solved if and only if B is S-open in X_0. In that case, $\varepsilon_S(B) \in L^{X_0}$ is the greatest solution.*

Analogously, given $A \in L^{X_0}$, defined by $A(x_i) = a_i$, for each $i \in \{1, \dots, n\}$. System (8) can be solved if and only if A is S-closed in X_0. In that case, $\delta_S(A) \in L^{X_0}$ is the least solution.

As a consequence, if the values b_i satisfy that $\delta_S(A)(x_i) = b_i \in L$, for all $i \in \{1, \dots, n\}$, then we can find a solution $A \in L^{X_0}$ of System (7). Analogously, if $\varepsilon_S(B)(x_i) = a_i \in L$ is known for all $i \in \{1, \dots, n\}$, then we can find $B \in L^{X_0}$ solving System (8).

The second result relates the independent term and greatest solution to a property-oriented concept.

Theorem 3. *System (7) can be solved if and only if $(\varepsilon_S(B), B)$ is an L-fuzzy property-oriented concept of the context (L, X_0, X_0, R_S), where $B \in L^{X_0}$ is defined by $B(x_i) = b_i$, for all $i \in \{1, \dots, n\}$.*

Similarly, System (8) can be solved if and only if $(A, \delta_S(A))$ is an L-fuzzy property-oriented concept of the context (L, X_0, X_0, R_S), where $A \in L^{X_0}$ is defined by $A(x_i) = a_i$, for all $i \in \{1, \dots, n\}$.

Now, we present an application to digital signals.

Example 1. Let us assume the set $X = \{0, 1, 2, \dots, 21, 22\} \subseteq Z$, the linear structure $L = \{0, 0.1, 0.2, \dots, 0.9, 1\}$, the mapping $B \colon X \to L$, which is represented in Figure 1, and the structuring set $S = \{-1, 0, 1\}$. Note that B can be interpreted as a 1-D discrete signal.

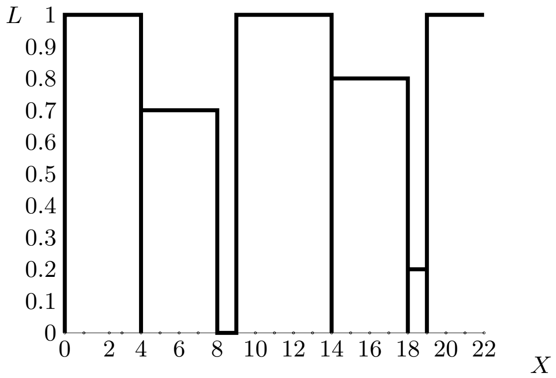


Fig. 1. Discrete signal received

From this environment a system of fuzzy relation equations similar to System (7) is considered, in order to obtain a signal A with dilation B.

First of all, we need to check if this system has a solution. Hence, we consider the context (L, X, X, R_S), where the fuzzy relation $R_S \subseteq X \times X$ is defined, for each $(x, y) \in X \times X$, as

$$R_S(x, y) = S(y - x) = \begin{cases} 1 & \text{if } |y - x| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

Since the signal B is S-open in X, that is $(\delta_S \circ \varepsilon_S)(B) = B$, by Theorems 2 and 3, we have that the considered system has, at least, one solution and the greatest solution A_g is $\varepsilon_S(B)$, which is given in Figure 2 and defined as

$$\varepsilon_S(B)(x) = \inf\{I(R_S(x, y), B(y)) \mid y \in X\} = \inf\{B(y) \mid |y - x| \leq 1\}$$

for all $x \in X$. Moreover, (A_g, B) is a fuzzy property-oriented concept.

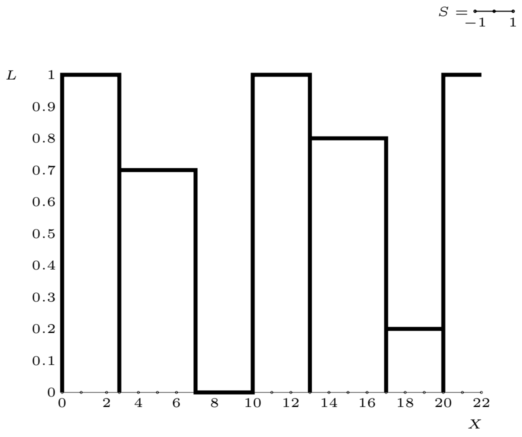


Fig. 2. Greatest solution

It is clear, in Example 1, that the original signals A may not be the greatest solution, but another one of the proposed system. In order to obtain the whole set of solutions the following result is introduced and can be proven from [12].

Theorem 4. *Given an S-open object $B \in L^{X_0}$, if $A \in L^{X_0}$ is the original object, such that $\delta_S(A) = B$, then either $A' < A \leq \varepsilon_S(B)$ for some (A', B') lower neighbour of $(\varepsilon_S(B), B)$ in $\mathcal{P}(X_0, X_0, R_S)$; or $A < \varepsilon_S(B)$ and A is incomparable with A', for all (A', B') lower neighbour of $(\varepsilon_S(B), B)$ in $\mathcal{P}(X_0, X_0, R_S)$.*

Analogously, given an S-closed object $A \in L^{X_0}$, if $B \in L^{X_0}$ is the original object, such that $\varepsilon_S(B) = A$, then either $\delta_S(A) \leq B < B'$ for some upper neighbour (A', B') of $(A, \delta_S(A))$ in $\mathcal{P}(X_0, X_0, R_S)$; or $\delta_S(A) < B$ and B is incomparable with B', for all (A', B') upper neighbour of $(A, \delta_S(A))$ in $\mathcal{P}(X_0, X_0, R_S)$.

Therefore, Theorem 4 can be applied in order to obtain the whole set of solutions of the system given in Example 1. However, we need to remark that, since straightforward $\varepsilon_S(B) \leq B$, we have that A_g is the solution closest to B. Hence, considering the greatest solution can be a good election.

The following example focus on images retrieval.

Example 2. We consider a two dimensional pixelated image. Hence, $X = \mathbb{Z}^2$, $L = \{0, 1\}$ and the mappings $A\colon \mathbb{Z}^2 \to L$ are interpreted as two dimensional images. The elements of X are denoted as $\overline{x} = (x_1, x_2) \in \mathbb{R}^2$.

In this case, the image $B\colon \mathbb{Z}^2 \to L$, given in Figure 3, is obtained and we want to retrieve the original image or a good approximation.

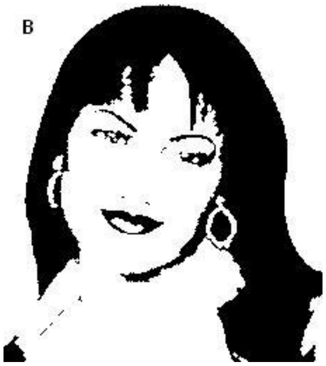


Fig. 3. Initial image

The structuring binary image S is the unit ball:

$$S = \{(x_1, x_2) \in \mathbb{Z} \times \mathbb{Z} \mid x_1 + x_2 \leq 1\}$$

given in Figure 4.

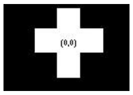


Fig. 4. Structuring set

Now, we consider a System (7) and a context $(\{0, 1\}, \mathbb{Z}^2, \mathbb{Z}^2, R_S)$, where the associated incidence relation $R_S \subseteq \mathbb{Z}^2 \times \mathbb{Z}^2$ is defined, for each $\overline{x} = (x_1, x_2)$,

$\overline{y} = (y_1, y_2)$, as

$$R_S(\overline{x}, \overline{y}) = S(\overline{y} - \overline{x}) = \begin{cases} 1 & \text{if } (y_1 - x_1) + (y_2 - x_2) \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

Therefore, by the previous results, the greatest solution $A_g \colon \mathbb{Z}^2 \to L$, associated with B and the structuring image, is the best approximation of B. That is, the image closer to B. This image is given in Figure 5 and it is defined as:

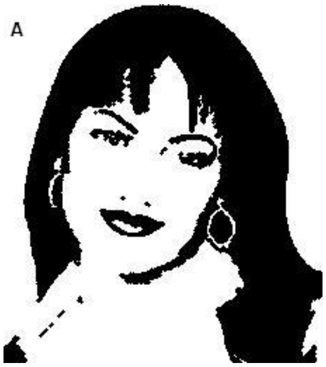


Fig. 5. Greatest solution

$$\begin{aligned} \varepsilon_S(B)(\overline{x}) &= (R_S)^{\forall}(B)(\overline{x}) \\ &= \inf\{I(R_S(\overline{x}, \overline{y}), B(\overline{y}) \mid \overline{y} \in X\} \\ &= \inf\{B((y_1, y_2)) \mid (y_1 - x_1) + (y_2 - x_2) \leq 1\} \\ &= \begin{cases} 0 & \text{if } \begin{array}{l} \text{there exists } (y_1, y_2) \in \mathbb{Z}^2 \text{ such that } B((y_1, y_2)) = 0 \\ \text{and } (y_1 - x_1) + (y_2 - x_2) \leq 1 \end{array} \\ 1 & \text{otherwise} \end{cases} \end{aligned}$$

By Theorem 3, the pair (A, B) is a property-oriented concept of the context $(\{0,1\}, \mathbb{Z}^2, \mathbb{Z}^2, R_S)$. However, in this case, the considered original image is not the greatest solution A_g but a solution less than it. This is given in Figure 6.

This is not a problem, but it shows that it is also interesting to compute the whole set of solutions. Theorem 4 provides an interesting mechanism to get all the solutions of Systems (7) and (8), although another more operative results will be studied in the future.

4 Conclusions and future work

In mathematical morphology, the usual procedure is, given a structuring image, to obtain the dilation and the erosion of an original image. This paper have

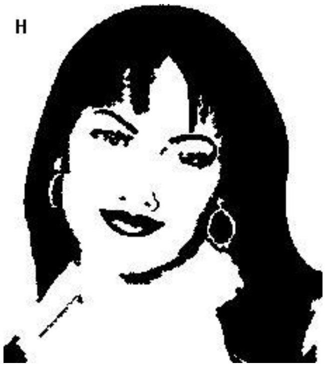


Fig. 6. Original image

studied the opposite problem, that is, given a fuzzy object $B\colon X \to L$ and a structuring object $S\colon X \to L$, find out the original object $A\colon X \to L$ such that B is the dilation of A, $\delta_S(A) = B$, or the erosion of A, $\varepsilon_S(A) = B$, and, if there exists, how to obtain it.

We have shown that this problem is associated with solving systems of fuzzy relation equations. Therefore, the results given in [12] and the relationship introduced in [1] have been used to obtain the original image or a good approximation. Moreover, we have introduced some results focus on searching the whole set of possible original images.

In the future, more properties and applications will be studied, for instance in object recognition.

References

1. C. Alcalde, A. Burusco, J. C. Díaz, R. Fuentes-González, and J. Medina-Moreno. Fuzzy property-oriented concept lattices in morphological image and signal processing. *Lecture Notes in Computer Science*, 7903:246–253, 2013.
2. C. Alcalde, A. Burusco, and R. Fuentes-González. An interpretation of the l-fuzzy concept analysis as a tool for the morphological image and signal processing. In *Proc. of CLA 2012*, pages 81–92, 2012.
3. E. Bartl, R. Bělohlávek, J. Konecny, and V. Vychodil. Isotone galois connections and concept lattices with hedges. In *4th International IEEE Conference "Intelligent Systems"*, pages 15.24–15.28, 2008.
4. R. Bělohlávek. Concept lattices and order in fuzzy logic. *Annals of Pure and Applied Logic*, 128:277–298, 2004.
5. I. Bloch. Duality vs. adjunction for fuzzy mathematical morphology and general form of fuzzy erosions and dilations. *Fuzzy Sets and Systems*, 160(0):1858–1867, 2009.
6. I. Bloch and H. Maître. Fuzzy mathematical morphologies: a comparative study. *Télécom Paris 94D001*, 1994.

7. P. Burillo, N. Frago, and R. Fuentes-González. Inclusion grade and fuzzy implication operators. *Mathware & Soft Computing*, VIII(0):31–46, 2001.
8. P. Burillo, R. Fuentes-González, and N. Frago. Inclusion grade and fuzzy implication operators. *Fuzzy Sets and Systems*, 114(0):417–429, 2000.
9. B. De Baets. Fuzzy morphology: a logical approach. In B. Ayyub and M. Gupta, editors, *Uncertainty Analysis, Engineering and Science: Fuzzy Logic, Statistics and neural network Approach*, pages 53–67. Kluwer Academic Publishers, 1997.
10. B. De Baets. Analytical solution methods for fuzzy relation equations. In D. Dubois and H. Prade, editors, *The Handbooks of Fuzzy Sets Series*, volume 1, pages 291–340. Kluwer, Dordrecht, 1999.
11. A. Di Nola, E. Sanchez, W. Pedrycz, and S. Sessa. *Fuzzy Relation Equations and Their Applications to Knowledge Engineering.* Kluwer Academic Publishers, Norwell, MA, USA, 1989.
12. J. C. Díaz and J. Medina. Solving systems of fuzzy relation equations by fuzzy property-oriented concepts. *Information Sciences*, 222:405–412, 2013.
13. G. Georgescu and A. Popescu. Non-dual fuzzy connections. *Arch. Math. Log.*, 43(8):1009–1039, 2004.
14. J. Goutsias and H. Heijmans. Fundamenta morphologicae mathematicae. *Fundamenta Informaticae*, 41(0):1–31, 2000.
15. H. Lai and D. Zhang. Concept lattices of fuzzy contexts: Formal concept analysis vs. rough set theory. *International Journal of Approximate Reasoning*, 50(5):695–707, 2009.
16. P. Maragos. Lattice image precessing: A unification of morphological and fuzzy algebric systems. *Journal of Mathematical Imaging and Vision*, 22(0):333–353, 2005.
17. M. Mas, M. Monserrat, and J. Torrens. S-implications and r-implications on a finite chain. *Kybernetika*, 40(1):3–20, 2004.
18. G. Matheron. *Random Sets and Integral Geometry.* Wiley, 1975.
19. J. Medina. Multi-adjoint property-oriented and object-oriented concept lattices. *Information Sciences*, 190:95–106, 2012.
20. I. Perfilieva and L. Nosková. System of fuzzy relation equations with inf-→ composition: Complete set of solutions. *Fuzzy Sets and Systems*, 159(17):2256–2271, 2008.
21. E. Sanchez. Resolution of composite fuzzy relation equations. *Information and Control*, 30(1):38–48, 1976.
22. J. Serra. *Image Analysis and Mathematical Morphology.* Academic Press, I (fourth printing 1990) and II (second printing 1992).

A Study on the Correspondence between FCA and $\mathcal{ELI}$ Ontologies

Melisachew Wudage Chekol, Mehwish Alam, and Amedeo Napoli

LORIA (INRIA, CNRS, and Université de Lorraine)
France
{melisachew.chekol,mehwish.alam,amedeo.napoli}@inria.fr

Abstract. The description logic $\mathcal{EL}$ has been used to support ontology design in various domains, and especially in biology and medicine. $\mathcal{EL}$ is known for its efficient reasoning and query answering capabilities. By contrast, ontology design and query answering can be supported and guided within an FCA framework. Accordingly, in this paper, we propose a formal transformation of $\mathcal{ELI}$ (an extension of $\mathcal{EL}$ with *inverse roles*) ontologies into an FCA framework, i.e. $K_{\mathcal{ELI}}$, and we provide a formal characterization of this transformation. Then we show that SPARQL query answering over $\mathcal{ELI}$ ontologies can be reduced to lattice query answering over $K_{\mathcal{ELI}}$ concept lattices. This simplifies the query answering task and shows that some basic semantic web tasks can be improved when considered from an FCA perspective.

1 Introduction

Relying on Semantic Web (SW) languages and principles, several ontologies have been created in various domains, especially, in biology and medicine. In addition to that, since the conception of linked data publishing principles, over 295 linked (open) datasets have been produced[1]. Querying these data is mainly done through the W3C recommended query language SPARQL[2].

In parallel, knowledge discovery in data represented by means of objects and their properties can be done using formal concept analysis (FCA) [9]. Concept lattices can reveal hidden relations within data and can be used for organizing and classifying data. A survey of the benefits of FCA to SW and vice versa has been proposed in [14]. As mentioned in that paper, a few of these benefits ranges from knowledge discovery, ontology completion, to computing subsumption hierarchy of least common subsumers. Additionally, studies in [7] and [12] are based on FCA for managing SW data while finite models of description logics (as $\mathcal{EL}$) are explored in [3,4]. All these studies propose methods to use FCA in the analysis of SW data. Nevertheless, none of them offer a precise way of representing SW data within a formal context. We deem it necessary to provide mathematically founded methods to formalize the representation and the analysis of SW data.

---

[1] `http://linkeddata.org/`

[2] `http://www.w3.org/TR/sparql11-query/`

© paper author(s), 2013. Published in Manuel Ojeda-Aciego, Jan Outrata (Eds.): CLA 2013, pp. 237–248, ISBN 978–2–7466–6566–8, Laboratory L3i, University of La Rochelle, 2013. Copying permitted only for private and academic purposes.

In this work, we focus particularly on $\mathcal{ELI}$ (an extension of $\mathcal{EL}$ with inverse roles) ontologies. $\mathcal{EL}$ is one of OWL 2 profiles (OWL 2 $\mathcal{EL}$). In fact, OWL 2 $\mathcal{EL}$ is used mainly for designing large biomedical ontologies such as SNOMED-CT[3], and the NCI thesaurus[4]. A common feature of these ontologies is that they possess large concept hierarchies that can be queried with SPARQL. Answering SPARQL queries is done by binding variables of the query into terms of the queried ontology. However, including inferred data in the query answers requires either a reasoner to infer all implicit data or query rewriting using regular expression patterns (that enable navigation in a hierarchy) [10]. The latter obliges the user to know the nuts and bolts of SPARQL. To overcome these difficulties, we reduce SPARQL query answering in $\mathcal{ELI}$ ontologies into query answering in concept lattices along with the transformation of the queried ontology into a formal context. Querying a concept lattice appears to be a less complex task than using SPARQ. Further, the lattice organization, i.e., partial ordering, can help understanding the relations between data and visualization of SW data.

Overall, in this paper, we work towards (i) a formal characterization of the translation of ontologies into a formal context, (ii) minimizing the difficulty of SPARQL query answering over ontologies into LQL (Lattice Query Language) query answering over concept lattices, and finally (iii) providing organization of SPARQL query answers with the use of concept lattices.

Outline: after presenting the basics of $\mathcal{ELI}$, SPARQL and FCA (§2), we show how to transform $\mathcal{ELI}$ ontologies into formal contexts (§3). We then present a query language for concept lattices called LQL (§4). Therefore, SPARQL query answering over $\mathcal{ELI}$ ontologies can be reduced to LQL query answering over $K_{\mathcal{ELI}}$ concept lattices (§5). Finally, we present the related works (§6) along with a summary of concluding remarks (§7).

2 Preliminaries

In this section, we provide a very brief and intuitive introduction of the description logic $\mathcal{ELI}$ and FCA. For a detailed discussion, we refer the readers to [11,2,9,6].

In $\mathcal{ELI}$, classes are inductively defined from a set $\mathrm{N_C}$ of *class* names, a set $\mathrm{N_R}$ of *role* names, and a set $\mathrm{N_I}$ of *individual* names ($\mathrm{N_C}$, $\mathrm{N_R}$, and $\mathrm{N_I}$ are finite), using the constructors: $\top$, $C \sqcap D$, and $\exists R.C$ are classes. Where C and D refer to classes, R refers to a role name or its inverse R^-, and in the assertion $C(a)$, a refers to an individual. In this paper, we consider $\exists R.C$ classes with $C \in \mathrm{N_C}$, i.e, C is an atomic concept in the expression of $\exists R.C$. The TBox of a $\mathcal{EL}$ knowledge base contains a set of class inclusion axioms such as $C \sqsubseteq D$. The ABox contains class and role assertions: $C(a)$ and $R(a, b)$. The semantics of $\mathcal{ELI}$ is broadly discussed in [2]. The semantics of $\mathcal{ELI}$-classes is defined in terms of an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$. The *domain* $\Delta^{\mathcal{I}}$ is a non-empty set of

---

[3] `http://www.ihtsdo.org/snomed-ct/`

[4] `http://ncit.nci.nih.gov/`

individuals and the *the interpretation function* $\cdot^{\mathcal{I}}$ maps each class name $A \in \mathrm{N_C}$ to a subset $C^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, each role name $R \in \mathrm{N_R}$ to a binary relation $R^{\mathcal{I}}$ on $\Delta^{\mathcal{I}}$, and each individual name $a \in \mathcal{N_I}$ to an individual $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. The extension of $\cdot^{\mathcal{I}}$ to arbitrary class descriptions is defined inductively [2].

SPARQL is a W3C recommended query language [13] based on simple graph patterns. It allows variables to be bound to components in the queried graph. In addition, operators akin to relational joins, unions, left outer joins, selections, and projections can be combined to build more expressive queries. Queries are formed from query patterns which in turn are defined inductively from *path patterns*, i.e., tuple $t \in$ UBV $\times\, e \,\times$ UBLV, with V a set of variables disjoint from UBL (URIs, Blank nodes and Literals – are used to identify values such as strings, integers and dates.), and e is regular path expression. Path patterns grouped together using operators AND (**.**) and UNION form *query patterns.*

Definition 1. *A query pattern q is inductively defined as:*

$$q ::= \ \mathrm{UBV} \times e \times \mathrm{UBLV} \mid q_1 \ . \ q_2 \mid \{q_1\} \ \mathit{UNION} \ \{q_2\}$$
$$e ::= \ \epsilon \mid U \mid V \mid e_1/e_2 \mid e_1 \mid e_2 \mid e^+ \mid e^*$$

A SPARQL *SELECT* query can be formed according to the following syntax: SELECT W FROM $\mathcal{O}$ WHERE $\{q\}$. The FROM clause identifies the queried ontology $\mathcal{O}$ on which the query will be evaluated, WHERE contains a query pattern q that the query answers should satisfy and SELECT singles out the answer variables $W \in V$ from the query pattern. For this work, we consider only AND and UNION SPARQL queries.

A formal context represents data using objects, attributes, and their relationships. Formally, it is a triple $K = (G, M, I)$ where G a set of objects, M a set of attributes, and $I \subseteq G \times M$ is a relation. A derivation operator ($'$) is used to compute *formal concepts* of a context. Given a set of objects, the operator derives common attributes of these objects and vice versa. A set of formal concepts ordered with the set inclusion relation form a *concept lattice* [6].

In the next section, we show the transformation of $\mathcal{ELI}$ ontologies into formal contexts.

3 Transforming $\mathcal{ELI}$ Ontologies into Formal Contexts

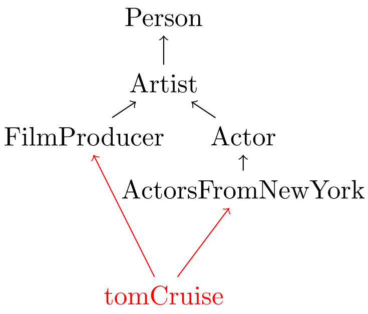
In the following, we introduce some terms and notions that we use. *Materialization* (closure) refers to computing the deductive closure of an ontology (alternatively, making all implicitly stored data explicit by using inference) [15]. *Ontology completion [5]*–refers to computing the closure of the ontology and adding additional instances by following class inclusions in the TBox (for instance, if Actor is a subclass of Artist and the instance Tom is an Actor, add another instance who is not an Actor but is an Artist. In this case, if an instance is not known, one can use anonymous resource to identify the unknown instance). *Loss of semantics*–the transformation of an ontology into a formal context results in loss of semantics if the context mixes TBox (schema axioms) and ABox (instance) data and if the concept lattice obtained from the formal context does not maintain the class hierarchy. Before presenting how a $\mathcal{ELI}$ ontology can be transformed into a formal context, we motivate our approach with an example.

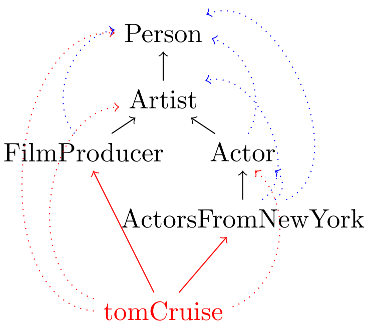
3.1 Motivation

Example 1. Consider the following $\mathcal{ELI}$ ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$:

$$\begin{aligned} \mathcal{T} = \{&\texttt{ActorsFromNewYork} \sqsubseteq \texttt{Actor}, \texttt{FilmProducer} \sqsubseteq \texttt{Artist}, \\ &\texttt{Actor} \sqsubseteq \texttt{Artist}, \texttt{Artist} \sqsubseteq \texttt{Person}\} \\ \mathcal{A} = \{&\texttt{tomCruise}^{\mathcal{I}} \in \texttt{ActorsFromNewYork}^{\mathcal{I}}\} \end{aligned}$$

In order to compare graphical representations of DL ontologies and their corresponding concept lattices, we represent $\mathcal{O}$ and its respective materialization $\mathcal{O}'$ as graphs as shown below:





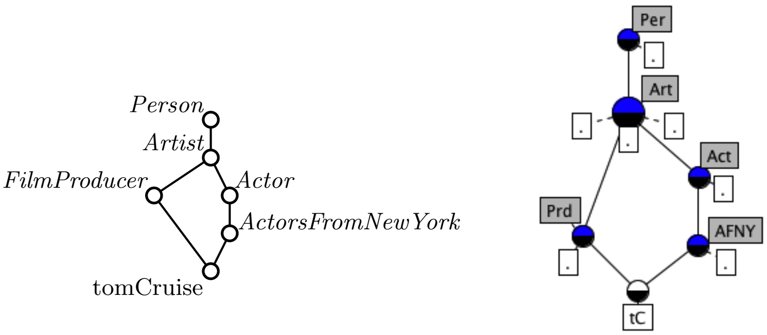
In the graphs, dotted edges denote inferred instance and class subsumption relations.

Starting with Example 1, one can ask whether it is possible to obtain a formal context from the ontology $\mathcal{O}$ while maintaining its semantics. The problem here is that DLs and FCA work on different assumptions, i.e, while DL languages are based on the *open world assumption (OWA)*, FCA relies on the *closed world assumption (CWA)*. The former permits to specify only known data whereas the later demands all data should be explicitly specified. To slightly close the gap between these two worlds:

- one can generate the formal context from the closure of the ontology. However, this approach fails when it is not possible to compute the closure of the ontology as this is the case for ontologies created from a DL language equipped with negation and disjunction constructs[5], and
- before transforming the ontology into a formal context, complete the ontology. A drawback of the second approach is that it adds unnecessary data, consequently, giving unwanted results when querying.

---

[5] `http://www.w3.org/TR/owl2-primer/`



(a) Target lattice associated with the ontology in Example 1.

(b) Lattice associated with the context in Example 2.

Fig. 1: tomCruise (tC) and (.) are objects and the rest are attributes.

To this end, our main objective is to come up with an approach which transforms an ontology into a formal context while maintaining the semantics. Accordingly, a formal context corresponding to the ontology in Example 1 has an associated lattice that looks like the one in Figure 1a. From this onwards, when we speak of this lattice, we refer to it as the *target* lattice. The target lattice *maintains the semantics* because: the class hierarchy of the ontology (TBox) is the same as that of the lattice, and the instance (ABox) and schema (TBox) part of the ontology are treated separately as discussed in Section 3.2.

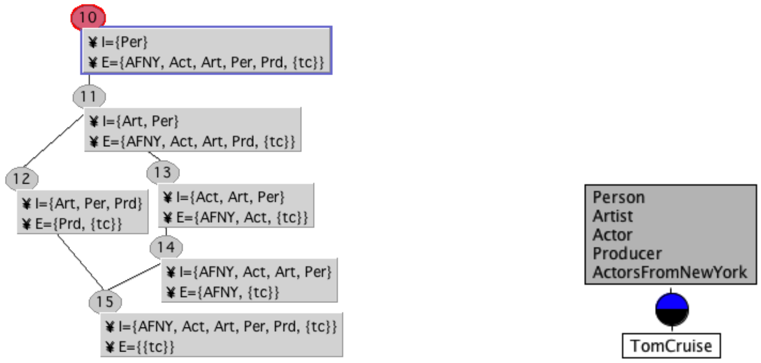
In the following, we provide various formal contexts associated with the ontology in Example 1. For the sake of readability, we shorten concept and individual names as: ActorsFromNewYork (AFNY), FilmProducer (Prd), Actor (Act), Artist (Art), Person (Per), and tomCruise (tC). Consider the following transformations:

Naive approach: materialized ABox

	AFNY	Prd	Act	Art	Per
tC	x	x	x	x	x

This formal context is obtained from the materialized ABox of the ontology. It does not include subclass relations as they can be acquired using *attribute exploration* [9]. But unfortunately, the resulting lattice considers all the attributes to be equivalent, implying loss of semantics, as it can be seen from the lattice in Figure 2b.

Direct approach: materialized ABox and TBox



(a) (b)

Fig. 2: Concept lattice associated with the ontology in Example 1.

	{tC}	AFNY	Prd	Act	Art	Per
{tC}	x	x	x	x	x	x
AFNY		x		x	x	x
Prd			x		x	x
Act				x	x	x
Art					x	x
Per						x

This formal context is produced by taking all the subclass hierarchy of all atomic concepts C and all nominal concepts $\{a\}$ for all individuals a. Formally, a formal context is constructed using: (i) $a^{\mathcal{I}} \in C^{\mathcal{I}}$ into $\{a\}, C \in G, M$, and $(\{a\},\{a\}),(C,C),(\{a\},C) \in I$, and (ii) $C \sqsubseteq D$ into $C, D \in G, M$, and $(C,D),(C,C),(D,D) \in I$. The context is a transformation of the materialized ontology (both the closures of the ABox and TBox are computed as depicted in the right-hand graph of Example 1). The concept lattice of this formal context is shown in Figure 2a. As it can be seen, it does not maintain the semantics because the concept hierarchy is different from that of our target lattice (in Figure 1a). In other words, the concept hierarchy of the concept lattice is different from that of the ontology (in Example 1). Everything is mixed: attributes are also objects and vice versa. Obviously, it is possible to find several other ways of transforming an ontology into a formal context. To avoid any semantic loss, we propose an another approach, where we separately manage the transformation of ABox and TBox assertions. In FCA, attribute exploration is used to discover implicit knowledge. In that, given a concept lattice, a domain expert is asked a series of questions to produce implications that correspond to DL like inclusion axioms. Ontologies contain instance and schema data, where the latter is similar to implications of concept lattices. Hence, when transforming, individuals in the ABox to become objects and concept names to become attributes, besides, assertions in the ABox are transformed into relations. Additionally, class inclusions of the TBox become background implications. The overall transformation procedure leads to a formal context with respect to existing knowledge (this is also known as *background implications* according to [8]). This procedure is formally described in definition 2.

3.2 Proposal

To transform a $\mathcal{ELI}$ knowledge base KB $= \langle \mathcal{T}, \mathcal{A} \rangle$ into a formal context $K = (G, M, I)$, the schema axioms in the TBox become background implications $\mathcal{L}((G, M, I))$ and the ABox assertions become objects, attributes and relations. To elaborate, in K, individuals in the ABox constitute objects G, class names in the ABox and TBox yield attributes in M, and ABox assertions create relations between objects and attributes $I \subseteq G \times M$. Here, we consider acyclic TBoxes so as to avoid class names becoming objects in a context.

Definition 2 (Transforming $\mathcal{ELI}$ Ontologies into Formal Contexts). *We define the transformation of* KB $= \langle \mathcal{T}, \mathcal{A} \rangle$ *into a formal context* (G, M, I) *thanks to a transformation function* σ *as follows:*

- *An axiom* $C \sqsubseteq D$ *in* $\mathcal{T}$ *corresponds to an implication in* $\mathcal{L}((G, M, I))$*, i.e., the set of implications based on* (G, M, I)*:* $C \sqsubseteq D \longmapsto C \rightarrow D \in \mathcal{L}((G, M, I))$.
- *Concept expressions* C *(class name),* $\exists R.C$*, and* $\exists R^{-}.C$*, correspond respectively to attributes* C*,* $\exists R.C$*, and* $\exists R^{-}.C$ *in* M.
- *An individual* a *in* $\mathcal{A}$ *corresponds to an object* a *in* G.
- *When* a *is an instance of* C *resp.* $\exists R.C$*,* $\exists R^{-}.C$*, then* $(a, C) \in I$ *resp.* $(a, \exists R.C) \in I$*,* $(a, \exists R^{-}.C) \in I$.
- *When* a *is related to* b *through* R*, then* $(a, \exists R.\top) \in I$ *and* $(b, \exists R^{-}.\top) \in I$.

Example 2. The translation of the ontology in Example 1 into a formal context K and its background implications $\mathcal{L}$ are shown below:

K	AFNY	Prd	Act	Art	Per
tC	x				

$$\mathcal{L} = \{\ \text{AFNY} \rightarrow \text{Act},\ \text{Prd} \rightarrow \text{Art},\ \text{Act} \rightarrow \text{Art},\ \text{Art} \rightarrow \text{Per}\ \}$$

Construction of concept lattices: there are several algorithms that can compute concept lattices associated with a formal context. Some of these are discussed in the literature [9,6] and have also been implemented. They work on an empty implication base. Thus, most are not suitable for contexts with background implications. In [8], the author provides an algorithm for attribute exploration with background implications. This technique can be employed for our purpose. As a result, the concept lattice associated with the formal context and background implications of Example 2 is depicted in Figure 1b.

Next we show that concept lattices associated with $\mathcal{ELI}$ ontologies can be queried by LQL – lattice query language.

4 Querying Concept Lattice

SPARQL query answering over $\mathcal{ELI}$ ontologies can be considered as lattice query answering over $K_{\mathcal{ELI}}$ concept lattices. To do this, we need to introduce a query language for concept lattices. Each node in a lattice can be seen as a query formed by a conjunction of: a concept intent and a concept extent. Intuitively, querying concept lattices amounts to fetching the objects given a set of attributes as query constants, alternatively, fetching the attributes given a set of objects as query constants or terms. Query terms can be connected using the logical operators: AND and OR to form a complex term. A term is either a set of objects called *object term* (OT) or a set of attributes called *attribute term* (AT).

Definition 3 (Object and Attribute Terms). *Given a formal context $K = (G, M, I)$, an object term (OT) and an attribute term (AT) are defined inductively as:*

$$\mathrm{OT} = \{g\} \mid \mathrm{OT}_1 \text{ AND } \mathrm{OT}_2 \mid \mathrm{OT}_1 \text{ OR } \mathrm{OT}_2, \ \textit{where } g \in G$$
$$\mathrm{AT} = \{m\} \mid \mathrm{AT}_1 \text{ AND } \mathrm{AT}_2 \mid \mathrm{AT}_1 \text{ OR } \mathrm{AT}_2, \ \textit{where } m \in M$$

The expression OT_1 AND OT_2 denotes the greatest lower bound (GLB) in the concept lattice $\underline{\mathcal{B}}(G, M, I)$. The expression OT_1 OR OT_2 denotes the least upper bound (LUB) in $\underline{\mathcal{B}}(G, M, I)$. Dually, the expression AT_1 AND AT_2 denotes the GLB in the concept lattice $\underline{\mathcal{B}}(G, M, I)$ (keeping the orientation of $\underline{\mathcal{B}}(G, M, I)$ based on the extents). Finally, the expression AT_1 OR AT_2 denotes the LUB in $\underline{\mathcal{B}}(G, M, I)$.

Based on the definitions of object and attribute terms, we introduce LQL queries. In this paper, we do not address the problem of negation in the query (and thus set difference).

Definition 4 (LQL - Lattice Query Language). *Given an object term* OT*, an attribute term* AT*, and variables $x, y \in V$ where V is a finite set of variables, an LQL query can take the following forms:*

$$q(y) = (\mathrm{OT}, y); \ q(x) = (x, \mathrm{AT}); \ q() = (OT, AT)$$

$q(y), q(x)$, and $q()$ do not necessarily correspond to formal concepts, only when OT and AT are closed sets. If OT is a closed set in $q(y) = (\mathrm{OT}, y)$, then y corresponds to the intent associated with OT. The same thing happens with x when AT is a closed set in $q(x) = (x, \mathrm{AT})$. For evaluating x and y in every possible case we do the following:

- if $\mathrm{OT} = \{g\}$, then $y = \{g\}'$, i.e., all attributes that are associated with the object g.
- if $\mathrm{OT} = \{g_1\}$ AND $\{g_2\}$, then $y = \{g_1, g_2\}'$
- if $\mathrm{OT} = \{g_1\}$ OR $\{g_2\}$, then $y = \{g_1\}' \cup \{g_2\}'$

Similarly, the evaluation of $q(x) = (x, \mathrm{AT})$ is given as follows:

- if AT $= \{m\}$, then $x = \{m\}'$, i.e., all objects that are associated with the attribute m.
- if AT $= \{m_1\}$ AND $\{m_2\}$, then $x = \{m_1, m_2\}'$
- if AT $= \{m_1\}$ OR $\{m_2\}$, then $x = \{m_1\}' \cup \{m_2\}'$

Finally, the evaluation of $q() = (\mathrm{OT}, \mathrm{AT})$ is:

- *true* if $\mathrm{OT} = \mathrm{AT}'$ or $\mathrm{AT} = \mathrm{OT}'$ and *false* otherwise.

Example 3. Let us consider querying the concept lattice shown in Figure 3.

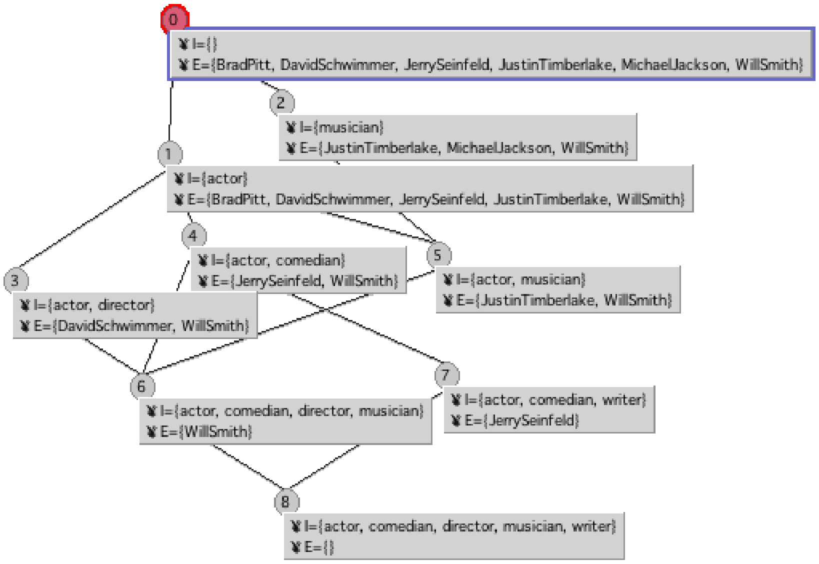

0
I={}
E={BradPitt, DavidSchwimmer, JerrySeinfeld, JustinTimberlake, MichaelJackson, WillSmith}
2
I={musician}
E={JustinTimberlake, MichaelJackson, WillSmith}
1
I={actor}
E={BradPitt, DavidSchwimmer, JerrySeinfeld, JustinTimberlake, WillSmith}
4
I={actor, comedian}
E={JerrySeinfeld, WillSmith}
5
I={actor, musician}
E={JustinTimberlake, WillSmith}
3
I={actor, director}
E={DavidSchwimmer, WillSmith}
7
I={actor, comedian, writer}
E={JerrySeinfeld}
6
I={actor, comedian, director, musician}
E={WillSmith}
8
I={actor, comedian, director, musician, writer}
E={}

Fig. 3: A concept lattice representing artists professions.

- For $q_1(x) = (x, \{actor\}$ AND $\{comedian\}$ AND $\{writer\})$, we have $x = \{JerrySeinfeld\}$.
- $q_2(x) = (x, \{writer\}$ OR $\{director\})$, we have $x = \{DavidSchwimmer, WillSmith, JerrySeinfeld\}$.
- $q_3(y) = (\{JustinTimberlake\}$ AND $\{WillSmith\}, y)$, we have $y = \{actor, musician\}$.
- $q_4(y) = (\{DavidSchwimmer\}$ OR $\{JustinTimberlake\}, y)$, we have $y = \{actor, director, musician\}$.

The complexity of answering LQL queries is *polynomial* in the size of the formal context, i.e., $\mathcal{O}|(G, M, I)|$. The advantage of LQL over SPARQL is that, it allows to compute the least upper bound and greatest lower bound of query answers. We now present one important part of this work which is reducing SPARQL query answering over $\mathcal{ELI}$ ontologies into LQL query answering over $K_{\mathcal{ELI}}$ concept lattices.

5 SPARQL query answering over ontologies vs LQL query answering over concept lattices

Recently, SPARQL has been extended with different entailment regimes and regular path expressions[6]. The semantics of SPARQL relies on the definition of basic graph pattern matching that is built on top of simple entailment [10]. However, it may be desirable to use SPARQL to query triples entailed from subclass, subproperty, range, domain, and other relations which can be represented using DL schema languages such as $\mathcal{ELI}$. The SPARQL specification defines the results of queries based on simple entailment. The specification also presents a general parametrized definition of graph pattern matching that can be expanded to other entailments beyond simple entailment. Query answering under an entailment regime can be achieved via: (1) materialization (computing the deductive closure of the queried graph), (2) rewriting the queries using the schema, and (3) hybrid (combining materialization and query rewriting) [10].

Example 4. Let us consider the evaluation of the SPARQL query Q on the ontology $\mathcal{O}$ and $\mathcal{O}'$ of Example 1. Q = select all those who are artists.

```
SELECT  ?x WHERE {?x a Artist.}
```

Under simple entailment evaluation of a SPARQL query, the answers of Q over $\mathcal{O}$ is empty, i.e., $Q(\mathcal{O}) = \emptyset$. For the reason that, simple entailment is based on graph matching which requires the variable $?x$ in the query to be bound with a term in the graph. Since there is no term where it can be bound to, the result is empty. However, under higher entailment regimes (such as the RDFS entailment [10]) the result of Q is non-empty because inferred instances obtained through reasoning are taken into account for computing the answers. To get a non-empty answers for the above query, one can use one of the following approaches:

1. *Materialization:* involves all implicit data to be computed before the evaluation of the query. This can be done by using a DL reasoner. Consequently, in Example 1, the materialization of $\mathcal{O}$ is $\mathcal{O}'$. Thus, the evaluation of Q over $\mathcal{O}$ is $Q'(\mathcal{O}) = \{\text{tomCruise}\}$.
2. *Query rewriting:* is the task of converting a SPARQL query into one that involves schema axioms. It can be done using SPARQL property paths (a.k.a. regular path expressions). For instance, the above query can be rewritten as:

   ```
   SELECT  ?x WHERE {?x a/⊑* Artist.}
   ```

   This query Q' selects all instances of Artist and that of its subclasses by navigating through the subclass relation ($\sqsubseteq^*$). The rewriting can be evaluated over $\mathcal{O}$ to obtain $Q'(\mathcal{O}) = \{\text{tomCruise}\}$.

In summary, materialization requires a reasoner to expand the knowledge base, the complexity of this task depends on the type of the schema language. On the other hand, query rewriting requires modifying query patterns using SPARQL

---

[6] `http://www.w3.org/TR/sparql11-query/`

property paths. This also results in a further jump in the complexity of query answering.

As described above, unlike SPARQL query answering over ontologies, query answering over a concept lattice is relatively easier. Due to the fact that once the concept lattice is obtained from the ontology, LQL can be used to query the lattice. Consequently, alleviating those expensive tasks. The above SPARQL query can be converted into an LQL query as: $q(x) = (x, Artist)$. The evaluation of this query over a concept lattice obtained from $\mathcal{O}$ is as expected $Q'(\mathcal{O}) = \{\text{tomCruise}\}$.

The complexity of SPARQL query answering over $\mathcal{ELI}$ ontologies is larger than that of LQL query answering over $K_{\mathcal{ELI}}$ concept lattices. Since, the expressive power of SPARQL is superior than that of LQL. For $\mathcal{ELI}$ ontologies a query language like LQL is sufficient to retrieve individuals (or objects) and classes (or attributes).

6 Related work

To date, several studies have been carried out to assess the relevance and benefits of FCA for DL [5,3,4,14,7,12]. Notably, the work in [14] presents a survey on the advantageous of FCA for DL ontologies. Accordingly, some of the benefits that FCA can bring to the DL world include: knowledge discovery, extended subsumption hierarchy (of conjunctions of concepts) [1], subsumption hierarchy of least common subsumers, exploring finite models [3,4], role assertion analysis, supporting bottom-up construction and completion of ontologies. Since the survey, other studies, [7] and [12], have carried out experiments to characterize and analyse SW data using FCA tools. The former provides an entry point to a linked data using questions in a way that can be navigated. It gives a translation of an RDF graph into a formal context where the subject of an RDF triple becomes the object, a composition of the predicate and object of the triple becomes an attribute. The latter obliges the user to specify objects and attributes of a context. With that, it creates SPARQL queries to extract content from linked data in order to populate the formal context. Despite the fact that all these works have employed FCA techniques, to the best of our knowledge, none of them provide a formal and precise translation of ontologies into a formal context as we did here.

7 Conclusion

In this work, firstly, we have proposed a formal transformation of $\mathcal{ELI}$ ontologies into formal contexts. This enables to benefit from some advantages that FCA may offer to the DL world. Then we have shown that SPARQL query answering over $\mathcal{ELI}$ ontologies can be considered as lattice query answering over $K_{\mathcal{ELI}}$ concept lattices. This alleviates some reasoning and query rewriting tasks that are required for SPARQL query answering.

Moreover, even if there already exist substantial work relating DL, semantic web and FCA, there remains a lot of research work to be carried out. Such a research work is concerned with the correspondence between concept lattices from FCA and DL-based class hierarchies, query answering and information retrieval, and scalability as well. In addition, as FCA could benefit from DL-based reasoning capabilities, semantic web and DL-driven applications can take advantage of FCA-based ontology design, data analysis and knowledge discovery capabilities of FCA.

In the future, we plan to extend and experiment with the proposed approach. We will investigate how well it scales, given the size of ontologies.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2007), iSBN 9780511717383
2. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope. In: IJCAI. vol. 5, pp. 364–369 (2005)
3. Baader, F., Distel, F.: A finite basis for the set of el-implications holding in a finite model. In: ICFCA. pp. 46–61. Springer (2008)
4. Baader, F., Distel, F.: Exploring finite models in the description logic EL gfp. In: ICFCA. pp. 146–161. Springer (2009)
5. Baader, F., Ganter, B., Sertkaya, B., Sattler, U.: Completing description logic knowledge bases using formal concept analysis. In: Proc. of IJCAI. vol. 7, pp. 230–235 (2007)
6. Carpineto, C., Romano, G.: Concept data analysis: Theory and applications. Wiley (2004)
7. d'Aquin, M., Motta, E.: Extracting relevant questions to an RDF dataset using formal concept analysis. In: Proceedings of the sixth international conference on Knowledge capture. pp. 121–128. ACM (2011)
8. Ganter, B.: Attribute exploration with background knowledge. Theoretical Computer Science 217(2), 215 – 233 (1999)
9. Ganter, B., Wille, R.: Formal Concept Analysis. Springer, Berlin (1999)
10. Glimm, B.: Using SPARQL with RDFS and OWL entailment. Reasoning Web. Semantic Technologies for the Web of Data pp. 137–201 (2011)
11. Hayes, P.: RDF semantics. W3C Recommendation (2004)
12. Kirchberg, M., Leonardi, E., Tan, Y.S., Link, S., Ko, R.K., Lee, B.S.: Formal concept discovery in semantic web data. In: ICFCA. pp. 164–179. Springer-Verlag (2012)
13. Prud'hommeaux, E., Seaborne, A.: SPARQL query language for RDF. W3C Rec. (2008)
14. Sertkaya, B.: A survey on how description logic ontologies benefit from FCA. In: CLA. vol. 672, pp. 2–21 (2010)
15. Ter Horst, H.: Completeness, decidability and complexity of entailment for RDF schema and a semantic extension involving the OWL vocabulary. Web Semantics: Science, Services and Agents on the World Wide Web 3(2-3), 79–115 (2005)

From Triadic FCA to Triclustering: Experimental Comparison of Some Triclustering Algorithms

Dmitry V. Gnatyshak, Dmitry I. Ignatov, and Sergei O. Kuznetsov

National Research University Higher School of Economics
`dignatov@hse.ru`
`http://www.hse.ru`

Abstract. In this paper we show the results of the experimental comparison of five triclustering algorithms on real-world and synthetic data wrt. resource efficiency and 4 quality measures. One of the algorithms, the OAC-triclustering based on prime operators, is presented first time in this paper. Interpretation of results for real-world datasets is provided.

Keywords: formal concept analysis, triclustering, triadic data, data mining

1 Introduction

Recently analysis of triadic data attracts more and more attention in Data Mining community [1,2,3,4,5]. One particular example is mining of so-called folksonomies, structures composed by three sets: users, objects and tags. One of the first attempts of such an analysis[1] was done within the framework of Formal Concept Analysis (FCA) [6], namely in Triadic Formal Concept Analysis (TCA) [7]. Triclustering methods, which are objects of our study, allow one to simultaneously discover three-component homogeneous groups in the considered three-sets. For example, these triclusters can be used for community detection [8] and recommender algorithms [9]. As a consequence, we would like to investigate peculiarities and reveal advantages and drawbacks of various triclustering approaches to choose the optimal one depending on the task.

In this paper we compare the following triclustering methods: object-attribute-condition triclustering (2 modifications including new one) [10,11], TriBox [2], spectral triclustering [11] and TRIAS algorithm [1]. Even though there are some recent efficient [12] and even more general algorithms than TRIAS [13], it is well-known to the FCA community and sometimes outperforms its competitors [12]. Moreover, the aim of the paper is not in comparison of different algorithms for triadic formal concepts generation, but it is rather in comparison of the original patterns with their various approximations (triclusters) in terms of the introduced quality measures. We also suggest investigating different matrix decomposition approaches, e.g. Boolean ones [3,4], in a further study; note that

© paper author(s), 2013. Published in Manuel Ojeda-Aciego, Jan Outrata (Eds.): CLA 2013, pp. 249–260, ISBN 978–2–7466–6566–8, Laboratory L3i, University of La Rochelle, 2013. Copying permitted only for private and academic purposes.

Boolean matrix factorization can be considered as an approach to the reduction of the number of the resulting triconcepts.

The rest of the paper is organised as follows. In section 2 we give main definitions and describe the triclustering methods selected for the comparison. Section 3 describes all the experiments and their results on real and synthetic data along with specially introduced quality measures. Section 4 concludes the paper and indicates some further research direction.

2 Triclustering models and methods

Object-attribute-condition triclustering (OAC-triclustering) is based on Formal Concept Analysis [6] and its triadic extension, Triadic Formal Concept Analysis (TCA) [7]. In this paper we consider 2 types of OAC-triclustering: box operator based (box OAC-triclustering) [10], and prime operator based (prime OAC-triclustering). The latter is introduced in this paper. The set of triclusters is generated one by one and complete enumeration strategy is used.

First, we recall some basic notions of Formal Concept Analysis (FCA) [6]. Let G and M be sets, called the set of objects and attributes, respectively, and let I be a relation $I \subseteq G \times M$: for $g \in G$, $m \in M$, gIm holds iff the object g has the attribute m. The triple $\mathbb{K} = (G, M, I)$ is called a *(formal) context*. If $A \subseteq G$, $B \subseteq M$ are arbitrary subsets, then the *Galois connection* is given by the following *derivation operators*:

$$\begin{aligned} A' &= \{m \in M \mid gIm \text{ for all } g \in A\}, \\ B' &= \{g \in G \mid gIm \text{ for all } m \in B\}. \end{aligned} \tag{1}$$

The pair (A, B), where $A \subseteq G$, $B \subseteq M$, $A' = B$, and $B' = A$ is called a *(formal) concept (of the context K)* with *extent* A and *intent* B (in this case we have also $A'' = A$ and $B'' = B$).

The concepts, ordered by $(A_1, B_1) \geq (A_2, B_2) \iff A_1 \supseteq A_2$ form a complete lattice, called *the concept lattice* $\underline{\mathfrak{B}}(G, M, I)$.

2.1 Object-attribute-condition triclustering

Here let us define the box operators and describe **box OAC-triclustering**. We use a slightly different introduction of the main TCA notions because of their further technical usage.

Let $\mathbb{K} = (G, M, B, I)$ be a triadic context, where G, M, and B are sets, and I is a ternary relation: $I \subseteq G \times M \times B$. In addition to set of objects, G, and set of attributes, M, we have B, a set of conditions.

Derivation (prime) operators for a triple $(\widetilde{g}, \widetilde{m}, \widetilde{b}) \in I$ from triadic context $\mathbb{K}$ can be defined as follows:

$$\widetilde{g}' := \{\, (m, b) \mid (\widetilde{g}, m, b) \in I\} \tag{2}$$

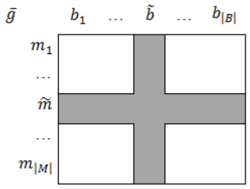


Fig. 1. $\overline{g}$ addition condition

$$(\widetilde{g}, \widetilde{m})' := \{\, b \,|\, (\widetilde{g}, \widetilde{m}, b) \in I \} \tag{3}$$

$\widetilde{m}'$, $\widetilde{b}'$, $(\widetilde{g}, \widetilde{b})'$, $(\widetilde{m}, \widetilde{b})'$ prime operators can be defined the same way.

Now for a triple $(\widetilde{g}, \widetilde{m}, \widetilde{b}) \in I$ let us define box operator $\widetilde{g}^{\Box}$ ($\widetilde{m}^{\Box}$ and $\widetilde{b}^{\Box}$ are introduced in the same way):

$$\widetilde{g}^{\Box} := \{\, g \,|\, \exists m (g, m) \in \widetilde{b}' \vee \exists b (g, b) \in \widetilde{m}' \} \tag{4}$$

Definition 1. *Suppose* $\mathbb{K} = (G, M, B, I)$ *is a triadic context. For a triple* $(g, m, b) \in I$ *a triple* $T = (g^{\Box}, m^{\Box}, b^{\Box})$ *is called a* box operator based OAC-tricluster. *Traditionally, its components are respectively called* extent, intent, and modus.

The density of a tricluster $T = (X, Y, Z)$ is defined as the fraction of all triples of I in $X \times Y \times Z$:

$$\rho(T) := \frac{|I \cap (X \times Y \times Z)|}{|X||Y||Z|} \tag{5}$$

Definition 2. *The tricluster* T *is called* dense *iff its density is not less than some predefined threshold, i.e.* $\rho(T) \geq \rho_{min}$.

Let us elaborate on the structure of box operator based triclusters. Suppose $\mathbb{K} = (G, M, B, I)$ is a triadic context, and the triple $(\widetilde{g}, \widetilde{m}, \widetilde{b}) \in I$ is considered. Then object $\overline{g}$ will be added to $\widetilde{g}^{\Box}$ iff $\{(\overline{g}, \widetilde{m}, b) \,|\, b \in B \wedge (\overline{g}, \widetilde{m}, b) \in I\} \neq \emptyset \vee \{(\overline{g}, m, \widetilde{b}) \,|\, m \in M \wedge (\overline{g}, m, \widetilde{b}) \in I\} \neq \emptyset$. It is clear that this condition is equivalent to the one in eq. (4), and can be easily illustrated (Fig. 1): if at list one of the elements from "grey" cells is an element of I, then $\overline{g}$ is added to $\widetilde{g}^{\Box}$.

The idea of box OAC-triclustering is to enumerate all triples of the ternary relation I for a context $\mathbb{K}$ generating a box operator based tricluster for each. If generated tricluster T was not added to the set of all triclusters $\mathcal{T}$ on previous steps, then T is added to $\mathcal{T}$. It is possible to implement hash-fuctions for triclusters in order to significantly optimize computational time by simplifying the comparison of triclusters. Also a minimal density threshold can be used.

Prime OAC-triclustering extends the biclustering method from [14] to the triadic case. It uses prime operators (eq. 3) to generate triclusters.

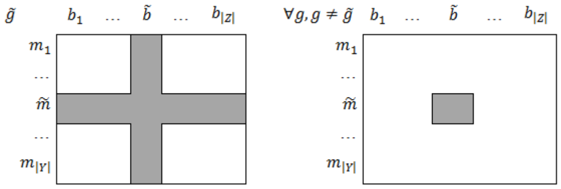

$\tilde{g}$
b_1 … $\tilde{b}$ … $b_{|Z|}$
m_1
…
$\tilde{m}$
…
$m_{|Y|}$
$\forall g, g \neq \tilde{g}$
b_1 … $\tilde{b}$ … $b_{|Z|}$
m_1
…
$\tilde{m}$
…
$m_{|Y|}$

Fig. 2. Prime operator based tricluster structure

Definition 3. *Suppose* $\mathbb{K} = (G, M, B, I)$ *is a triadic context. For a triple* $(g, m, b) \in I$ *a triple* $T = ((m, b)', (g, b)', (g, m)')$ *is called a* prime operator based OAC-tricluster. *Its components are called respectively* extent, intent, and modus.

Prime based OAC-triclusters are more dense than box operator based ones. Their structure is illustrated on Fig. 2: every element corresponding to the “grey” cell is an element of I. Thus, prime operator based OAC-triclusters in a three-dimensional matrix form contain a cross-like structure of ones.

It may also be useful to implement hash functions for triclusters.

Algorithm 1 Algorithm for prime OAC-triclustering.

```
Input: $\mathbb{K} = (G, M, B, I)$ — tricontext;
    $\rho_{min}$ — density threshold
Output: $\mathcal{T} = \{T = (X, Y, Z)\}$
$\mathcal{T} := \emptyset$
for all $(g, m)$: $g \in G, m \in M$ do
    $PrimesObjAttr[g, m] = (g, m)'$
end for
for all $(g, b)$: $g \in G, b \in B$ do
    $PrimesObjCond[g, b] = (g, b)'$
end for
for all $(m, b)$: $m \in M, b \in B$ do
    $PrimesAttrCond[m, b] = (m, b)'$
end for
for all $(g, m, b) \in I$ do
    $T = (PrimesAttrCond[m, b], PrimesObjCond[g, b], PrimesObjAttr[g, m])$
    $Tkey = hash(T)$
    if $Tkey \notin \mathcal{T}.keys \wedge \rho(T) \geq \rho_{min}$ then
        $\mathcal{T}[Tkey] := T$
    end if
end for
```

2.2 TriBox method

The TriBox method [2] implements an optimization approach in tricluster generation. Suppose $\mathbb{K} = (G, M, B, I)$ is a triadic context. The idea is to select some triple of I, take it for the initial tricluster, and then to modify its extent, intent, and modus so that they covered a significant part of the context while maintaining high density. Thus, TriBox aims at finding a set of triclusters $\mathcal{T} = \{T = (X, Y, Z)\}$ that maximize the criterion 6. The resulting triclusters compose locally optimal solution for the trade-off problem between the density and the volume of various possible triclusters.

$$f(T) = \rho(T)^2|X||Y||Z| \tag{6}$$

Once again, hash-functions for triclusters may be used for optimizations.

2.3 Spectral triclustering method

Spectral triclustering method [11] is based on the graph partition problem. The idea is to represent the given triadic context as a tripartite graph and then recursively divide it. To find an optimal partitioning spectral clustering uses the second minimal eigenvector of the Laplacian matrix.

Let us elaborate on this technique. Suppose $\mathbb{K} = (G, M, B, I)$ is a triadic context First we need to transform $\mathbb{K}$ into tripartite graph $\Gamma = \langle V, E \rangle$. Since I is a *ternary* relation it is only possible to represent $\mathbb{K}$ as a tripartite hypergraph without the loss of information. The following transformation technique is considered: $V := G \sqcup M \sqcup B$, for each triple $(g, m, b) \in I$ edges (g, m), (g, b) and (m, b) are added to E to form an undirected non-weighted graph. As the result some additional triples will be added to I after inverse transformation. Still it is clear that these triples will be added only in ,,dense“ areas of I thus possibly filling missing values and “optimizing” tricontext for methods aiming at finding formal concepts. Thus this technique is acceptable for solving the problem.

After the transformation Laplacian matrix is built for Γ:

$$L_{ij} = \begin{cases} degree(v_i), & \text{if } i = j \\ -1, & \text{if } i \neq j \text{ and } \exists \text{ edge } (v_i, v_j) \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where v_i is the i^{th} vertex of V.

The second minimal eigenvector of L is an optimal solution of the continuous variant of the optimal partition problem for Γ (finding the minimal set $\tilde{E} \subseteq E$ so that the graph $\tilde{\Gamma} = (V, E \setminus \tilde{E})$ is not connected). The sign of each component of this vector indicates one of the 2 new connected components. For convenience we find the optimal partition vector as the approximation of the obtained vector by setting its components to ±1 values.

In order to avoid partitioning of dangling vertices or small subgraphs the generalized eigenvalue problem must be considered ([11], Appendix I):

$$Lv = \lambda Dv \tag{8}$$

where D is a diagonal matrix containing vertices' degrees on the main diagonal.

Also, some minimum size constraint can be used to avoid too deep partitioning. Since spectral triclustering is not able to generate the same tricluster more than once, it is not necessary to use hash functions to speed up the calculations.

2.4 Trias method

Formally, TRIAS [1] is a method for finding triadic formal concepts that are closed 3-sets. Since triadic formal concepts can be interpreted as absolutely dense triclusters, this method was added to the study.

TRIAS is based on the NEXTCLOSURE algorithm that enumerates all formal concepts of the dyadic context in lexicographical order. In TRIAS this approach is extended to the triadic case and minimal support constraints are added (triclusters with too small extent, intent or modus are skipped).

As well as spectral triclustering, TRIAS is not able to generate the same tricluster more than once.

3 Experiments

3.1 Noise-tolerance.

In order to test noise-tolerance of the algorithms 26 triadic contexts have been generated. The initial context contains 30 objects, 30 attributes, 30 conditions, and 3 non-overlapping $10 \times 10 \times 10$ cuboids of ones on the main diagonal in its three-dimensional matrix form. Then this context has been sequentially noised by the inversions with the probability of an inversion of a triple varying from 0.1 to 0.5 with 0.1 interval (the latter context can be called uniform context, because probability of $(g, m, b) \in I$ is equal for every triple). There have been 5 such series of context. Table 1 contains the average number of triples and total density for these sets of contexts.

Table 1. Noised contexts.

Context	# triples	Density
$p = 0$	3000	0.1111
$p = 0.1$	5069.6	0.1873
$p = 0.2$	7169.4	0.2645
$p = 0.3$	9290.2	0.3440
$p = 0.4$	11412.8	0.4222
$p = 0.5$	13533.4	0.5032

The noise tolerance of an algorithm has been defined as the ability to build triclusters similar to initial cuboids. We used the Jaccard similarity coefficient to find the most similar tricluster t for the given cuboid c and their similarity. Total similarity has been defined as follows (C is a number of cuboids):

$$sim(c) = \frac{1}{C} \sum_{c=c_1}^{c_C} \max_{t=t_1,\ldots,t_T} \frac{|G_c \cap G_t|}{|G_c \cup G_t|} \frac{|M_c \cap M_t|}{|M_c \cup M_t|} \frac{|B_c \cap B_t|}{|B_c \cup B_t|} \tag{9}$$

Following size measure for spectral triclustering has been chosen:

$$Size(X, Y, Z) = \frac{|X| + |Y| + |Z|}{|G| + |M| + |B|} \tag{10}$$

s_{min} has been set to 0.34 to stop at the triclusters of correct size.

All of the methods have been implemented by authors and incorporated in a single triclustering toolbox in order to make the comparison more accurate. The toolbox has been implemented in C# using MS Visual Studio 2010/2012. All the experiments have been performed on Windows 7 SP1 x64 system equipped with an Intel Core i7-2600 @ 3.40GHz processor and 8 GB of RAM. AlgLib[1] library was used for eigenvalue decomposition.

The results of the experiments are represented on Figure 3. It is clear that every method has managed to successfully find initial cuboids, but the results quickly deteriorate for most of methods with the growth of inversion probability. TriBox has shown the best results as it tries to optimize the density-volume trade-off (which most probably is the best for the areas of the former cuboids for small error probability). Though prime OAC-triclustering has been also rather noise-tolerant, it generated significantly more triclusters (most likely the high number of triclusters is the reason for these results). All the other methods have been unable to provide significant results for noisy contexts. Moreover, as it was expected, no adequate triclusters were generated by any of the methods for the contexts with inversion probability 0.5.

3.2 Time, quantity, coverage, density and diversity.

The experiments on the computation time, triclusters count, coverage, density, and diversity were conducted on the following contexts (Table 2):

1. Uniform context $(\forall (g, m, b)\, P((g, m, b) \in I) = 0.1)$
2. Top 250 movies context from `www.imdb.com`, objects — movie titles, attributes — tags, conditions — genres
3. Random sample of 3000 of the first 100000 triples of the `www.bibsonomy.org` dataset, objects — users, attributes — tags, conditions – bookmark names

Parallel versions of OAC-triclustering algorithms and TriBox have also been implemented via parallelization of their outer loops and the computation time for them has been compared. Coverage and diversity measures were introduced as additional quality measures. *Coverage* is defined simply as a fraction of the triples of the context (alternatively, objects, attributes or conditions) included

---

[1] `http://www.alglib.net/`

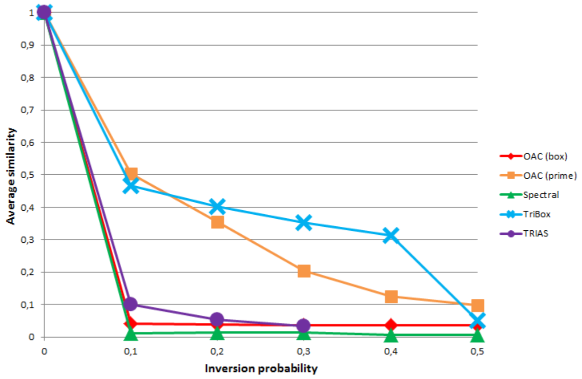


Fig. 3. Similarity for the noise-tolerance experiments

Table 2. Contexts for the experiments with 5 chosen evaluation measures.

Context	$\lvert G\rvert$	$\lvert M\rvert$	$\lvert B\rvert$	# triples	Density
Uniform	30	30	30	2660	0.0985
IMDB	250	795	22	3818	0.00087
BibSonomy	51	924	2844	3000	0.000022

in at least one of the triclusters of the resulting set. To define diversity we will use a binary function of 2 triclusters $intersect(\mathcal{T}_i, \mathcal{T}_j)$ that equals to 1 if both triclusters $\mathcal{T}_i$ and $\mathcal{T}_j$ have nonempty intersection of the sets of contained triples, and 0 otherwise.

It is also possible to define *intersect* for the sets of objects, attributes and conditions. For instance, $intersect_G(\mathcal{T}_i, \mathcal{T}_j)$ is equal to 1 if triclusters $\mathcal{T}_i$ and $\mathcal{T}_j$ have nonempty intersection of their extents, and 0 otherwise.

Now we can define diversity of the tricluster set $\mathcal{T}$:

$$diversity(\mathcal{T}) = 1 - \frac{\sum_j \sum_{i<j} intersect(\mathcal{T}_i, \mathcal{T}_j)}{\frac{|\mathcal{T}|(|\mathcal{T}|-1)}{2}} \qquad (11)$$

The diversity for the sets of objects, attributes or conditions is similarly defined.

Table 3 contains the results for the experiments. The following values for parameters were selected:

1. OAC-triclustering: $\rho_{min} = 0$
2. SpecTric: $s_{min} = 0$
3. Trias: $\tau_G = \tau_M = \tau_B = 0$

Table 3. Results of the experiments on the computation time (t), tricusters count (n), density (ρ), coverage (Cov), and diversity (Div).

Algorithm	t, ms	t_{par}, ms	n	ρ_{av}, %	Cov, %	Div, %	Div_G, %	Div_M, %	Div_B, %
	Uniform random context								
OAC (□)	407	**196**	73	9.88	**100.00**	0.00	0.00	0.00	0.00
OAC (′)	**312**	877	2659	32.23	**100.00**	92.51	60.07	59.80	59.45
SpecTric	277	-	**5**	8.74	8.84	**100.00**	**100.00**	**100.00**	**100.00**
TriBox	6218	1722	1011	74.00	96.02	97.42	66.25	79.53	84.80
Trias	29367	-	38356	**100.00**	**100.00**	99.99	99.93	4.07	3.51
	IMDB								
OAC (□)	2314	**1573**	1500	1.84	**100.00**	15.65	9.67	0.70	7.87
OAC (′)	**547**	2376	1274	53.85	**100.00**	96.55	94.56	92.14	28.52
SpecTric	98799	-	**21**	17.07	20.88	**100.00**	**100.00**	**100.00**	**100.00**
TriBox	197136	55079	328	91.65	98.90	98.89	98.46	95.21	30.94
Trias	102554	-	1956	**100.00**	**100.00**	99.89	99.69	52.52	26.18
	BibSonomy								
OAC (□)	19297	**6803**	398	4.16	**100.00**	79.59	67.28	42.83	79.54
OAC (′)	**13556**	9400	1289	94.66	**100.00**	99.74	88.58	99.51	99.53
SpecTric	5906563	-	**2**	50	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**
TriBox	> 24 hrs								
Trias	110554	-	1305	**100.00**	**100.00**	99.98	91.70	99.78	99.92

3.3 Results

Trias is one of the most time consuming algorithms compared in the paper along with TriBox and SpecTric for large contexts. Though the resulting triclusters (triconcepts) can easily be interpreted their number and small size make it impossible to understand the general structure of the context. Since all triconcepts have been generated and thus every triple has been covered, the coverage is equal to 1. Since the concepts are small general diversity is rather high. Still, the set diversity depends on the size of the corresponding set: the smaller the set — the greater the chance of intersection and the lower the diversity.

Examples of the triconcepts for the IMDB context:

1. {The Princess Bride (1987), Pirates of the Caribbean: The Curse of the Black Pearl (2003)}, {Pirate}, {Fantasy, Adventure}
2. {Platoon (1986), Letters from Iwo Jima (2006)}, {Battle}, {Drama, War}
3. {V for Vendetta (2005)}, {Fascist, Terrorist, Government, Secret Police , Fight}, {Action, Sci-Fi, Thriller}

SpecTric has displayed rather good computation time only for small contexts. Most of this time is used for the eigenvalue decomposition of Laplacian matrix. Thus we intend to test some alternative linear algebra libraries in the toolbox and compare the results as well. The resulting triclusters can be reasonably interpreted, though their average density is low. Their small number makes this method good for dividing the context into several non-overlapping parts. Also the diversities for SpecTric are always equal to 1 because the method generates partitions of the initial context. High diversity though leads to low coverage.

Examples of the triclusters for the IMDB context:

1. $\rho = 23.08\%$, {Alien (1979), The Shining (1980), The Thing (1982), The Exorcist (1973)}, {Spaceship, Egg, Parasite, Creature, Caretaker, Colorado, Actress, Blood, Helicopter, Scientist, Priest, Washington D.C., Faith}, {Horror}
2. $\rho = 2.09\%$, {The Shawshank Redemption (1994), The Godfather (1972), The Godfather: Part II (1974), . . . , Bonnie and Clyde (1967), Arsenic and Old Lace (1944)}, {Prison, Cuba, Business, 1920s, . . . , Texas, Cellar}, {Crime, Thriller }

TriBox in this study generates the best triclusters. The only drawback of this method is high computation time, though the use of the parallel version of TriBox can significantly lower it for multi-core processors. Average density of the resulting triclusters is rather high, they have good interpretability. Coverage and diversities are also high in most of the cases. The only exception is set diversity in the situation when some of the sets are small, just as for TRIAS.

Examples of the triclusters for the IMDB context:

1. 100%, {Million Dollar Baby (2004), Rocky (1976), Raging Bull (1980)}, {Boxer, Boxing}, {Drama, Sport}
2. 83.33%, {The Sixth Sense (1999), The Exorcist (1973), The Silence of the Lambs (1991)}, {Psychiatrist}, {Drama, Thriller}
3. 33.33%, {Platoon (1986), All Quiet on the Western Front (1930), Glory (1989), Apocalypse Now (1979), Lawrence of Arabia (1962), Saving Private Ryan (1998), Paths of Glory (1957), Full Metal Jacket (1987)}, {Army, General, Jungle, Vietnam, Soldier, Recruit}, {Drama, Action, War}

Box OAC-triclustering has been not that successful. Despite being rather fast (only OAC-triclustering based on prime operators and SpecTric for small contexts are faster) and having good parallel version the resulting triclusters are quite large, have relatively low density and many intersections. It leads to the high coverage (1 for $rho_{min} = 0$) and rather low diversities. Also these triclusters are difficult to interpret (unlike SpecTric's triclusters that also have large size and low density). In many cases extent sizes are small. Examples are given below:

1. 0.9%, {The Shawshank Redemption (1994), The Godfather (1972), Ladri di biciclette (1948), Unforgiven (1992), Batman Begins (2005), Die Hard (1988), . . . , The Green Mile (1999), Sin City (2005), The Sting (1973)}, {Prison, Murder, Cuba, FBI, Serial Killer, Agent, Psychiatrist,. . . , Window, Suspect, Organized Crime , Revenge, Explosion, Assassin, Widow}, {Crime, Drama, Sci-Fi, Fantasy, Thriller, Mystery}

2. 1.07%, {The Great Escape (1963), Star Wars: Episode VI - Return of the Jedi (1983), Jaws (1975), Batman Begins (2005), Blade Runner (1982), Die Hard (1988),. . . , Metropolis (1927), Sin City (2005), Rebecca (1940)}, {Prison, Murder, Cuba, FBI, Serial Killer, Agent, Psychiatrist,. . . , Shower, Alimony, Phoenix Arizona, Assassin, Widow}, {Drama, Thriller, War}

Prime OAC-triclustering showed rather good results. It is one of the fastest algorithms (though some additional optimizations specified for unparallel version made parallelization inefficient for small contexts). The number of triclusters is high, but they are easily interpreted. Once again for $\rho_{min} = 0$ coverage is equal to 1, but remains high for different ρ_{min}. At the same time diversities are also rather high. Examples of the triclusters for the IMDB context are given below:

1. 36%, {The Shawshank Redemption (1994), Cool Hand Luke (1967), American History X (1998), A Clockwork Orange (1971), The Green Mile (1999)}, {Prison, Murder, Friend, Shawshank, Banker}, {Crime, Drama}
2. 56, 67%, {The Godfather: Part II (1974), The Usual Suspects (1995)}, {Cuba, New York, Business, 1920s, 1950s}, {Crime, Drama, Thriller}
3. 60%, {Toy Story (1995), Toy Story 2 (1999)}, {Jealousy, Toy, Spaceman, Little Boy, Fight}, {Fantasy, Comedy, Animation, Family, Adventure}

4 Conclusion

We compared several different triclustering approaches and showed that there is no algorithm-winner with respect to the considered criteria. Also, we can conclude that either (dense) prime OAC-triclustering or TriBox is a good alternative for TCA approach because the total number of triclusers for real data example is drastically less than the number of triconcepts. The proposed algorithm has a good scalability on real-world data and shows acceptable noise-tolerance level. Probably investigated spectral hierarchical partitioning of ternary relations can be a good alternative of concept-based triclustering as a tool for fast exploratory (preliminary) analysis.

Our further work on triclustering will go in the following directions:

- developing a unified theoretical framework for n-clustering,
- mixing several constraint-based approaches to triclustering (e.g., mining dense triclusters first and then frequent tri-sets in them),
- finding better approaches for estimating tricluster's density,
- taking into account the nature of real-world data for optimization (their sparsity, value distribution, etc.).
- investigation of the existing approaches and developing their extensions to triadic numerical data for comparison purposes,
- applying triclustering in recommender systems and social network analysis.

Acknowledgements. We would like to thank our colleagues Boris Mirkin, Jonas Poelmans, Leonid Zhukov, Andrey Kramarenko, Svetlana Vasyukova for helpful comments, and Ruslan Magizov (box OAC-triclustering) and Zarina Sekinaeva (SpecTric) for their programming support. This research was carried out in 2013 within a project of the Basic Research Program of the National Research University Higher School of Economics (Moscow) and affiliated Laboratory of Intelligent Systems and Structural Analysis.

References

1. Jäschke, R., Hotho, A., Schmitz, C., Ganter, B., Stumme, G.: Trias–an algorithm for mining iceberg tri-lattices. In: Proceedings of the Sixth International Conference on Data Mining. ICDM '06, Washington, DC, USA, IEEE Computer Society (2006) 907–911
2. Mirkin, B., Kramarenko, A.V.: Approximate bicluster and tricluster boxes in the analysis of binary data. [15] 248–256
3. Miettinen, P.: Boolean tensor factorization. In Cook, D., Pei, J., Wang, W., Zaïane, O., Wu, X., eds.: ICDM 2011, 11th IEEE International Conference on Data Mining, Vancouver, Canada, IEEE Computer Society, CPS (2011) 447–456
4. Belohlávek, R., Glodeanu, C., Vychodil, V.: Optimal factorization of three-way binary data using triadic concepts. Order **30**(2) (2013) 437–454
5. Kaytoue, M., Kuznetsov, S., Macko, J., Napoli, A.: Biclustering Meets Triadic Concept Analysis. Ann. of Math. and Art. Intell. (2013) (to appear).
6. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. 1st edn. Springer-Verlag New York, Inc., Secaucus, NJ, USA (1999)
7. Lehmann, F., Wille, R.: A triadic approach to formal concept analysis. In: Proceedings of the Third International Conference on Conceptual Structures: Applications, Implementation and Theory, London, UK, Springer-Verlag (1995) 32–43
8. Gnatyshak, D., Ignatov, D.I., Semenov, A., Poelmans, J.: Gaining insight in social networks with biclustering and triclustering. In: BIR. Volume 128 of Lecture Notes in Business Information Processing., Springer (2012) 162–171
9. Nanopoulos, A., Rafailidis, D., Symeonidis, P., Manolopoulos, Y.: Musicbox: Personalized music recommendation based on cubic analysis of social tags. IEEE Transactions on Audio, Speech & Language Processing **18**(2) (2010) 407–412
10. Ignatov, D.I., Kuznetsov, S.O., Magizov, R.A., Zhukov, L.E.: From triconcepts to triclusters. [15] 257–264
11. Ignatov, D.I., Kuznetsov, S.O., Poelmans, J., Zhukov, L.E.: Can triconcepts become triclusters? International Journal of General Systems **42**(6) (2013) 572–593
12. Trabelsi, C., Jelassi, N., Ben Yahia, S.: Scalable mining of frequent tri-concepts from folksonomies. In Tan, P.N., Chawla, S., Ho, C., Bailey, J., eds.: Advances in Knowledge Discovery and Data Mining. Volume 7302 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2012) 231–242
13. Cerf, L., Besson, J., Robardet, C., Boulicaut, J.F.: Closed patterns meet n-ary relations. ACM Trans. Knowl. Discov. Data **3** (March 2009) 3:1–3:36
14. Ignatov, D.I., Kuznetsov, S.O., Poelmans, J.: Concept-based biclustering for internet advertisement. In: ICDM Workshops, IEEE Computer Society (2012) 123–130
15. Rough Sets, Fuzzy Sets, Data Mining and Granular Computing - 13th International Conference, 2011. Proceedings. In: RSFDGrC. Volume 6743 of Lecture Notes in Computer Science., Springer (2011)

A lattice-free concept lattice update algorithm based on $*$CbO[⋆]

Jan Outrata

Dept. Computer Science, Palacky University, Olomouc
17. listopadu 12, CZ-77146 Olomouc, Czech Republic
`jan.outrata@upol.cz`

Abstract. Updating a concept lattice when introducing new objects to input data can be done by any of the so-called incremental algorithms for computing concept lattice of the data. The algorithms use and update the lattice while introducing new objects one by one. The present concept lattice of input data without the new objects is thus required before the update. In this paper we propose an efficient algorithm for updating the lattice from the present and new objects only, not requiring the possibly large concept lattice of present objects. The algorithm results as a modification of the CbO algorithm for computing the set of all formal concepts, or its modifications like FCbO, PCbO or PFCbO, to compute new and modified formal concepts only and the changes of the lattice order relation when input data changes. We describe the algorithm and present an experimental evaluation of its performance and a comparison with AddIntent incremental algorithm for computing concept lattice.

1 Introduction

In applications of Formal Concept Analysis (FCA) [3, 16] the input data are often not fixed during the life of the application and concept lattice is not computed from the data once. The changes of input data result in corresponding changes of concept lattice of the data and to compute the new, changed, concept lattice we would like to compute the changes only and "update" the present lattice instead of recomputing the whole lattice from new, changed, input data.

The particular change of object-attribute relational data processed in FCA, namely the introduction of new objects (described by particular values of attributes), can be handled by the so-called incremental algorithms for computing concept lattice, Norris's [13], Object Intersections [2] or, more recent, AddIntent [12], for instance, or the incremental lattice update algorithms presented in [2]. The algorithms build/update the lattice incrementally by adding objects of input data, one by one, to present concept lattice computed so far from already processed objects, starting from the first object and empty concept lattice.

[⋆] Support by the ESF project No. CZ.1.07/2.3.00/20.0059, the project co-financed by the European Social Fund and the state budget of the Czech Republic, is acknowledged.

The lattice is required for the computation and the present lattice of input data before the introduction of new objects would be required for the update of the lattice when adding the objects. However, that lattice can be large, the application may not store it (it can be even impossible due to space requirements) or it can be stored at different (presentation) place than the computation place, or there can be other drawbacks and difficulties of keeping the whole lattice. Moreover, handling of the other changes of input data like deletion or alteration (i.e. changing of values of attributes) of existing objects would call for more or less (depending on the particular algorithm) extensive modifications of the incremental algorithm.

In the following we propose efficient algorithm for updating the concept lattice, i.e. computing the lattice changes resulting by input data changes, from the present (already processed) and new objects only, not requiring the concept lattice of present objects. The algorithm results as a modification of Kuznetsov's Close-by-One (CbO) algorithm [8–10] for computing the set of all formal concepts or any of its recent derivatives including FCbO [14], PCbO [7] or PFCbO [6]. When introducing new objects to input data the modified algorithm computes and outputs the resulting new and modified formal concepts only and the respective changes in the lattice order relation (pairs of formal concepts to be created or deleted). Note that (1) new formal concept here is a concept in new data with (some of the) introduced objects in its object set (extent) and attribute set (intent) not equal to intent of any concept in the data before the update and modified formal concept is a concept in new data with the same intent as some existing concept in data before update and enlarging its extent by (some of the) introduced objects, other formal concepts in new data are called old; (2) since the concept lattice (before update) is not required by the algorithm the computed changes in the lattice order relation are output only and have to be applied where the (part of the) lattice is stored and (3) introducing new objects to input data cannot result in removal of formal concepts from the concept lattice [4, 15]. Moreover, considering instead the new objects to be some of the objects of input data to be deleted from the data, the present objects to be the set of objects of input data after the deletion and the computed formal concepts to be resulting concepts to remove or modify together with the "inverse" changes of the lattice order relation, we have easily handled also the change of input data consisting of deleting existing objects. The change by altering objects can be handled by first deleting the objects and then by introducing the altered objects, interpreting the output formal concepts accordingly. Finally, changes of input data by introducing new and deleting or altering existing attributes are completely analogical and can be handled (better than by altering objects) by an algorithm re-described with objects and attributes switched or by the present algorithm with objects and attributes switched in input data (transposed data) and in output formal concepts.

The algorithm is described in Section 2, for the case of changing input data by introducing new objects, including an illustrative example. In Section 3 we present an experimental evaluation of performance of the algorithm and a comparison with AddIntent algorithm [12], which is considered one of the up-to-date most efficient incremental algorithms for computing concept lattice.

2 The algorithm

We describe the algorithm as a modification of Fast Close-by-One (FCbO) algorithm [14], as it happens to be one of the up-to-date most efficient (sequential/serial) derivatives of CbO [5]. In essence, the modification equally applies also to other CbO derivatives and CbO itself. A deep knowledge of FCbO is not required in the description, however, basic knowledge is beneficial. The modification consists of two parts: (1) computing new and modified formal concepts only when introducing new objects to input data and (2) determining changes in the lattice order relation. We describe the parts separately in Sections 2.1 and 2.2.

In the descriptions below we assume the reader is familiar with basics of FCA, see [2, 3] for reference. Input object-attribute data (formal context) is denoted by the triplet $\langle X, Y, I\rangle$, with assumed finite nonempty sets of objects $X = \{0, 1, \ldots, m\}$ and attributes $Y = \{0, 1, \ldots, n\}$, and $I \subseteq X \times Y$ being an incidence relation with $\langle x, y\rangle \in I$ saying that object $x \in X$ has attribute $y \in Y$. Concept-forming operators defined on I as usual [3] are denoted by ${}^{\uparrow_I} : 2^X \mapsto 2^Y$ and ${}^{\downarrow_I} : 2^Y \mapsto 2^X$, $\mathcal{B}(X, Y, I)$ denotes the set of all formal concepts in I and $\leq$ the partial order relation on $\mathcal{B}(X, Y, I)$ forming together a concept lattice.

2.1 New and modified concepts

In this section we describe how to compute new and modified formal concepts when introducing new objects to input data. Let us suppose we are introducing to input data $\langle X, Y, I\rangle$ (finite nonempty set of) new objects $X_N = \{m+1, \ldots, m'\}$ not present in X ($X_N \cap X = \emptyset$), sharing (finite nonempty set of) attributes $Y_N = \{i, \ldots, k\}$. We do not assume any overlap of Y_N and Y (Y_N can contain new attributes not present in Y) but in usual scenario of introducing new objects to input data we have $Y_N \subseteq Y$. Denote the incidence relation between X_N and Y_N by $N \subseteq X_N \times Y_N$ and the new input data with new objects added by the triplet $\langle X', Y', I'\rangle$, where $X' = X \cup X_N = \{0, \ldots, m'\}, Y' = Y \cup Y_N = \{0, \ldots, n'\}$, $n' = k$ if $k > n$ and $n' = n$ otherwise, and $I' \subseteq X' \times Y'$ such that $I' \cap (X \times Y) = I$, $I' \cap (X_N \times Y_N) = N$ and $I' \cap (X \times (Y_N \setminus Y)) = I' \cap (X_N \times (Y \setminus Y_N)) = \emptyset$. Hence I' is the union of I and N both extended to X' and Y'.

First, observe that attributes of Y_N which are not shared by any of objects X_N cannot participate in any new nor modified formal concept of $\mathcal{B}(X', Y', I')$, with the exception of formal concept $\langle Y'^{\downarrow_{I'}}, Y'\rangle$. Hence we will assume in the following, without loss of generality, that all attributes of Y_N are shared by at least one object from X_N. For an algorithm for computing new and modified formal concepts only it is then sufficient to process only attributes Y_N.

Now, for any $B' = B^{\downarrow_{I'}\uparrow_{I'}} \subseteq Y_N$, if $B' = B'' = B^{\downarrow_I\uparrow_I}$ then the formal concept $\langle A', B'\rangle \in \mathcal{B}(X', Y', I')$ with the same intent $B' = B''$ as formal concept $\langle A'', B''\rangle \in \mathcal{B}(X, Y, I)$ is a modified formal concept enlarging the extent of

Algorithm 1: Procedure UpdateFastGenerateFrom($\langle A, B\rangle, y, \{N_y \,|\, y \in Y\}$), cf. [14]

```
// list $\langle A, B\rangle$, e.g., print it on screen or store it
if $(A \cap X)^{\uparrow_{I'}} = B$ then
    if $(A \cap X) \subset A$ then
        list $\langle A, B\rangle$ as modified;
    end
else
    list $\langle A, B\rangle$ as new;
end
if $B = Y'$ or $y > n'$ then
    return
end
for $j$ from $y$ upto $n'$ do
    set $M_j$ to $N_j$;
    // go through attributes from $Y_N$ only
    if $j \notin B$ and $j \in Y_N$ and $N_j \cap Y_j \subseteq B \cap Y_j$ then
        set $C$ to $A \cap \{j\}^{\downarrow_{I'}}$;
        set $D$ to $C^{\uparrow_{I'}}$;
        if $B \cap Y_j = D \cap Y_j$ then
            put $\langle\langle C, D\rangle, j\rangle$ to queue;
        else
            set $M_j$ to $D$;
        end
    end
end
while get $\langle\langle C, D\rangle, j\rangle$ from queue do
    UpdateFastGenerateFrom($\langle C, D\rangle, j + 1, \{M_y \,|\, y \in Y\}$);
end
return
```

$\langle A'', B''\rangle$ by objects $A' \setminus A'' \subseteq X_N$ if $A' \supset A''$ (otherwise $\langle A', B'\rangle = \langle A'', B''\rangle$ is old). Otherwise, if $B' \subset B'' = B^{\downarrow_I \uparrow_I}$ (since the $\supset$ inclusion cannot happen since $X' \supset X$) then the formal concept $\langle A', B'\rangle \in \mathcal{B}(X', Y', I')$ is a new formal concept and the formal concept $\langle A'', B''\rangle \in \mathcal{B}(X, Y, I)$ is called its generator [4, 15]. In both cases, $A' \cap X = A''$.

We use the above presented ideas to modify FCbO algorithm described in [14] as recursive procedure FastGenerateFrom. The procedure computes and lists the set $\mathcal{B}(X, Y, I)$ of all formal concepts of input data $\langle X, Y, I\rangle$. We modify the procedure to compute and list the new and modified formal concepts of $\langle X', Y', I'\rangle$ only when adding new objects X_N described by attributes Y_N to $\langle X, Y, I\rangle$. The modified procedure UpdateFastGenerateFrom is depicted in Algorithm 1. The original procedure FastGenerateFrom is thoroughly described in [14] so we describe only the modifications introduced in UpdateFastGenerateFrom.

The procedure accepts as its arguments a formal concept $\langle A, B\rangle$ (an initial formal concept), an attribute $y \in Y_N$ (first attribute to be processed) and a set $\{N_y \subseteq Y \,|\, y \in Y\}$ of subsets of attributes Y (see [14] for the meaning of the set), and uses local variables *queue* as a temporary storage for computed formal concepts and M_y $(y \in Y)$ as sets of attributes which are used in place of the third argument for further invocations of the procedure. After invocation, the procedure recursively descends through the space of new and modified formal concepts of $\langle X', Y', I'\rangle$ resulted by adding new objects X_N described by attributes Y_N to $\langle X, Y, I\rangle$.

When invoked with $\langle A, B\rangle$ and $y \in Y_N$ (first attribute to be processed) and $\{N_y \,|\, y \in Y\}$, UpdateFastGenerateFrom first checks if $(A \cap X)^{\uparrow_{I'}}$ equals B (line 1), in which case, if further $(A \cap X) \subset A$ (line 2), $\langle A, B\rangle$ is a modified formal concept (see above) and it is listed as such. In the other case, $\langle A, B\rangle$ is a new formal concept. Then the procedure behaves exactly as the original procedure FastGenerateFrom, see [14] for full description, with the exception that it goes through attributes $j \in Y_N$ only (line 13). Let us recall that the set $Y_j \subseteq Y'$ is defined by

$$Y_j = \{y \in Y' \,|\, y < j\}.$$

In order to list all new and modified formal concepts of $\langle X', Y', I'\rangle$ which are not formal concepts of $\langle X, Y, I\rangle$, each of them exactly once, we invoke UpdateFastGenerateFrom with $\langle \emptyset^{\downarrow_{I'}}, \emptyset^{\downarrow_{I'}\uparrow_{I'}}\rangle$, y being the first attribute in Y_N and $\{N_y = \emptyset \,|\, y \in Y\}$ as its initial arguments. The correctness of Algorithm 1 follows from the correctness of FCbO algorithm [14] and the above description of its modification.

Example 1. We illustrate Algorithm 1 on the following example. Consider an input data given in an usual way (rows correspond to objects, columns to attributes and table entries indicate whether an object has an attribute) by the upper part, rows 0 to 2, of the table depicted below (left). The data induces 8 formal concepts C_1 to C_8 depicted on the right.

I	0	1	2	3	4	5
0	×		×	×		
1		×		×	×	×
2	×	×				×
3		×		×		×
4	×		×	×		×

$C_1 = \langle\{0,1,2\}, \emptyset\rangle$, $C_5 = \langle\{0\}, \{0,2,3\}\rangle$,
$C_2 = \langle\{0,2\}, \{0\}\rangle$, $C_6 = \langle\{1,2\}, \{1,5\}\rangle$,
$C_3 = \langle\{2\}, \{0,1,5\}\rangle$, $C_7 = \langle\{1\}, \{1,3,4,5\}\rangle$,
$C_4 = \langle\emptyset, \{0,1,2,3,4,5\}\rangle$, $C_8 = \langle\{0,1\}, \{3\}\rangle$.

$C_1^* = \langle\{0,1,2,3,4\}, \emptyset\rangle$, $C_6^* = \langle\{1,2,3\}, \{1,5\}\rangle$,
$C_2^* = \langle\{0,2,4\}, \{0\}\rangle$, $C_{11} = \langle\{1,3\}, \{1,3,5\}\rangle$,
$C_5^* = \langle\{0,4\}, \{0,2,3\}\rangle$, $C_8^* = \langle\{0,1,3,4\}, \{3\}\rangle$,
$C_9 = \langle\{4\}, \{0,2,3,5\}\rangle$, $C_{12} = \langle\{1,3,4\}, \{3,5\}\rangle$,
$C_{10} = \langle\{2,4\}, \{0,5\}\rangle$, $C_{13} = \langle\{1,2,3,4\}, \{5\}\rangle$.

Now we introduce to the data two new objects represented by rows 3 and 4 of the lower part of the table. The introduction results in induction of 5 new formal concepts C_9 to C_{13} and 5 modified formal concepts C^*_- depicted below the old formal concepts. The concepts are listed down-right in order in which they are listed by procedure UpdateFastGenerateFrom.

Remark 1. Algorithm 1 can be easily used to list all formal concepts of input data $\langle X'', Y'', I''\rangle$ incrementally, i.e. processing the data object by object, as incremental algorithms for computing concept lattice do. Namely, we invoke procedure UpdateFastGenerateFrom with initial arguments repeatedly for each object $i \in X'' = \{0, \ldots, n''\}$, setting $\langle X, Y, I\rangle := \langle\{0, \ldots, i-1\}, \bigcup_{j=0}^{i-1}\{j\}^{\uparrow_{I''}}, I \cap (\{0, \ldots, i-1\} \times \bigcup_{j=0}^{i-1}\{j\}^{\uparrow_{I''}})\rangle$, $X_N := \{i\}$, $Y_N := \{i\}^{\uparrow_{I''}}$ and $N := X_N \times Y_N$ for each invocation, and filter out listed modified formal concepts listing new formal concepts only. Such a computation of all formal concepts of data is, however, quite inefficient due to many repeated (though efficient) computations of formal concepts listed as modified by procedure UpdateFastGenerateFrom, see the performance evaluation in Section 3. The concepts are, moreover, filtered out from the listing but, however, necessary to compute in order to decide whether a concept is new or modified. Incremental algorithms iterate over (so far) incrementally computed and stored concept lattice to decide and compute new formal concepts which is more efficient. On the other hand, as discussed in introduction Section 1, the concept lattice is required for the computation and must be stored by the incremental algorithms, while Algorithm 1 requires input data only/instead and does not need to store anything.

2.2 Lattice order relation

In this section we describe how to determine the changes in the concept lattice order relation which need to be done with the addition of new formal concepts to the lattice. Note that the modified formal concepts cannot raise a change in the relation since attribute set (intent) of a modified concept remains unchanged with introduction of new objects to input data, as discussed above.

In order to determine the changes in concept lattice order relation by the modified FCbO algorithm introduced in previous Section 2.1, we first have to determine the concept lattice order relation alone since the FCbO algorithm, as well as the modification, computes the set of formal concepts of input data only. So, in the following, we describe an extension of Fast Close-by-One (FCbO) algorithm [14] to determine the concept lattice order relation $\leq$ on the set of all formal concepts $\mathcal{B}(X, Y, I)$ of input data $\langle X, Y, I\rangle$ computed by the original algorithm, i.e. making an algorithm for computing concept lattice of $\langle X, Y, I\rangle$. In fact, we will not determine the whole order relation but rather its cover relation. Recall that the cover relation on $\mathcal{B}(X, Y, I)$ for $\leq$ is defined such that a formal concept $\langle A_2, B_2\rangle$ covers a formal concept $\langle A_1, B_1\rangle$ if $\langle A_1, B_1\rangle \leq \langle A_2, B_2\rangle$ and there is no formal concept $\langle A_3, B_3\rangle$ distinct from both $\langle A_1, B_1\rangle$ and $\langle A_2, B_2\rangle$ such that $\langle A_1, B_1\rangle \leq \langle A_3, B_3\rangle \leq \langle A_2, B_2\rangle$, i.e. the cover relation is the transitive reduct of $\leq$. And since we do not store and use the computed concept lattice in our algorithm for updating the lattice, we will not store and use the computed formal concepts nor the concept lattice order cover relation in the extended FCbO algorithm as well. Besides listing the formal concepts, we will only list the pairs of formal concepts to be created or deleted in the relation at the place where it is stored.

We now describe how to extend FCbO algorithm, as described in [14], to determine the concept lattice order cover relation on the set of computed formal concepts, i.e. to compute concept lattice. We can use the pseudocode in Algorithm 1 if we look apart from the modifications to FCbO introduced there (the check whether $\langle A, B\rangle$ is new or modified formal concept between lines 1 and 6 and going through attributes from Y_N only at line 12), in which case we obtain the original procedure FastGenerateFrom from [14]. We will need a little knowledge of FCbO (or CbO) now. The algorithm, see the pseudocode of procedure UpdateFastGenerateFrom, computes a formal concept $\langle C, D\rangle = \langle A \cap \{j\}^{\downarrow}, (A \cap \{j\}^{\downarrow})^{\uparrow}\rangle$ (lines 14 and 15) from a formal concept $\langle A, B\rangle$ for all attributes $j \in Y$ such that $y \leq j \leq n$ which are not present in B (lines 12 and 13). Certainly $\langle C, D\rangle \leq \langle A, B\rangle$. If $\langle C, D\rangle$ passes the canonicity test (line 16) and is listed we list the pair $\langle\langle C, D\rangle, \langle A, B\rangle\rangle$ as to be created in the cover relation in spite of that the formal concepts in it need not fulfill the definition of cover relation. We do it since a test for the fulfilment would be too expensive compared to that in the case of nonfulfilment we will list the pair later again as to be deleted from the relation.

Next, since further formal concepts $\langle C \cap \{j'\}^{\downarrow}, (C \cap \{j'\}^{\downarrow})^{\uparrow}\rangle$ are computed from $\langle C, D\rangle$ in the next recursive invocation of (Update)FastGenerateFrom for attributes $j' \geq j + 1$ ($j + 1$ is passed in y argument in the invocation, line 20), in order to list pairs $\langle\langle E, F\rangle, \langle C, D\rangle\rangle$ to be created in the cover relation, we determine formal concepts $\langle E, F\rangle = \langle C \cap \{i\}^{\downarrow}, (C \cap \{i\}^{\downarrow})^{\uparrow}\rangle$ for all attributes $j - 1 \geq i \geq 0$ (which are not present in D). Due to the order in which formal concepts are computed by FCbO, formal concepts $\langle E, F\rangle$ were already computed (and listed) in some of the previous stages of the algorithm before the computation of $\langle C, D\rangle$. However, since formal concepts are not stored in FCbO nor in our extension, we have to compute the concepts $\langle E, F\rangle$ repeatedly. Again, formal concepts $\langle E, F\rangle, \langle C, D\rangle$ in the pairs need not fulfill the definition of cover relation. Here, however, we can use a cheap test known from Lindig's NextNeighbor algorithm [11], but limited to attributes $0, \ldots, j - 1 \in Y$. The test is based on the fact that a formal concept $\langle C, D\rangle \neq \langle Y^{\downarrow}, Y\rangle$ covers a formal concept $\langle E, F\rangle = \langle C \cap \{i\}^{\downarrow}, (C \cap \{i\}^{\downarrow})^{\uparrow}\rangle, i \notin D$ iff $(C \cap \{k\}^{\downarrow})^{\uparrow} = F$ for all attributes $k \in F \setminus D$ (cf. Theorem 1 in [11]). If the test passes, we list the pair $\langle\langle E, F\rangle, \langle C, D\rangle\rangle$ as to be created in the cover relation. We do it even for formal concepts in the pair which pass the test and do not fulfill the definition of cover relation due to the limitation of the test to attributes $0, \ldots, j - 1 \in Y$ for the same reason as above for pairs $\langle\langle C, D\rangle, \langle A, B\rangle\rangle$. To resolve these cases we simply, together with listing the pair $\langle\langle E, F\rangle, \langle C, D\rangle\rangle$, list as to be deleted from the relation the pair $\langle\langle E, F\rangle, \langle A, B\rangle\rangle$ which does not fulfill the definition of the relation and could have been listed as to be created in the cover relation in this or the previous stage (previous recursive invocation of procedure (UPDATE)FASTGENERATEFROM) of the extended algorithm. The justification of this and that all such pairs will be listed as to be deleted from the cover relation is postponed to the full version of the paper.

The modified procedure LATTICEFASTGENERATEFROM, which implements the above presented ideas to procedure FASTGENERATEFROM to extend FCbO algorithm to determine the concept lattice order cover relation on the set of formal concepts computed by FCbO, i.e. to compute concept lattice, is depicted in Algorithm 2. As with the procedure UPDATEFASTGENERATEFROM in Section 2.1, we describe only the modifications introduced to FASTGENERATEFROM.

The original FCbO algorithm remains in essence intact, including its arguments and local variables, all the modifications to original procedure FASTGENERATEFROM are in the computed formal concept $\langle C, D\rangle$ processing part before the recursive call of the procedure (line 28), between lines 16 and 29. First note that the listing of $\langle C, D\rangle$ moved from the beginning of the procedure (see Algorithm 1) here, resulting effectively in the need to list the formal concept passed in the initial invocation of the procedure before the invocation. Note also that this move does not change the order of listed formal concepts. Then, after listing the pair $\langle\langle C, D\rangle, \langle A, B\rangle\rangle$ as to be created in the cover relation, between lines 18 and 27, formal concepts $\langle E, F\rangle = \langle C \cap \{i\}^{\downarrow}, (C \cap \{i\}^{\downarrow})^{\uparrow}\rangle, i \notin D$ covered by $\langle C, D\rangle$ are re-computed (lines 20 and 21) and listed in pairs $\langle\langle E, F\rangle, \langle C, D\rangle\rangle$ as to be created in the cover relation, for attributes $j - 1 \geq i \geq 0$. The test of covering (line 23) is performed using the *min* local variable in a slightly modified form borrowed from the original description of Lindig's NextNeighbor algorithm in [11]. With listing of $\langle\langle E, F\rangle, \langle C, D\rangle\rangle$, the pair $\langle\langle E, F\rangle, \langle A, B\rangle\rangle$ is listed as to be deleted from the cover relation as discussed above.

In order to output concept lattice of $\langle X, Y, I\rangle$, that means to list all formal concepts of $\langle X, Y, I\rangle$, each of them exactly once, together with all pairs of formal concepts to create or delete (if the pair exists) in order to obtain the cover relation of concept lattice of $\langle X, Y, I\rangle$, each pair of formal concepts in the cover relation listed exactly once, we first list $\langle\emptyset^{\downarrow}, \emptyset^{\downarrow\uparrow}\rangle$ and then invoke LATTICEFASTGENERATEFROM with $\langle\emptyset^{\downarrow}, \emptyset^{\downarrow\uparrow}\rangle$, $y = 0$ and $\{N_y = \emptyset \,|\, y \in Y\}$ as its initial arguments. The correctness of Algorithm 2 follows from the correctness of FCbO algorithm [14] and the above description of its extension.

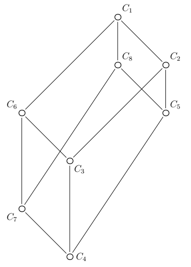
Determination of changes in (the cover relation of) concept lattice of $\langle X, Y, I\rangle$ needed to be done in reaction to the introduction of new objects to input data is then a matter of merging the Algorithms 1 and 2. In addition to that we list, for each new formal concept $\langle C, D\rangle$ computed from formal concept $\langle A, B\rangle$ (or $\langle C, D\rangle = \langle\emptyset^{\downarrow_{I'}}, \emptyset^{\downarrow_{I'}\uparrow_{I'}}\rangle$) and its generator $\langle E, F\rangle = \langle C \cap X, (C \cap X)^{\uparrow_{I'}}\rangle$, with listing of $\langle C, D\rangle$ also pairs $\langle\langle F^{\downarrow_{I'}}, F\rangle, \langle C, D\rangle\rangle$ as to be created in the cover relation and $\langle\langle F^{\downarrow_{I'}}, F\rangle, \langle A, B\rangle\rangle$ as to be deleted from the cover relation. The pairs will then not be listed in the extension introduced in procedure LATTICEFASTGENERATEFROM. Due to space limitations of the paper, the merged algorithm is postponed to the full version of the paper.

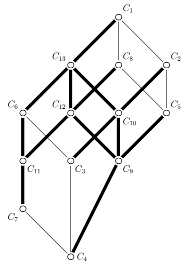
Algorithm 2: Procedure LatticeFastGenerateFrom($\langle A, B\rangle, y, \{N_y \,|\, y \in Y\}$), cf. [14]

```
if $B = Y$ or $y > n$ then
    return
end
for $j$ from $y$ upto $n$ do
    set $M_j$ to $N_j$;
    if $j \notin B$ and $N_j \cap Y_j \subseteq B \cap Y_j$ then
        set $C$ to $A \cap \{j\}^{\downarrow}$;
        set $D$ to $C^{\uparrow}$;
        if $B \cap Y_j = D \cap Y_j$ then
            put $\langle\langle C, D\rangle, j\rangle$ to $queue$;
        else
            set $M_j$ to $D$;
        end
    end
end
while get $\langle\langle C, D\rangle, j\rangle$ from $queue$ do
    // list $\langle C, D\rangle$, e.g., print it on screen or store it
    // list $\langle\langle C, D\rangle, \langle A, B\rangle\rangle$ as to be created in the cover relation,
       e.g., print $C$ and $A$ (or $D$ and $B$) on screen or store them
    set $min$ to $Y_j$;
    for $i$ from $j - 1$ downto $0$ do
        if $i \notin D$ then
            set $E$ to $C \cap \{i\}^{\downarrow}$;
            set $F$ to $E^{\uparrow}$;
            set $min$ to $min \setminus \{i\}$;
            if $D \cap Y_j \cap min = F \cap Y_j \cap min$ then
                // list $\langle\langle E, F\rangle, \langle C, D\rangle\rangle$ as to be created in the cover
                   relation
                // list $\langle\langle E, F\rangle, \langle A, B\rangle\rangle$ as to be deleted from the cover
                   relation
                set $min$ to $min \cup \{i\}$;
            end
        end
    end
    LatticeFastGenerateFrom($\langle C, D\rangle, j + 1, \{M_y \,|\, y \in Y\}$);
end
return
```

Example 2. We illustrate Algorithm 2 and the merged algorithm for the input data from Example 1. Below (top left) there is depicted concept lattice consisting of the 8 formal concepts C_1 to C_8, with the concepts and pairs of concepts in cover relation of the lattice, listed in the order in which they are listed by procedure LatticeFastGenerateFrom, below the lattice.

Next (to the right) to the lattice there is then the concept lattice and below the listing of the 8 formal concepts the listing of the 5 new formal concepts C_9 to C_{13} and changes in the cover relation after the introduction of the two new objects to the data. The changes are depicted in bold face in the lattice and the listing of concepts and pairs of concepts in the changes are again listed down-right in order in which they would be listed by a procedure describing the merged algorithm.





C_1,
$C_2 : \langle C_2, C_1\rangle$,
$C_3 : \langle C_3, C_2\rangle$,
$C_4 : \langle C_4, C_3\rangle$,
$C_5 : \langle C_5, C_2\rangle, \langle C_4, C_5\rangle$,
$C_6 : \langle C_6, C_1\rangle, \langle C_3, C_6\rangle$,
$C_7 : \langle C_7, C_6\rangle, \langle C_4, C_7\rangle$,
$C_8 : \langle C_8, C_1\rangle, \langle C_7, C_8\rangle, \langle C_5, C_8\rangle$.

$C_9 : \langle C_9, C_5^*\rangle, \langle C_4, C_9\rangle$,
$C_{10} : \langle C_{10}, C_2^*\rangle, \langle C_3, C_{10}\rangle, \langle C_9, C_{10}\rangle$,
$C_{11} : \langle C_{11}, C_6^*\rangle, \langle C_7, C_{11}\rangle$,
$C_{12} : \langle C_{12}, C_8^*\rangle, \langle C_{11}, C_{12}\rangle, \langle C_9, C_{12}\rangle$,
$C_{13} : \langle C_{13}, C_1^*\rangle, \langle C_6^*, C_{13}\rangle, \langle C_{12}, C_{13}\rangle, \langle C_{10}, C_{13}\rangle$.

3 Experimental evaluation

The asymptotic worst-case time complexity of Algorithm 1 remains the same as of FCbO (and CbO) algorithm, $O(|\mathcal{B}(X,Y,I)|\cdot|Y|^2\cdot|X|)$, because when "introducing" all objects to empty data it actually performs FCbO. The complexity of Algorithm 2 is in the worst case $|Y|$ factor higher but on average it performs a constant factor slower than FCbO (and ramification of the worst-case time complexity of FCbO itself remains a challenging open problem [14]).

We have run several experiments to evaluate the performance of Algorithms 1 and 2 and also the merged algorithm. For listing all formal concepts or concept lattice of input data we also compared the algorithms with AddIntent incremental algorithm [12] for computing concept lattice. In the comparison, we also run Algorithm 1 and the merged algorithm in the way processing input data incrementally as mentioned in Remark 1, for the sake of presenting more fair comparison with true incremental algorithm, though such usage of our algorithms is not efficient (as mentioned in the remark).

In the experiments, we used our implementations of the presented algorithms in ANSI C, which are modifications of our (performance efficient) FCbO algorithm implementation used for performance evaluation in [14], while the implementation of AddIntent algorithm was borrowed from one of the authors of [12].

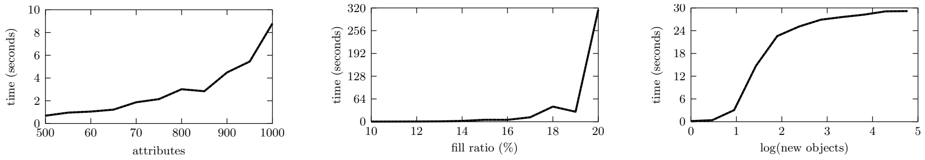


Fig. 1. Running time of Algorithm 1 on random data dependent on number of attributes (on the left, introducing a single new object), on fill ratio (percentage of ×s, in the middle, introducing a single new object) and on number of new objects (on the right).

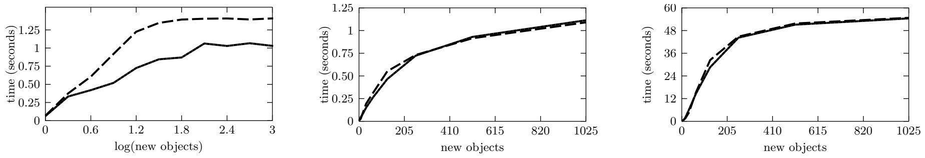


Fig. 2. Running time of Algorithm 1 dependent on number of new (last) objects for mushroom (on the left), anonymous-msweb (in the middle) and T10I4D100K datasets (on the right), solid line—orig. object ordering, dashed line—random object ordering.

The experiments were run on otherwise idle 32-bit i386 hardware (2× Intel Xeon 3.2 GHz, 3 GB RAM) and we were interested in the performance of all algorithms measured by running time. We have run the algorithms on synthetic randomly generated data with various size and percentage of ×s in the table (fill ratio, with normal distribution) as well as with three selected real benchmark datasets from the UCI Machine Learning Repository [1].

In the first set of experiments we evaluated Algorithm 1 for updating the set of all formal concepts (computing new and modified concepts) when introducing new objects to input data. The performance for introducing a single new object (with randomly generated attributes) to random data with 100000 objects is illustrated in Figure 1. The graphs show the dependency of time required to compute the update on the number of attributes, of data with fixed table fill ratio 5 % (the graph on the left) and on the fill ratio of tables with fixed 150 attributes (the graph in the middle). Figure 1 (right) then illustrates the performance for introducing a growing number of new objects to random data with the number of objects being 100000 minus the number of the new objects, 200 attributes and 5 % table fill ratio. Note that the number of new objects in the graph is in logarithmic scale. The illustration for the benchmark datasets, of the performance of introducing a growing number of last objects of the data to the data without the objects, is presented in Figure 2 (right). In the graph the solid line is for data with objects ordered as in the original dataset and the dashed line is for data with randomly ordered objects (the time is an average from three orderings).

We can see from the graphs showing the performance of updating the set of all formal concepts when introducing a single new object to the data (Figure 1,

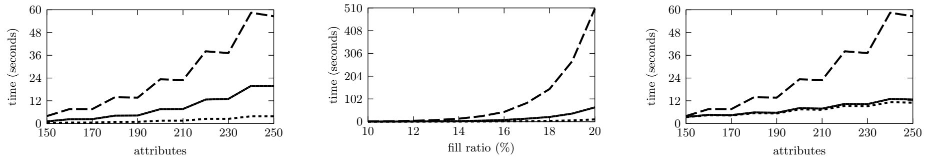


Fig. 3. Running time dependent on number of attributes (on the left and right) and on fill ratio (percentage of ×s, in the middle), solid line—Algorithm 2 (left, middle)/merged algorithm (right), dashed line—AddIntent, dotted line—FCbO (left, middle)/Algorithm 1 (right).

Table 1. Running time (in seconds) of determining the cover relation of concept lattice and numbers of formal concepts for selected datasets.

dataset	size	fill ratio	#concepts	orig. object ordering			random object ordering		
				Alg. 2	AddInt.	FCbO	Alg. 2	AddInt.	FCbO
mushroom	8124 × 119	19.167 %	238710	10.010	9.760	0.830	11.025	8.061	0.919
anon. web	32711 × 296	1.019 %	129009	2.860	6.540	1.326	2.841	6.521	1.306
T10I4D100K	100000 × 1000	1.010 %	2347376	453.060	342.966	55.523	452.893	343.056	55.490

left and middle) that this operation is extremely fast compared to computing all formal concepts which took 7061 seconds for 500 attributes and 355 seconds for fill ratio 10 %! The reason is of course computing only a bare fraction of new and modified formal concepts among all concepts. Introducing more new objects (Figure 1, right, and Figure 2) is much faster too, up to a limit of the number of new objects depending on the total number of objects in data and then being close to the time of computing all formal concepts (compare for benchmark datasets with FCbO times in Table 1). Note also that running times for mushroom dataset differ for original and random orderings of objects.

In the second set of experiments we evaluated Algorithm 2 for determining the cover relation of concept lattice of input data. The performance for random data is illustrated in Figure 3. The graph on the left shows the dependency of required time on the number of attributes, of data with 10000 objects and fixed table fill ratio 5 %, the graph in the middle shows the dependency on the fill ratio of tables with 1000 objects and fixed 100 attributes. The graphs show also the comparison with AddIntent algorithm and the original FCbO algorithm; the solid line is for Algorithm 2, the dashed line is for AddIntent and the dotted line is for FCbO. The performance for the benchmark datasets is presented in Table 1. Again, the times are given for data with objects ordered as in the original dataset and for data with randomly ordered objects (the time is an average from three orderings), and we also put information on size, fill ratio and the number of all formal concepts for each dataset.

From the graphs we can see that Algorithm 2 considerably outperforms AddIntent algorithm on synthetic random data (in particular for higher fill ratio of data table), while for real benchmark datasets this must not always be the case (T10I4D100K, closely mushroom). This, however, heavily depends on the concept lattice structure (size of the cover relation) of the datasets. We can also

Table 2. Running time (in seconds) of computing all formal concepts or concept lattice when processing input data incrementally and numbers of formal concepts computed in total for selected datasets.

dataset	#concepts in total	orig. object ordering			random object ordering		
		merged alg.	AddInt.	Alg. 1	merged alg.	AddInt.	Alg. 1
mushroom	9577435	130.746	9.760	128.993	284.983	8.061	284.130
anon. web	992259	41.386	6.540	41.320	41.342	6.521	41.077
T10I4D100K	14240918	2389.180	342.966	2384.160	2388.793	343.056	2378.703

see from the comparison with FCbO algorithm, as to the rest expected, that determining the cover relation takes considerably more time than computing just the set of the formal concepts. Also, ordering of objects in the benchmark datasets makes almost no difference in the running times.

Finally, the comparison of performance of Algorithm 1 and the merged algorithm with AddIntent, of computing the set of all formal concepts or concept lattice when processing input data incrementally as mentioned in Remark 1, is presented in Figure 3 (right) and Table 2. The graph shows the performance for the data with 10000 objects and fixed table fill ratio 5 % from the preceding evaluation of Algorithm 2 (the time is an average from three random orderings of objects); the solid line is for the merged algorithm, the dashed line is for AddIntent and the dotted line is for Algorithm 1. In the table we also put, for our algorithms, for the sake of comparison, the numbers of formal concepts computed in total (for original ordering of objects in datasets).

The results are really interesting here. For the real benchmark datasets AddIntent algorithm significantly outperforms our algorithms, see Table 2. This was expected since, as hinted by Remark 1, the algorithms are not designed and ment to be used as incremental algorithms (processing objects one by one). What is, however, very surprising is that on synthetic random data both Algorithm 1 and the merged algorithm considerably outperform AddIntent and, moreover, computing concept lattice (determining cover relation) takes just a little more time than computing the set of formal concepts only. It seems that the usage of the algorithms as incremental algorithms, after all, deserves more attention!

4 Conclusion

We have introduced algorithms for updating the set of all formal concepts of object-attribute relational data when the data change (new objects or attributes are introduced or existing are deleted or altered) by computing only new and modified formal concepts and for determining the concept lattice order relation, i.e. computing concept lattice of the data. Together the algorithms form an algorithm for updating concept lattice of object-attribute data from the data only, not requiring the possibly large concept lattice computed before the update as the so-called incremental (update) algorithms do. The algorithms result as modification or extension of FCbO algorithm for computing the set of all formal concepts of data and the modifications can be equally applied to any other recent algorithms (PCbO, PFCbO) derived from Kuznetsov's CbO algorithm. The experimental evaluation of performance of the algorithms have shown that the update is performed by the first algorithm significantly faster than re-computing all formal concepts or whole concept lattice and that the second algorithm is performance comparable to incremental algorithms for computing concept lattice.

The future research will be focused on further experimental and real-world application use evaluation of the algorithms and performance comparison with other incremental (update) algorithms for computing concept lattice.

References

1. Bache K., Lichman M.: *UCI Machine Learning Repository.* University of California, Irvine, CA, School of Information and Computer Sciences, 2013. `http://archive.ics.uci.edu/ml`
2. Carpineto C., Romano G.: *Concept data analysis. Theory and applications.* J. Wiley, 2004.
3. Ganter B., Wille R.: *Formal concept analysis. Mathematical foundations.* Berlin: Springer, 1999.
4. Godin R., Missaoui R., Alaoui H.: Incremental Concept Formation Algorithms Based on Galois Lattices. *Computation Intelligence* **11**(1995), 246-267.
5. Kirchberg M., Leonardi E., Tan Y. S., Link S., K L Ko R., Lee B. S.: Formal Concept Discovery in Semantic Web Data. In: *Proc. ICFCA 2012, LNAI*, **7278**(2012), 164–179.
6. Krajca P., Outrata J., Vychodil V.: Advances in algorithms based on CbO. In: *Proc. CLA 2010*, 325–337.
7. Krajca P., Outrata J., Vychodil V.: Parallel algorithm for computing fixpoints of galois connections. *Annals of Mathematics and Artificial Intelligence* **59**(2)(2010), 257–272.
8. Kuznetsov S. O.: Interpretation on graphs and complexity characteristics of a search for specific patterns. *Automatic Documentation and Mathematical Linguistics*, **24**(1)(1989), 37–45.
9. 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.
10. Kuznetsov S. O.: Learning of Simple Conceptual Graphs from Positive and Negative Examples. *PKDD 1999*, 384–391.
11. Lindig C.: Fast concept analysis. *Working with Conceptual Structures––Contributions to ICCS 2000*, 2000, 152–161.
12. van der Merwe D., Obiedkov S. A., Kourie D. G.: AddIntent: A New Incremental Algorithm for Constructing Concept Lattices. In: *Proc. ICFCA 2004, LNAI* **2961**(2004), 205–206.
13. Norris E. M.: An Algorithm for Computing the Maximal Rectangles in a Binary Relation. *Revue Roumaine de Mathématiques Pures et Appliquées*, **23**(2)(1978), 243–250.
14. Outrata J., Vychodil V.: Fast Algorithm for Computing Fixpoints of Galois Connections Induced by Object-Attribute Relational Data. *Information Sciences* **185**(1)(2012), 114-127.
15. Valtchev P., Missaoui R.: Building Concept (Galois) Lattices from Parts: Generalizing the Incremental Methods. *LNAI* **2120**(2001), 290-303.
16. Wille R.: Restructuring lattice theory: an approach based on hierarchies of concepts. *Ordered Sets*, 1982, 445–470, 1982.

Applying User-Guided, Dynamic FCA to Navigational Searches for Earth Science Data and Documentation

Bruce R. Barkstrom

15 Josie Ln, Asheville NC 28804, USA

Abstract. This paper describes data structures and algorithms that allow disciplinary taxonomic experts to embed Formal Contexts within a graph of Archive Information Packages (AIP's). The AIP's are standardized objects that provide access to the Inventoried Objects (IO's) in an archive. For an archive containing Earth science data, IO's may be physical specimens or numerical data files. They are not just textual files that provide a corpora of keywork phrases. The graph serves as a Table of Contents for the archive's collection. A data user familiar with the discipline's taxonomy having a recognizable search target can navigate through the graph to identify and access the archive's IO's.

1 Introduction

An archive containing Earth science data may contain physical objects or numerical data files. Its Inventoried Objects (IO's) do not necessarily contain textual content. Thus, they may not provide keyword phrases chosen from their textual content. This paper's algorithms assume that a disciplinary taxonomist develops an IO classification. To do this, the taxonomist creates vertices in a network. Each nonterminal vertex points to a unique Partitioned Formal Context (PFC). Each terminal vertex points to a unique IO.

The Archive Information Packages (AIP's) described in the ISO standard known as the Open Archive Information System (OAIS) Reference Model (RM) [1] serve as prototypical data structures for the network's vertices. The nonterminal vertices are Archive Information Collections (AIC's). The terminal ones are Archive Information Units (AIU's) that point to IO's. Disciplinary taxonomists create the network as the Archive adds to its collection of IO's. Archive users familiar with that taxonomy can navigate through the network's paths to select a target IO that satisfies their search criteria. Navigation differs from a browsing approach in which a user does not have a specific kind of target IO in mind.

This taxonomic approach markedly reduces the size of the Formal Contexts a FCA algorithm uses to create Formal Concepts from the PFC's. The computed Formal Concepts guide the user's navigation from AIC to AIC until the user finds an AIU with the target IO. The next section includes brief examples of the kinds of IO's in archives of Earth science data. Then, it describes the way they enter the taxonomic network. Later sections include outlines of data structures for the network vertices and pseudo code for the key algorithms in this approach.

2 Creating a Taxonomic Classification Network

This section discusses the basis for creating a taxonomic classification network. The first subsection notes that a disciplinary taxonomist provides the a priori categories for the taxonomy. This subsection also provides examples of categories for two kinds of Earth science data. These categories depend on the disciplines providing the data. Thus, this paper treats them as axiomatic. The second subsection provides a formal discussion of the AIC and AIU data structures. The third subsection deals with a rough scaling analysis of a taxonomic network.

2.1 The Axiomatic Disciplinary Basis for a Taxonomy

The a priori basis for creating a taxonomy for an archive's collection lies in the scientific disciplines that produce the IO's. As with the OAIS RM, we assume that the IO's are atomic so they have no internal components that the archive records. In Earth science, IO's may be physical objects, such as biological or geological specimens. They may include hand-written or printed observational records or technical reports. Finally, Earth science IO's may be digital files, the bulk of which contain numerical values based on measurements from automated instruments.

A second axiomatic basis is the Formal Context (FC) formed by all the IO's in the archive. The union of the IO's forms the extent of this FC. Its intent is the union of all the IO attributes.

The NOAA Emergency Response Imagery Collection is an archive of digital images taken by an automated camera mounted on an aircraft [2]. The aircraft flew one or two days after disastrous storms. There have been about twenty storms in the last decade that were disastrous enough to lead to aircraft missions. These included Hurricane Katrina in 2005 and Hurricane Sandy in 2012. Each mission produces three kinds of IO's: high resolution gif images, low resolution gif thumbnails, and zipped files containing collections of images. The archive's curators organize the images in a zipped file in a sequence along an aircraft flight path.

The ERI project provides a web site through which emergency responders, such as federal disaster coordinators or insurance adjustors can download IO's. These users evaluate the storm damage and plan appropriate action based on the images. Altogether, the total collection probably contains about 60,000 IO's. Each IO has an ID, a storm name, a data collection date, and a text storm centroid location. The high resolution and thumbnail gif's have four latitude/longitude positions at the corners of each image. With eleven attributes for most of the files, there will be about 660,000 attributes ($11 \times 60,000$) in the Formal Context intent for this collection's IO's.

In the taxonomic network, the AIC's that lie in the first level below the root refer to images from specific storms. Thus, these collections include images from storms such as Hurricanes Katrina, Ike, or Sandy. In the level below that, each storm AIC has two collections. One is an AIC for a collection of images in coarse geographic bins. Those images include the high-resolution gif images and the thumbnails. The second AIC is a collection of zipped files along the flight paths.

The AIC's that are children of each geographic bin AIC contain AIC's with one high resolution image and one low-resolution image. The AIC's below the one with the flight path collection contain AIU's based on an individual flight path. The network replicates this layered AIC structure across the entire collection of IO's. Even so, each PFC is unique.

For biological specie classification, the categories for the AIC's are Kingdoms, Subkingdoms, Classes, Orders, Families, Genera, Species, and Races. The individual specimens in an archive's biological specimen collection are the IO's in the AIU's.

2.2 A Formal Definition of AIC's and AIU's

A disciplinary taxonomist creates AIP's and AIU's. This is a familiar process in biology. [3] [p. viii] notes that such a "grouping . . . though based on natural characters and relationships is not governed by any rule drawn from nature for determining just what [. . .attributes] shall be sufficient to constitute a Specie, a Genus or a Family. These groups are, therefore, necessarily more or less arbitrary and depend upon the judgement of scientific experts."

The AIC's and AIU's are vertices in a graph. The pseudo code in the Taxonomy Network Data Structure provides a pointer-based formalization of the relationship between the AIC's and the AIU's.

Taxonomy Network Data Structure

```
type p_PFC is access PFC_Type; -- Pointer to a PFC
type PFC_Type(N_O : in positive; N_A : in positive)
    is array(1 .. N_O, 1 .. N_A) of Boolean;
type Type_Of_AIP is (Collection, IO);
type p_AIP is access AIP; -- Pointer to an AIP
type AIP(AIP_Type : Type_Of_AIP := Collection) is record
    AIP_Identifier : Bounded_String;
    Parent_AIP : p_AIP;
    Next_Sibling_AIP : p_AIP;
    case AIP_Type is
        when Collection =>
            PFC : p_PFC;
            First_Child_AIP : p_AIP;
        when IO =>
            Object_Identifier : Bounded_String;
    end case;
end record;
```

If all of the PFC's in the graph were diagonal, each object in the PFC would have only one unique attribute. The taxonomic graph would collapse to a tree. There would be a unique edge that linked each AIC to each of its children. When the parent PFC is not diagonal, the formal concepts formed from the PFC create a lattice that links parents to their children. For a taxonomic network, the lattice must provide a unique path from a parent to each of its children.

2.3 Preliminary Scaling Analysis of Taxonomic Classification

The number of children and the number of attributes for the PFC of a particular AIP can vary widely across the network. This variability makes it difficult to formulate simple scaling relationships for the work of computing formal concepts from the PFC's. For example, the root PFC in the ERI example has about twenty storm objects. Each storm has three attributes: storm name, date, and centroid location name. Thus, for this PFC, $N_O = 20$ and $N_A = 60$. At the next level, the PFC's have $N_O = 2$ and $N_A = 2$. The children of each PFC have only a coarse geographic bin collection and a flight path collection. At the level below this, the coarse bin PFC's may have twenty to fifty objects. The flight path PFC's have five to twenty objects. Clearly, the partitioning produces much smaller binary matrices than the unpartitioned Formal Context.

A rough approximation for scaling assumes that the graph reduces to a multiway tree. The number of children for each AIC is M. The number of levels is L. The total number of AIC nodes in the tree, N, is

$$N = 1 + M + M^2 + \ldots + M^L \tag{1}$$

If the number of attributes in each PFC is $N_A = aM$, then the size of each PFC is (M, aM). Both M and a are usually small compared with the dimensions of the Formal Context for the archive's total collection. This approximation suggests that the computational work from the network partitioning can reduce that burden by several orders of magnitude.

The archive does not have to calculate the Formal Concepts for any of the PFC's when it establishes the network. That calculation can occur when the user's navigational search reaches an AIC. Under this simplifying assumption, a user with a successful navigation search will only select L formal contexts. The archive only needs to calculate Formal Concepts for each of the selected PFC's. User attribute pruning to remove attributes the user regards as irrelevant will further reduce the computational load.

Disciplinary users familiar with a taxonomy form a much smaller community than the public using commercial search engines. For example, only about 1% to 2% of students entering U.S. colleges want to enter scientific or mathematical curricula. The reduction in computational burden and in user search requests should make the taxonomic partitioning approach computationally acceptable.

3 Disciplinary Specialist Knowledge of a Taxonomy

A disciplinary specialist, such as a research scientist or resource manager, spends a substantial amount of time learning a discipline's taxonomy for classifying objects. Such a specialist is likely to have a specific target for a search. Thus, such a user seems highly likely to use the discipline's taxonomy classification.

A biologist who has a new, unidentified plant would usually use a biological taxonomy to see if there were any previously identified specimen's in an archive's collection that matched her new one. Even if the biologist moved rapidly through the upper levels of the collection, she would eventually get to a level where she would need to match the attributes of her sample against the standard attributes that identify the particular specie within a probable Genus.

The search behavior of a disciplinary specialist differs markedly from that of an individual with an unclear target for his or her search. A person *browsing* needs a recommender site rather than a navigational one [4].

4 User-Guided Navigational Searches

The taxonomic classification approach expects that a user can identify a useful IO based on the taxonomy network traversal. If the user obtains his or her goal, then the search terminates successfully. If the user recognizes that the search is not likely to reach the target, he or she can back up to a higher level and make different selections. Thus, we expect the user search to be iterative. In addition, the user can get tired of searching and terminate the interaction. The following pseudo code outlines the search algorithm.

User-Guided Navigational Search Algorithm

```
1    $Level$ := 0; AIP := Root; Done := False;
2    while not Done loop
3        User prunes AIP attributes;
4        Compute all Formal Concepts for the pruned Formal Contex;
5        Construct Nearest Superset Navigation (NSN) graph;
6        User navigates through the graph to a Candidate AIP
             at $Level + 1$; AIP := Candidate AIP; $Level := Level + 1$;
7        if Candidate AIP is target AIU then
8            Done := True; −−Success
9        elsif User judges the search unlikely to reach target then
10           AIP := Previous AIP; $Level := Level - 1$; −−Back up
11       elsif User is tired of search then
12           Done := True; −−Abandon Search
13       end if;
14   end loop;
```

The user-guided navigational search algorithm outlined in the pseudo code starts at the root AIP. The loop moves down the taxonomic AIP graph from the root at Level 0 to deeper levels where the user can find the desired target IO. The first step for the user is to prune the selected AIP attributes to create a pruned formal context. In pruning, the user removes attributes regarded as irrelevant to finding the target IO. Pruning the static Formal Context removes the irrelevant columns and thereby produces a smaller Formal Context. The pruning also removes any rows that contain no objects after removal of the attribute columns. The pseudo code for the NSN subalgorithm provides the logic for lines 5 and 6 of this listing.

Nearest Superset Navigation (NSN) Subalgorithm

```
Order Pruned Context objects into layers based on increasing number
   of attributes;
Construct a directed graph with objects identified as vertices
   and edges that connect vertices to their immediate successors
   in the layer with the smallest increase in number of attributes;
Construct a web site in which each page contains information on
   a single vertex and has links to the immediate successors;
Search Successful := False; Done := False;
Select page with the vertex that has no attributes and all objects;
while not Done loop
    User selects pages from the links on the current page;
    if selected page has only one IO then
        if IO is User Target then
            Search Successful := True;
            Done := True;
        else
            Done := True;
        end if;
    end if;
end loop;
```

5 Conclusion

This paper shows how a taxonomic classification with FCA can fit within the OAIS RM's standard structure for information packages in an Archive of Earth science data. The scaling analysis of the computational load of user-guided navigation and dynamic FCA needs refinement. Even so, it suggests that this approach is affordable. [2] shows that a navigational approach similar to one derived from this algorithm offers an alternative to the conventional metadata query approach for selecting IO's from an archive of Earth science data.

6 Acknowledgements

The author gratefully acknowledges helpful discussions of this paper's contents with Dr. Mike Folk, Ms. Beth Huffer, Dr. Nancy Hoebelheinrich, and Mr. Gustaf Barkstrom.

References

1. CCSDS: *Reference Model for an Open Archival Information System (OAIS): Recommended Practice; CCSDS 650.0-M-2* (2012) CCSDS Secretariat, Washington.
2. NOAA: Emergency Response Imagery (2013) URL: `http://storms.ngs.noaa.gov/eri_page/index.html`
3. Britton, N. and Brown, A.: *An Illustrated Flora of the Northern United Sates and Canada: in Three Volumes.* Dover Publications, Mineola, NY. (1970)
4. Agarwal, D., Chen, B-C., Elango, P., and Ramakrishnan, R.: Content Recommendation on Web Portals. *Comm. ACM*, **5**6 (6) (2013) 92-101

A Collaborative Approach for FCA-Based Knowledge Extraction

My Thao Tang and Yannick Toussaint

Loria-Campus Scientifique,
BP 239 - 54506 Vandoeuvre-les-Nancy Cedex, France
`{My-Thao.Tang,Yannick.Toussaint}@loria.fr`

Abstract. In this paper, we propose an approach for an FCA-based knowledge extraction process relying on the collaboration between humans and machines. Evaluation of the results is performed on the lattice by experts through an interactive process where they may specify their wishes for changes using a set of predefined operations. Thus, the system then may suggest several strategies to reach their goal. In such an interactive and iterative process, the system converges towards a knowledge model close to the experts' needs. We illustrate the process on a small preliminary experiment.

Keywords: Formal Concept Analysis, Knowledge Extraction, Expert Interaction

1 Introduction

Several approaches ([1–4]) showed how powerful is Formal Concept Analysis (FCA) ([5]) for knowledge modelling and ontology design, and the lattice is usually considered as the knowledge model. This paper focuses on providing domain experts with capabilities to control and customize the lattice using retro-engineering operations. Our approach is motivated by a desire of keeping a trace between text sources and concepts of the resulting ontology. Objects and properties in texts are annotated and used to build the formal context. Annotations and the lattice evolved simultaneously thanks to retro-engineering. In this way, we can keep a trace between the linguistic level and the conceptual level making one two separated processes in literature, namely **semantic annotation** which identifies concepts from an ontology in texts and **ontology building** which builds concepts from texts.

FCA is a bottom-up process which builds concepts from a set of instances of objects and properties (the formal context). To improve the lattice and to make it closer to expert needs, we want to define an iterative and interactive process where experts are asked, at each loop, to evaluate the "quality" of the concepts. Unfortunately, in Knowledge Engineering, building ontology is not a straightforward process and usually results from trial and error process. There are several reasons for experts to ask for changes in the lattice: (1) there may be noise in resources or in the information extraction processes and thus, some concepts result from this noise, (2) experts are not satisfied with the resulting knowledge model and wish it to be more in accordance with their needs, *i.e.* the application that will use the knowledge model.

The rest of the paper is structured as follows. Firstly, section 2 explains how the system and experts collaborate in building and changing the lattice, illustrates the approach by removing a concept from a lattice. Next, section 3 presents experimental results. Finally, section 4 concludes with a summary and draws perspectives.

2 Collaboration for Changing A Lattice

In our approach, experts do not have access to the formal context; they can browse concepts, run through subsumption paths, look at extents and intents of concepts. Then, they express their wishes to change the lattice using operations. An operation on the lattice is a kind of retro-engineering: experts select an operation on the lattice (ex: remove this concept), the system assesses the impact of the change, suggests different strategies and then, for the chosen strategy, calculates what to change in data for the new lattice to meet the requirements. The lattice is then built accordingly. Experts repeat the process until they reach an agreement between the model and their knowledge.

2.1 Defining Operations on a Lattice

We list basic changes on a lattice to add or remove one element from the lattice (Table 1). There could be more complex operations such as merging two concepts into one, splitting one concept into two sub-concepts, creating a common concept for a set of concepts... For each operation, we need to define an algorithm to compute the set of possible strategies to perform the change, to identify related changes in the lattice, to rank them according to a cost function and to keep a trace in a history. Adding or removing objects or properties in a formal context motivated the development of incremental algorithms to update a lattice [6–9]. Concept removing is somehow complex: several strategies are possible and impacts on other concepts in the lattice (side effects). We detail this operation to explain our approach in the next part.

2.2 Removing a Concept from a Lattice

Removing a concept while keeping the lattice structure means moving it down to one of its children or up to one of its parents. These solutions correspond to what we call different *strategies*.

Strategy 1 moves concept C down to one of its children, C_{child}. The relation I of the formal context should be modified to I^* as follows:

$I^* = I \cup \{(g, m) : m \in \{\mathtt{Intent(C_{child})} \setminus \mathtt{Intent(C)}\}, g \in \{\mathtt{Extent(C)} \setminus \mathtt{Extent(C_{child})}, gIm\}$.

	Add	**Remove**
Object	*Add an object to a lattice*	*Remove an object from a lattice*
	Add an object to a concept	*Remove an object from a concept*
Attribute	*Add an attribute to a lattice*	*Remove an attribute from a lattice*
	Add an attribute to a concept	*Remove an attribute from a concept*
	Add an attribute to an object	*Remove an attribute from an object*
Concept	*Add a concept to a lattice*	*Remove a concept from a lattice*
	Add a supper concept to a concept	*Remove a supper concept from a concept*
	Add a sub concept to a concept	*Remove a sub concept from a concept*

Table 1: Basic changes in a lattice.

Strategies	**NOMC**	**NODC**	**NONC**
1.1 Move c_4 down to c_7	2	2	2
1.2 Move c_4 down to c_9	2	2	2
2.1 Move c_4 up to c_0	0	3	0

Table 2: Strategies for removing concept c_4 associated with the number of modified concepts (NOMC), the number of deleted concepts (NODC) and the number of new concepts (NONC) between the current lattice and the new one.

Strategy 2 moves concept C up to one of its parents, C_{parent}. The relation I of the formal context should be modified to I^* as follows:

$I^* = I \setminus \{(g,m) : m \in \{\mathtt{Intent(C)} \setminus \mathtt{Intent(C_{parent})}\}, g \in \{\mathtt{Extent(C)}\}, gIm\}$.

Table 2 shows an example of strategies for removing concept c_4 from the initial lattice given in Fig. 1. Here, we have three strategies, two for moving c_4 down and one for moving c_4 up. In *strategy 1.1*, $c_4 = (\{\mathtt{F}, \mathtt{HD}\}, \{\mathtt{CoWi\_D}\})$ and $c_7 = (\{\mathtt{F}\}, \{\mathtt{Ca\_RAS}, \mathtt{CoWi\_D}\})$. To move concept c_4 down to concept c_7, the attribute in set $\{\mathtt{Intent(c_7)} \setminus \mathtt{Intent(c_4)}\} = \{\mathtt{Ca\_RAS}\}$ is added to the object in set $\{\mathtt{Extent(c_4)} \setminus \mathtt{Extent(c_7)}\} = \{\mathtt{HD}\}$. Similarly, in *strategy 2.1*, to move concept c_4 up to concept c_0 with $\mathtt{Intent(c_0)} = \emptyset$, the attribute `CoWi_D` is removed from the objects in set $\{\mathtt{F}, \mathtt{HD}\}$.

We benefit from incremental algorithms that have been implemented to build lattices ([6], [8]) not only to avoid complete recalculation of the lattice but also for identifying the different possible strategies to perform a given change in the current lattice. Incremental algorithms, at the i^{th} step, produce the concept set or diagram graph for i first objects of the context. The new object $i+1^{th}$ is dynamically added by modifying the existing lattice without recomputing the whole lattice from scratch. ([8])

Let us explain our approach through the algorithm for moving concept down - *Strategy 1*. In *Strategy 1*, the set of objects and the set of properties remain unchanged. Only the relation I is enriched. Moreover, a new closed set $(\mathtt{Extent(C)}, \mathtt{Intent(C_{child})})$ is a formal concept of the new lattice $\mathcal{L}^*$.

From the new closed set $(\mathtt{Extent(C)}, \mathtt{Intent(C_{child})})$ and the existing lattice $\mathcal{L}$, we generate the new lattice $\mathcal{L}^*$ by identifying the categories of concepts. The new lattice is obtained from the existing lattice by taking, deleting or modifying some concepts and creating new concepts. Thus, concepts are classified into four

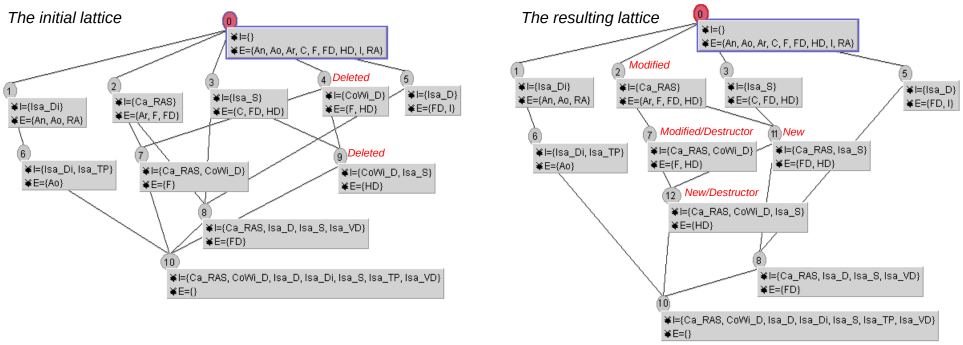


Fig. 1: The initial lattice and the resulting lattice after removing concept c_4 according to the strategy moving concept c_4 down to concept c_7.

categories, *old* concepts, *deleted* concepts, *modified* concepts and *new* concepts as follows. Let C be a concept in $\mathcal{L}^*$:

- C is an *old* concept if there exists a concept with the same `Extent(C)` and `Intent(C)` in $\mathcal{L}$.
- C is a *modified* concept if there exists a concept with the same `Intent(C)` in $\mathcal{L}$ but the extent is different from `Extent(C)`.
- C is a *new* concept if `Intent(C)` doesn't exist in $\mathcal{L}$.
- C in $\mathcal{L}$ is a *deleted* concept if `Intent(C)` doesn't exist in $\mathcal{L}^*$.

Moreover, we identify *destructors*. A *destructor* is a concept that a *deleted* concept will jump to. Thus links to the *deleted concepts* should be reported to *destructors*.

Fig. 1 shows an example of removing concept c_4 according to the strategy *moving it down to concept* c_7. Here, two concepts, c_4 and c_9, in the initial lattice, are *deleted*; two concepts, c_2 and c_7, are *modified*, adding the object `HD` to their extent; two *new* concepts, c_{11} and c_{12}, are created. In addition, concept c_7 is *destructor* of the *deleted* concept c_4 and concept c_{12} is *destructor* of the *deleted* concept c_9.

The algorithm calculating the new lattice is based on the fundamental property of lattices ([10]). For any closed set (X, X') in the lattice:

$$X' = f(X) = \bigcap_{x \in X} f(\{x\}) \qquad \text{and } X = g(X') = \bigcap_{x' \in X'} g(\{x'\})$$

Moreover, for any set of $f(x)$ (resp. $g(\{x'\})$), their intersection should be in the lattice.

The main process of the algorithm is to update the set of intersections of extents. We perform a top-down traversal of the lattice $\mathcal{L}$ to identify whether a concept is *old*, *modified* or *deleted*, to create *new* concepts and to keep the information of *destructors*.

Thanks to the categories of concepts, the resulting lattice can be calculated and we can keep a trace from the previous lattice. By this way, we can know the impact of a change on the lattice (lists of modified, deleted and new concepts).

Finally, all the strategies are suggested to the expert with their impacts (Table 2) so that the expert can consider choosing one strategy. Once the expert makes a choice, the system will update the lattice accordingly.

3 Experiment of Removing A Concept

We experimented our approach for extracting knowledge from a real text corpus of *Fibromuscular Dysplasia* extracted from PUBMED[1]. The corpus consists of 400 texts.

In the preprocessing step, we used SEMREP ([11]) to annotate the texts. 2402 annotation objects were identified from the corpus, of which only 668 objects are shown in the right or left side of a triplet relationship and 481 objects on the right side. 36 different relationships are identified in these texts. Given an annotation (`object1`, `relationship`, `object2`), we consider `object1` as an object and the name of the relation is concatenated with `object2` to become a binary attribute of `object1`. For example, `Fibromuscular Dysplasia` is an object and `ISA_Disease` is one of its attributes. SEMREP annotation process can be noisy but, of course, no automatic tool is perfect. Moreover, any annotation process could annotate the texts, including manual annotation. This is one of our goals to take the advantage of expert interaction in the construction of the knowledge model.

Next, we built a Java script to transform the set annotations into a formal context. The lattice was produced by Galicia[2] and exported as the XML document. The context built from our corpus contains 481 objects, 545 attributes formed by the association of the relationship of the object with which it relates. The lattice contains 523 concepts and the longest path from the *Top* to the *Bottom* is 7.

Then, we implemented a system with operation removing a concept from a lattice. When an expert, during the evaluation phase, asks for removing a concept, the system suggests a list of strategies and makes explicit their impacts on the lattice (lists of modified, deleted and new concepts) to the expert. Once the expert chooses one strategy, the system will update the lattice accordingly.

The required time for the expert to remove concepts has not yet been evaluated. Our experience shows that a judicious choice of a strategy in order to remove concepts has a strong impact on the speed of convergence towards a satisfactory knowledge model.

4 Conclusion and Perspective

We presented a collaborative approach for FCA-based knowledge extraction. Our study shows how domain experts and machines can collaborate in building and changing a lattice. The system handles changes in a lattice and keeps a trace

---

[1] http://www.ncbi.nlm.nih.gov/pubmed

[2] http://sourceforge.net/projects/galicia/

from the previous lattice. In that way, experts know the impact of a change. Our iterative and interactive approach takes advantage from FCA and reduces limitations due to the use of a formal method to model a complex cognitive process.

This approach opens several perspectives. We can do more to help the expert. Refining the cost function associated to changes can make easier the choice of one strategy. Tagging concepts that the expert agrees with and wants to keep unchanged all along the process could reduce the number of the suggested strategies. Performing several changes at once needs also more investigations.

References

1. Bendaoud, R., Napoli, A., Toussaint, Y.: Formal concept analysis: A unified framework for building and refining ontologies. In Gangemi, A., Euzenat, J., eds.: EKAW. Volume 5268 of Lecture Notes in Computer Science., Springer (2008) 156–171
2. Obitko, M., Snásel, V., Smid, J.: Ontology design with formal concept analysis. In: Proceedings of the CLA 2004 International Workshop on Concept Lattices and their Applications, Ostrava, Czech Republic, September 23-24, 2004. (2004)
3. Cimiano, P., Völker, J.: Text2onto - a framework for ontology learning and data-driven change discovery. In Springer, ed.: Proceedings of Natural Language Processing and Information Systems (NLDB 2005). Number 3513 in Lecture Notes in Computer Science (LNCS) (June 2005) 227–238
4. Huchard, M., Napoli, A., Rouane, M.H., Valtchev, P.: A proposal for combining formal concept analysis and description logics for mining relational data. In: proceeding of the 5th International Conference Formal Concept Analysis (ICFCA'07), Clermond-Ferrand, France (2007)
5. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical Foundations. Springer (1999)
6. Godin, R., Missaoui, R., Alaoui, H.: Incremental concept formation algorithms based on Galois (concept) lattices. Computational Intelligence **11**(2) (May 1995) 246–267
7. Valtchev, P., Missaoui, R.: Building galois (concept) lattices from parts: Generalizing the incremental approach. In Delugach, H., Stumme, G., eds.: Proceedings of the ICCS 2001. Volume 2120 of LNCS., Springer Verlag (2001) 290–303
8. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. J. Exp. Theor. Artif. Intell. **14**(2-3) (2002) 189–216
9. Merwe, D., Obiedkov, S., Kourie, D.: Addintent: A new incremental algorithm for constructing concept lattices. In Eklund, P., ed.: Concept Lattices. Volume 2961 of Lecture Notes in Computer Science. Springer, Berlin/Heidelberg (2004) 205–206
10. Barbut, M., Monjardet, B.: Ordre et classification, Algèbre et combinatoire. Hachette, Paris (1970)
11. Rindflesch, T.C., Fiszman, M.: The interaction of domain knowledge and linguistic structure in natural language processing: Interpreting hypernymic propositions in biomedical text. Journal of Biomedical Informatics **36**(6) (2003) 462–477

Towards Description Logic on Concept Lattices*

J.V. Grebeneva, N.V. Shilov, and N.O. Garanina

A.P. Ershov Institute of Informatics Systems,
Lavren'ev av., 6, Novosibirsk 630090, Russia `j.grebeneva@gmail.com`,
shilov@iis.nsk.su, garanina@iis.nsk.su

Abstract. In this position paper we start with a motivation of our study of modal/description logics with values in concept lattices. Then we give a brief survey of approaches to lattice-valued modal and/or description logics. After that we study some methods of context symmetrization, because the description logic on concept lattices is defined for symmetric co ntexts only. We conclude with a list of problems related to comparison of different lattice-valued modal/description logics, different variants of context symmetrization and resulting description logics, decidability and axiomatization of these logics.

1 Introduction

Let us start with an example that can explain our interest to study of polymodal and/or description logics with values in concept lattices. For it let us first fix some moment of time and let (1) URL be the set of all Uniform Resource Locator that are valid (exist) at this moment, (2) Key be the set of all Key-words in any existing language that are conceivable in this time, (3) F, S and T be binary relations on $URL \times Key$ that are implemented in some (non-real we assume) search engines First, Second and Third at the same moment of time that we fixed above.

Then let *Sh&Ga* be the set of all web-sites (their URL's hereinafter) that a search engine First finds by two key-words *Shilov* and *Garanina*; In terms of Formal Concept Analysis (FCA) [4] $Sh\&Ga = \{Shilov, Garanina\}'$ in the following formal context $\mathbb{F} = (URL,\ KW,\ F)$. Similarly, let Gr — be the set of all web-sites that Second finds by searching for a single key-word *Grebeneva*; in FCA terms one can write $Gr = \{Grebeneva\}'$ in the next one formal context $\mathbb{S} = (URL,\ Key,\ S)$.

Assume that we need to know all sites $Sh\&Ga \setminus Gr$ that are found by First by key-words *Shilov* and *Garanina* but that (according to Third) does not contain any word that is common for all sites that are found by Second for the key-word *Grebeneva*. In terms of set theory expanded by FCA-derivatives the desired set can be written as $URL_{Sh\&Ga} \setminus URL''_{Gr}$, where '$'$'

---

* This research is supported by Integration Grant *Mathematical and methodological aspects of intellectual Information Systems* n.15/10 of Siberian Branch, Russian Academy of Science.

represents derivative in the formal context $\mathbb{T} = (URL,\ Key,\ F)$. Recall that $URL_{Sh\&Ga} = \{Shilov, Garanina\}'$ in $\mathbb{K}_G$, $URL_{Gr} = \{Grebeneva\}'$ in $\mathbb{K}_Y$. So we can not write the following equality

$$URL_{Sh\&Ga\setminus Gr} = \{Shilov, Garanina\}' \setminus \{Grebeneva\}''',$$

but have to write another one

$$URL_{Sh\&Ga\setminus Gr} = \{Shilov, Garanina\}^{\downarrow_F} \setminus \{Grebeneva\}^{\downarrow_S\uparrow_T\downarrow_T},$$

where '$\downarrow_F$' represents the lower derivative in the context $\mathbb{F}$, '$\downarrow_S$' — the lower derivative in the context $\mathbb{S}$, and '$\downarrow_T$' and '$\uparrow_T$' — the lower and upper derivatives in the context $\mathbb{T}$. We believe that it would be nice to process queries like this one but (unfortunately) modern search engines can not do it; a part of the reason for this inability is due to lack of theory for processing such multi-context queries.

At the same time polymodal and/or descriptive logics (DL) [1] provide language for presentation of queries as above. In particular, if T^d denotes the inverse of the binary relation T, then $Sh\&Ga \setminus Gr$ may be represented in syntax of a polymodal logic by the following formula

$$[F](Shilov\&Garanina)\ \&\ \neg[T][T^d][S]Grebeneva$$

or in syntax of a description logic as the following concept term

$$\forall F.(Shilov \sqcap Garanina)\ \sqcap\ \neg\forall T.\forall T^d.\forall S.Grebeneva.$$

An interpretation of FCA constructs in DL has been studied in [7]. In these studies DL has been extended by FCA-derivatives and provided with Kripke semantics; as a result all concept terms are interpreted by sets of objects, not by concepts (or their extents), a lattice-theoretic structure of formal concepts (that is so important advantage of FCA) is lost.

A variant of description logic (namely $\mathcal{ALC}$, Attribute Language with Complements) with values in concept lattices was defined in [8] but without a concept negation; the concept negation was defined only in concept lattices for symmetric contexts (i.e. in contexts where sets of objects and attributes are equal and the binary relation is symmetric). It implies that if we would like to define concept lattice semantics for the DL concept term above, we have to symmetrize contexts $\mathbb{F}$, $\mathbb{S}$ and $\mathbb{T}$ in some manner. In this position paper we present some preliminary results of our studies of ways of context symmetrization, formulate and discuss some topics that need more research.

2 Lattice-valued Modal and Description Logics

Modal and Description Logic are closely related but different research paradigms: they have different syntax and pragmatic, but very closely related semantics (in spite of different terminology). Lattice-valued modal logics were introduced in

[3, 2] by M.C. Fitting. They were studied in the cited papers from a proof-theoretic point of view. Later several authors attempted study of these logics from algebraic perspective [5, 6, 9]. Basic definitions related to modal logics on lattices and completeness theorems can be found in [5].

Description Logic (DL) is a logic for reasoning about concepts. But there is also an algebraic formalism developed around concepts in terms of concept lattices, namely Formal Concept Analysis (FCA). In this section we recall in brief definition of description logic $\mathcal{ALC}$ on concept lattices of (symmetric) contexts and some properties that follow from this definition, please refer [8] for full details. We use notation and definitions for Description Logics from [1][1]. For the basics and notation of Formal Concept Analysis, please, refer to [4].

Semantics of description logics on concept lattices comes from lattice-theoretic characterization of 'positive' (i.e. without negation) concept constructs (for close world semantics) that is given in the following proposition [8].

Proposition 1. *Let* (Δ, Υ) *be a terminological interpretation and* $P(\Delta) = (2^{\Delta}, \varnothing, \subseteq, \Delta, \cup, \cap)$ *be the complete lattice of subsets of* Δ*. Then semantics of* $\mathcal{ALC}$ *positive concept constructs* $\top$, $\bot$, $\sqcup$, $\sqcap$, $\forall$, *and* $\exists$ *enjoys the following properties in* $P(\Delta)$*: (1)* $\Upsilon(\top) = \sup P(\Delta)$*, and* $\Upsilon(\bot) = \inf P(\Delta)$*; (2)* $\Upsilon(X \sqcup Y) = \sup(\Upsilon(X), \Upsilon(Y))$*, and* $\Upsilon(X \sqcap Y) = \inf(\Upsilon(X), \Upsilon(Y))$*; (3)* $\Upsilon(\forall R.\, X) = \sup\{S \in P(\Delta) : \forall s \in S \forall t \in \Delta((s,t) \in \Upsilon(R) \Rightarrow t \in \Upsilon(X))\}$*, (4)* $\Upsilon(\exists R.\, X) = \sup\{S \in P(\Delta) : \forall s \in S \exists t \in \Delta((s,t) \in \Upsilon(R) \;\&\; t \in \Upsilon(X))\}$*.*

Conceptual interpretation is a formal context provided by an interpretation function.

Definition 1.
Conceptual interpretation is a four-tuple (G, M, I, Υ) where (G, M, I) is a formal context, and an interpretation function $\Upsilon = I_{CS} \cup I_{RS}$, where CS and RS are standard concept and role symbols, and (1) $I_{CS} : CS \to \mathfrak{B}(G, M, I)$ maps concept symbols to formal concepts, (2) $I_{RS} : RS \to 2^{(G\times G)\cup(M\times M)}$ maps role symbols to binary relations. A formal context (G, M, I) or conceptual interpretation (G, M, I, Υ) is said to be homogeneous (symmetric) if $G = M$ (and binary relation I is symmetric in addition).

Semantics of $\mathcal{ALC}$ positive concept constructs $\top$, $\bot$, $\sqcup$, $\sqcap$, $\forall$, and $\exists$ as well as semantics of negative construct $\neg$ are defined in [8] as follows.

Definition 2.
Let (G, M, I, Υ) be a conceptual interpretation, $\mathbb{K}$ be a formal context (G, M, I), and $\mathfrak{B} = (\mathbb{K})$ be the concept lattice of $\mathbb{K}$. The interpretation function Υ can be extended to all role terms in a terminological interpretation $((G \cup M), \Upsilon)$ in the standard manner so that $\Upsilon(R)$ is a binary relation on $(G \cup M)$ for every role term R. The interpretation function Υ can be extended to all positive $\mathcal{ALC}$ concept terms as follows. (1) $\Upsilon(\top) = \sup \mathfrak{B}$ and $\Upsilon(\bot) = \inf \mathfrak{B}$; (2) $\Upsilon(X \sqcup Y) = \sup(\Upsilon(X), \Upsilon(Y))$, and $\Upsilon(X \sqcap Y) = \inf(\Upsilon(X), \Upsilon(Y))$; (3) Let $\Upsilon(X) =$

---

[1] But we use Υ instead of '$\cdot$' for terminological interpretation function for readability.

$(Ex', In') \in \mathfrak{B}$. Then (a) $\varUpsilon(\forall R.\ X) = \sup_K\{(Ex, In) \in \mathfrak{B} : \forall o \in Ex\ \forall a \in In\ \forall o' \in G\ \exists a' \in M\ ((o, o') \in \varUpsilon(R) \Rightarrow o' \in Ex',\ (a, a') \in \varUpsilon(R)$, and $a' \in In')\}$, (b) $I(\exists R.\ X) = \sup_K\{(Ex, In) \in \mathfrak{B} : \forall o \in Ex\ \forall a \in In\ \exists o' \in G\ \forall a' \in M\ ((a, a') \in \varUpsilon(R) \Rightarrow (o, o') \in \varUpsilon(R),\ o' \in Ex'$, and $a' \in In')\}$. In addition, if $\mathbb{K}$ is a symmetric context and $\varUpsilon(X) = (Ex, In) \in \mathfrak{B}$, then $\varUpsilon(\neg X) = (In, Ex)$.

The following proposition [8] states that for any conceptual interpretation every positive $\mathcal{ALC}$ concept term is an element of concept lattice; in addition, if an interpretation is symmetric then this fact holds for all $\mathcal{ALC}$ concept terms.

Proposition 2. *For any conceptual interpretation* $(G, M, I, \varUpsilon)$*, for every positive* $\mathcal{ALC}$ *concept term* X*, semantics* $\varUpsilon(X)$ *is an element of* $\mathfrak{B}(G, M, I)$*. For any symmetric conceptual interpretation* $(D, D, I, \varUpsilon)$*, for every* $\mathcal{ALC}$ *concept term* X*, semantics* $\varUpsilon(X)$ *is an element of* $\mathfrak{B}(D, D, I)$*.*

3 Ways to Build a Symmetric Context

The above proposition 2 leads to the following idea: to define semantics of $\mathcal{ALC}$ with values in an arbitrary concept lattice by isomorphic embedding of the background context into a symmetric one in such a way that for the positive fragment of $\mathcal{ALC}$ the original semantics and the induced semantics equal each other. Below we examine some opportunities how to "symmetrize" a given context, i.e. to build a symmetric context from an arbitrary given background context. Below we are going to study how to build a symmetric context from a given one by set-theoretic and algebraic manipulations with a binary relation of the context. Without loss of generality we may assume that the background context is reduced [4] and has disjoint sets of objects and attributes.

Let $\mathbb{K} := (G, M, I)$ be a reduced context where $G \cap M = \varnothing$ and $\mathbb{K}^d = (M, G, I^-)$ be its dual context. Let also use the following notation for binary relations (on M and/or G): (1) $\varnothing$ be the empty binary relation, (2) $\times$ be a total binary relation, (3) E be the identity binary relation, (4) E^c be the complement for E. We would like to combine the cross-tables of $\mathbb{K}$ and the dual context $\mathbb{K}^d$ into the symmetric one in the following way: $\begin{array}{c|cc} & G & M \\ \hline G & ? & I \\ M & I^{-1} & ? \end{array}$. Let us represent the above cross-table in a shorter way as $\begin{array}{c|c} ? & \mathbb{K} \\ \hline \mathbb{K}^d & ? \end{array}$ and denote the corresponding symmetric context by $\mathbb{K}_\circ := (G_\circ, M_\circ, I_\circ)$. Recall that $\mathfrak{B}(G, M, I)$ is the concept lattice of the context $\mathbb{K}$, $\mathfrak{B}(G_\circ, M_\circ, I_\circ)$ is the concept lattice of the context $\mathbb{K}_\circ$. Let us use the standard notation '$'$' for derivatives in the background context $\mathbb{K}$ but (for distinction) the notation '$^\circ$' for derivatives in the symmetric one. We are going to fill question quadrants by different combinations of $\varnothing$, $\times$, E and E^c. Below we study 9 of these 16 combinations.

CASE 1. $\begin{array}{c|c} \varnothing & \mathbb{K} \\ \hline \mathbb{K}^d & \varnothing \end{array}$. It is the disjoint union of $\mathbb{K}$ and $\mathbb{K}^d$. The concept lattice $\mathfrak{B}(\mathbb{K}_\circ) = \mathfrak{B}(\mathbb{K} \mathbin{\dot\cup} \mathbb{K}^d)$ is a *horizontal sum* [4], i.e. the union of two sublattices $\mathfrak{B}(\mathbb{K})$ and $\mathfrak{B}(\mathbb{K}^d)$, such that $\mathfrak{B}(\mathbb{K}) \cap \mathfrak{B}(\mathbb{K}^d) = \{\perp, \top\}$.

Case 2. $\begin{array}{c|c} \times & \mathbb{K} \\ \hline \mathbb{K}^d & \varnothing \end{array}$. The concept lattice of this context is isomorphic to the *vertical sum* [4] of the concept lattices $\mathfrak{B}(\mathbb{K}^d)$ and $\mathfrak{B}(\mathbb{K})$ (where the concept lattice $\mathfrak{B}(\mathbb{K}^d)$ is upper than $\mathfrak{B}(\mathbb{K})$).

Case 3. ($\varnothing$,$\times$). This case is the same like a previous, but we have to swap components of the vertical sum.

Case 4. $\begin{array}{c|c} \times & \mathbb{K} \\ \hline \mathbb{K}^d & \times \end{array}$. We have here the direct sum $\mathbb{K} + \mathbb{K}^d$ of contexts $\mathbb{K}$ and $\mathbb{K}^d$ [4] and the concept lattice of the sum is isomorphic to the product of the concept lattices $\mathfrak{B}(\mathbb{K}) \times \mathfrak{B}(\mathbb{K}^d)$. A pair (A, B) is a concept of the direct sum $(\mathbb{K} + \mathbb{K}^d)$ iff $(A \cap G, B \cap M)$ is a concept of $\mathbb{K}$ and $(A \cap M, B \cap G)$ is a concept of $\mathbb{K}^d$. It implies that isomorphism is given by $(A, B) \mapsto ((A \cap G, B \cap M), (A \cap M, B \cap G))$.

Case 5. $\begin{array}{c|c} E & \mathbb{K} \\ \hline \mathbb{K}^d & E \end{array}$. Let $(X, Y) \in \mathfrak{B}(\mathbb{K}_\circ)$ be a concept and let $X = A \dot{\cup} B$ where $A \subseteq G$, $B \subseteq M$. We have the following cases:
(1) $B = \varnothing$. Let $X = A$. If $|A| = 1$, then $(X, Y) = (\{a\}, \{a\} \cup A')$, and $(X, Y) = (A, A')$ otherwise.
(2) $A = \varnothing$. Let $X = B$. If $|B| = 1$, then $(X, Y) = (\{b\}, \{b\} \cup B')$, and $(X, Y) = (B, B')$ otherwise.
(3) $|B| = 1$ and $A \neq \varnothing$. Let $X = A \cup \{b\}$. If $\{b\} \in A'$ and $|A| = 1$, then $(X, Y) = (\{a\} \cup \{b\}, \{a\} \cup \{b\})$, and if $\{b\} \in A'$ $|A| > 1$, then $(X, Y) = (A \cup \{b\}, \{b\})$.
(4) $|B| > 1$ and $A \neq \varnothing$. Let $|B| > 1$, then $X = A \cup B$. If $B \subseteq \{a\}'$ and $|A| = 1$, then $(X, Y) = (\{a\} \cup B, \{a\})$.

Case 6. $\begin{array}{c|c} \varnothing & \mathbb{K} \\ \hline \mathbb{K}^d & E \end{array}$. This case is a very similar to the previous one.
(1) $B = \varnothing$. $(X, Y) = (A, A')$.
(2) $A = \varnothing$. If $|B| = 1$ then $(X, Y) = (\{b\}, \{b\} \cup B')$ else $(X, Y) = (B, B')$.
(3) $|B| = 1$. If $\{b\} \in A'$, we have $(X, Y) = (A \cup \{b\}, \{b\})$.
(4) $|B| > 1$. All the concepts in this case will be either $\top$ or $\bot$.

Case 7. (E,$\varnothing$). This case is similar to the previous.

Case 8. $\begin{array}{c|c} \times & \mathbb{K} \\ \hline \mathbb{K}^d & E \end{array}$. Let us use subcases as in the case 5.
(1) $B = \varnothing$. $(X, Y) = (A, G \cup A')$.
(2) $A = \varnothing$. If $|B| = 1$, then $(X, Y) = (\{b\}, \{b\} \cup B')$, and $(X, Y) = (B, B')$ otherwise.
(3) $|B| = 1$. If $\{b\} \in A'$, then $(X, Y) = (A \cup \{b\}, \{b\} \cup \{b\}')$, else $(X, Y) = (A \cup \{b\}, \{b\}')$.
(4) $|B| > 1$. $(X, Y) = (A \cup B, B')$.

Case 9. (E,$\times$). This case is similar to the case 8.

4 Conclusion

Now we are ready to formulate several topics and problems that we consider natural and important for further research.

In [5] the definition for modal logics with values in a given finite distributive lattice L is presented. This definition is easy to expand on polymodal logics. In section 2 we represented the definition for description logic $\mathcal{ALC}$ (that can be considered as a polymodal version of $\mathbf{K}$) with values in concept lattices of symmetric contexts. Assume that $\mathbb{K}$ is a finite symmetric context; then $\mathfrak{B}(\mathbb{K})$ is a finite lattice, but it may not be a distributive lattice. Question: Assuming that $\mathfrak{B}(\mathbb{K})$ is a finite distributive lattice, whether $\mathcal{ALC}$ with values in $\mathfrak{B}(\mathbb{K})$ is a polymodal $\mathfrak{B}(\mathbb{K})$-**ML**?

In section 2 we represented the definition for description logic $\mathcal{ALC}$ with values in concept lattices of symmetric contexts and for positive fragment of $\mathcal{ALC}$ with values in arbitrary concept lattices. Questions: (1) Is decidable or axiomatizable the positive fragment of $\mathcal{ALC}$ with values in concept lattices? (2) Is decidable or axiomatizable $\mathcal{ALC}$ with values in concept lattices of symmetric contexts?

In the section 3 we examine 9 of 16 variants of context symmetrization. Topics for further research are following: (1) to study the remaining 7 cases of context symmetrization and isomorphic embedding with E^c in one or two free quadrants; (2) to examine under which embedding from these in these 16 the induced semantics of the positive fragment of $\mathcal{ALC}$ is equal to the original semantics.

References

1. Baader F., D. Calvanese, D. Nardi D.McGuinness, and P. Patel-Schneider (editors). *The Description Logic Handbook: Theory,Implementation and Applications.* Cambridge University Press, 2003.
2. Fitting M.C. *Many-valued modal logics II.* Fund. Inform., 1992, v.17, p.5573.
3. Fitting M.C. *Many-valued modal logics.* Fund. Inform., 1991, v.15, p.235254.
4. Ganter B., Wille R. *Formal Concept Analysis.* Mathematical Foundations. Springer Verlag, 1996.
5. Maruyama Y. *Algebraic Study of Lattice-Valued Logic and Lattice-Valued Modal Logic.* In ICLA'09 Proceedings of the 3rd Indian Conference on Logic and Its Applications. Lecture Notes in Computer Science. Springer-Verlag Berlin, Heidelberg 2009, v.5378, p.170-184.
6. Maruyama Y. *Reasoning about Fuzzy Belief and Common Belief: With Emphasison Incomparable Beliefs.* In Proceedings of the 22nd International Joint Conference on Artificial Intelligence. AAAI Press/International Joint Conferences on Artificial Intelligence, Menlo Park, California, 2011, Vol. 2, p.1008-1013.
7. Shilov N.V. Garanina N.O., Anureev I.S. *Combining Two Formalism for Reasoning about Concepts.* Proceedings of the 2007 International Workshop on Description Logics (DL2007), Brixen Italy, 2007. CEUR Workshop Proceedings v.250. p.459-466.
8. Shilov N.V., Han S.-Y. *A proposal of Description Logic on Concept Lattices.* Proceedings of the Fifth International Conference on Concept Lattices and their Applications, 2007. CEUR Workshop Proceedings, v.331, 2008, p.165-176.
9. Shi H.-X., Wang G.-J. *Lattice-valued modal propositional logic and its completeness.* Science China, Information Sciences. 2010, v.53(11), p.2230-2239.

Computing Left-Minimal Direct Basis of implications

Pablo Cordero, Manuel Enciso, Angel Mora, and Manuel Ojeda-Aciego

University of Málaga, Spain.
e-mail: {pcordero,enbciso}@uma.es, {amora,aciego}@ctima.uma.es

Abstract. The most popular basis in Formal Concept Analysis is the Duquenne-Guigues basis, which ensure minimality in the number of dependencies and it is built with pseudo-intents, and some method to calculate these basis from an arbitrary set of implications have been introduced. We propose in this paper, an automated method to calculate a left-minimal direct basis from the set of all implications built between a closed set and its corresponding minimal generators. The new basis also has the minimal property demanded in the Duquenne-Guigues basis. It is minimal in the cardinal of the set of implications, and minimal in the size of the left-hand side of the implications.

1 Introduction

Formal Concept Analysis [8] has shown to be a powerful framework to discover knowledge inside date sets. It provides a solid and formal theory which enhances other well known approaches. The main element of FCA community are binary relations between objects and attributes, which are described using matrixes (contexts) and represent the appearance of the attributes in the corresponding objects. One outstanding element in FCA is attribute implication, which is used to specify a constraint in the context. Thus we write an implication between attribute sets X and Y in the form $X \rightarrow Y$ whenever any object in the context which has all the attributes in X also has all the attributes in Y.

Attribute implication can be managed syntactically using Armstrong's Axioms [1], a sound and complete inference system. This axiomatic system allows us to derive new attribute implications that hold in a given context. This "inference" relation leads to the following problem: How to characterize the minimal set of implications for a given set of implications? Among the different basis notions, the Duquenne-Guigues basis [9] also called stem basis seems has to be cited because of their widely acceptation in the FCA area and because of its minimality notion (w.r.t. the number of implications). Nevertheless, minimality in the number of implications is a criteria that may be enhanced.

In [5] K. Bertet and B. Monjardet provided a set of orthogonal characteristics of the basis and established the equivalence of five definitions presented by different authors in several areas which correspond with the same notion of basis.

© paper author(s), 2013. Published in Manuel Ojeda-Aciego, Jan Outrata (Eds.): CLA 2013, pp. 293–298, ISBN 978–2–7466–6566–8, Laboratory L3i, University of La Rochelle, 2013. Copying permitted only for private and academic purposes.

In [6] we present a method to compute all the closed sets and its minimal generators from a context. This information allows us to built a set of implications whose left had side is a minimal generator and with its closed set in the right hand side. We propose in this paper a definition of basis with the good minimality property of Duquenne-Guigues basis and the characteristics of the above implications: minimal information in the left had side and a fast computation of attributes closures from them.

We also introduce an automated method to calculate from a set of implications a left-minimal direct basis. The new method is based on the Simplification Logic for Funcional Dependency $\mathbf{SL}_{\text{FD}}$ [**?**], a sound and complete inference system for implications. The main characteristics of $\mathbf{SL}_{\text{FD}}$ is that its inference system is not built around the transitivity rule, like other well known Armstrong-like axiomatic systems for implications.

The work is organized as follows. In Section 2 we summarize preliminary concepts and results on FCA concerning implications, basis, etc. In Section 3 we outline the automated method that we have proposed in [6] for the computation of minimal generators and we introduce a new method to calculate the left-minimal direct basis from the original set of implications. The paper ends with a Conclusion Section.

2 Background

We will use the well-kwon notation used on Formal Concept Analysis (FCA) [8]. For the analysis of the information contained in the context $\mathbf{K} = (G, M, I)$, a direction is the study of the pair (closed sets - minimal generators). 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 $X = A$.

A relevant notion in the framework of Formal Concept Analysis is the concept of attribute implication [8]. This area is devoted to obtain knowledge from a context that is a table in which attributes and objects are related. An attribute implication is an expression $A \to B$ where A and B are sets of attributes. A context satisfies $A \to B$ if every object that has all the attributes in A has also all the attributes in B. It is well known that the sets of attribute implications that are valid in a context satisfies the Armstrong's Axioms. Although the two interpretations of formulas (functional dependency and attribute implication) are different, they have the same concept of semantic entailment. [2]

Alternatively, attribute implications allow us to capture all the information which can be deduced from a context. The set of all valid implications in a context may be syntactically managed by means of the following inference system known as Armstrong's axioms. 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 an implication basis with minimal size. This condition is satisfied by the so-called Duquenne-Guigues (or stem) basis [9]. However, the definition of the Duquenne-Guigues basis refers to minimality only in the cardinality of the set of formulas, but as we have showed in [6] with an illustrative example, redundant attributes use to appear in this kind of minimal basis.

In the following, a summary of some interesting result of this survey are showed. More specifically we present the definitions of some characteristics studied in the survey that will be used to identify the kind of basis we introduce in this paper. In the practice, it is interesting considering some properties [5] related with minimality of them, in order to achieve efficiency.

Definition 1. *A set of implications Γ it is said*

- *minimal or non-redundant if $\Gamma \setminus \{X \to Y\}$ is not equivalent to Γ.*
- *minimum if if it is of least cardinality, that is, $\mid \Gamma \mid \leq \mid \Gamma' \mid$ for all set of implications Γ' equivalent to Γ.*
- *optimal if $\| \Gamma \| \leq \| \Gamma' \|$ for all set of implications Γ' equivalent to Γ, where the size of Γ is defined as*

$$\| \Gamma \| = \sum_{\{X \to Y \in \Gamma\}} (\mid X \mid + \mid Y \mid)$$

And finally the two characteristics that constitutes the center of our basis are introduced:

Definition 2. *Let $\Gamma = \{X_0 \to Y_0, \ldots X_n \to Y_n\}$ be a set of implications, it is said a left-minimal basis if there does not exist a $X_i \to Y_i$ and a subset $X'_i \subsetneq X_i$ such that $\Gamma \setminus \{X_i \to Y_i\} \cup \{X'_i \to Y_i\}$ is equivalent to Γ.*

A set of implications Γ and is said direct if for all implication $A \to B$ the set $A \cup B$ is a closed set w.r.t. Γ.

3 Obtaining basis from minimal generators

Some methods to obtain generators of closed sets have been studied in [7, 12, 13]. Moreover, minimal generators [10, 11] appear in the literature under different names in various fields, for instance they are the minimal keys of the tables in relational databases. In [13], the authors emphasize the importance of studying minimal generators although "they have been paid little attention so far in the FCA literature".

We agree with these authors about the importance of the study of closed sets and minimal generators. They constitute a source of essential information to analyze a formal context. As we mention in the introduction, in [6] we illustrated the use of the Simplification paradigm to guide the search of all minimal generator sets. Thus, we introduce a method named `MinGen` [6] which computes a list of all pairs Φ =`<closed set, minimal generators>` from an arbitrary set of implications The goal of this paper can be considered as the reserve direction of the way we presented there. We will introduce a method to transform these set of pairs into a basis, preserving two good properties (fast computation of attribute closure and minimal left hand side in the implications) and providing minimality in the number of implications.Thus, our goal is to achieve from the minimal generators and closed sets a basis of implications fulfilling left-minimality and directness.

In the literature, the most cited algorithm to compute Duquenne-Guigues basis is the Ganter Algorithm [8]. Algorithm 1 is an adaptation of the algorithm showed in [3] to compute a Duquenne-Guigues basis from a set of LSI (labelled set of items) [6], and we obtain a Duquenne-Guigues basis.

Algorithm 1: Algorithm for computing a Duquenne-Guigues basis

```
input  : An LSI $\Phi$
output: A Duquenne-Guigues basis
$T := \emptyset$;
foreach $\langle B, mg(B)\rangle \in \Phi$ do
    foreach $A \in mg(B)$ do  $T := T \cup \{A \to B\}$
repeat
    $S := T$;
    foreach $A \to B, C \to D \in T$ such that $A \subsetneq C$ and $B \not\subseteq C$ do
        $T := T \smallsetminus \{C \to D\}$;
        if $B \cup C \neq D$ then $T := T \cup \{BC \to D\}$
until $T = S$ ;
$S := \emptyset$;
foreach $A \to B \in T$ do $S := S \cup \{A \to B \smallsetminus A\}$;
return $S$
```

In the following example, we show how to link the above algorithm with the work presented in [6].

Example 1. For the input $T = \{ab \to c, ac \to bd, b \to d, d \to c\}$, Algorithm MinGen_0 (see [6]) returns $\Phi = \{< abcd, \{ab, ac, ad\} >, < bcd, \{b\} >, < cd, \{d\} > \}$ and from here Algorithm 1 renders $\Gamma = \{d \to cd, b \to bcd, ac \to abcd\}$, which corresponds to a Duquenne-Guigues basis. ☐

The following theorem ensures the minimality (w.r.t. the cardinality) of the Duquenne-Guigues basis.

Theorem 1. *Any Duquenne-Guigues basis is a minimum basis.*

In the following, we describe the algorihtm 2 to calculate a Left-Minimal Direct Basis based on Algorithm 1. The algorithm is polynomial and it is described searching a good understanding. Of course, an implementation using lectic order would improve considerably its efficiency.

First, we consider the following equivalence rules.

Definition 3 (Aggregation rules). *Let A, B, C and D be sets of attributes.*

1. *If* $A \subseteq C$ *then* $\{A \to B, C \to D\} \equiv \{A \to B, BC \to D \smallsetminus B\}$.
2. *If* $A \subseteq C \subseteq A \cup B$ *then* $\{A \to B, C \to D\} \equiv \{A \to BD\}$.

Algorithm 2: Algorithm for computing a Left-Minimal Direct Basis

```
input  : An LSI $\Phi$
output: A Minimal Direct Basis
$T := \emptyset$;
foreach $\langle B, mg(B) \rangle \in \Phi$ do
    foreach $A \in mg(B)$ do  $T := T \cup \{(A, A \to B)\}$
repeat
    $S := T$;
    foreach $(M, A \to B), (N, C \to D) \in T$ such that $A \subsetneq C$ and $B \nsubseteq C$ do
        $T := T \smallsetminus \{(N, C \to D)\}$;
        if $B \cup C \neq D$ then $T := T \cup \{(N, BC \to D)\}$
until $T = S$ ;
$S := \emptyset$;
foreach $(M, A \to B) \in T$ do $S := S \cup \{M \to B \smallsetminus M\}$;
return $S$
```

We let for an extended version of this paper the proof that these equivalences rules are derived rules of equivalences presented in Theorem **??**.

We propose in the Algorithm 2 the use of the two aggregation rules for reducing the number of implications and also the consequent of implications.

Theorem 2. *Let $T = \{(M_1, A_1 \to B_1), \ldots\}$ be a set of pairs with minimal generators and implications obtained from minimal generators and closed sets. The exhaustive application of the two Aggregation rules produces a left-minimal direct basis.*

Example 2. Let $T = \{b \to agh, d \to a, bn \to h, ab \to defg, abc \to djk\}$ be a set of implications, Algorithm MinGen_0 ([6]) returns a list with closed sets and their minimal generators, $\Phi = \{< abdefgh, \{b\} >, < abcdefghkj, \{bc\} >, < abdefghn, \{bn\} >, < abcdefghkn, \{bcn\} >, < ad, \{d\} >\}$

In the first step of the Algorithm 2 with Φ we build $T = \{$(b, b→abdefgh), (bc,bc →abcdefghkj), $(bn, bn \to abdefghn), (bcn, bcn \to abcdefghkn), (d, d \to ad)\}$.

Then, we apply the Aggregation rules foreach couple of elements in T. At the end of these comparisons, we have $T = \{(b, b \to abdefgh), (bc, abcdefgh \to abcdefghkj), (d, d \to ad)\}$. And from this, in the last foreach of the Algorithm 2, it renders the following left-minimal direct basis $S = \{b \to adefgh, bc \to adefghkj, d \to a\}$.

By the other side, Algorithm 1 return the following Duquenne-Guigues basis $\{b \to adefgh, abcdefgh \to kj, d \to a\}$ which in a minimal basis, but not a left-minimal one.

4 Conclusion

In this paper we present an algorithm which allows the transformation of a set of all closed sets and their corresponding minimal generators into a left-minimal direct basis. The study about the soundness, completeness, and complexity of the algorithms proposed are left to a extended paper.

The new method uses some equivalences deduced in the $\mathbf{SL}_{\text{FD}}$ Logic and follows the Lectic order to traverse the list of minimal generators and implications associated and return a set of implications but with good properties.

As future work we propose to extend the Duquenne-Guigues basis definition to consider a generalized fuzzy extension of implications. We propose the definition introduced in [2] that has been shown to be the most general one. In [4] a non trivial extension of the $\mathbf{SL}_{\text{FD}}$ Logic for the generalized definition of fuzzy functional dependency was introduced. The generalized version of the $\mathbf{SL}_{\text{FD}}$ Logic will be used to develop a method to get basis for the generalized fuzzy implications.

Acknowledgment

Supported by Grants TIN2011-28084 of the Science and Innovation Ministry of Spain, and No. P09-FQM-5233 of the Junta de Andalucía.

References

1. William W. Armstrong, *Dependency structures of data base relationships*, Proc. IFIP Congress. North Holland, Amsterdam: 580–583, 1974.
2. Radim Belohlávek, and Vilém Vychodil, *Functional Dependencies of Data Tables Over Domains with Similarity Relations*, Proc. of the 2nd Indian International Conference on Artificial Intelligence, 2005.
3. Radim Belohlávek, and Vilém Vychodil, *Formal Concept Analysis with Constraints by Closure Operators*, Lecture Notes in Computer Science, 4068:131–143, 2006.
4. Radim Belohlávek, Pablo Cordero, Manuel Enciso, Ángel Mora, and Vilém Vychodil, *An Efficient Reasoning Method for Dependencies over Similarity and Ordinal Data*, Lecture Notes in Computer Science, 7647: 408–419, 2012.
5. K. Bertet, B. Monjardet, *The multiple facets of the canonical direct unit implicational basis*, Theor. Comput. Sci., 411(22-24): 2155–2166, 2010.
6. Pablo Cordero, Manuel Enciso, Ángel Mora, Manuel Ojeda-Aciego, *Computing Minimal Generators from Implications: a Logic-guided Approach*, CLA 2012: 187–198, 2012.
7. A. Day, *The lattice theory of functional dependencies and normal decompositions*, International Journal of Algebra and Computation, 2: pp. 209–431, 1990.
8. B. Ganter, *Two basic algorithms in concept analysis*, Technische Hochschule, Darmstadt, 1984.
9. 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.
10. 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.
11. L. Szathmary and P. Valtchev and A. Napoli and R. Godin, Efficient Vertical Mining of Frequent Closures and Generators, LNCS 5772: 393–404, 2009.
12. C. Frambourg, P. Valtchev, and R. Godin, Merge-based computation of minimal generators. Discrete Mathematics, LNCS 3596: 181–194, 2005.
13. 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.

Comparing Performance of Formal Concept Analysis and Closed Frequent Itemset Mining Algorithms on Real Data

Lenka Pisková, Tomáš Horváth

University of Pavol Jozef Šafárik, Košice, Slovakia
`lenka.piskova@student.upjs.sk, tomas.horvath@upjs.sk`

Abstract. In this paper, an experimental comparison of publicly available algorithms for computing intents of all formal concepts and mining frequent closed itemsets is provided. Experiments are performed on real data sets from UCI Machine Learning Repository and FIMI Repository. Results of experiments are discussed at the end of the paper.

1 Introduction

Formal Concept Analysis (FCA) [9] is a method for analysing data in the form of a table with applications in many disciplines.A formal concept is a formalization of the concept of a "concept" which consists of two parts, a set of objects which forms its extension and a set of attributes which forms its intension [25]. Formal concepts can be ordered according to the subconcept-superconcept relation resulting in a concept lattice.

Frequent itemset mining (FIM) introduced in [1] was proposed as a method for market basket analysis. The identification of sets of items (itemsets) which often occur together in a database (the frequency is not less than a user defined minimum support threshold) is one of the basic tasks in Data Mining. When the minimum support is set low, a huge number of itemsets is generated. To overcome this problem, closed and maximal frequent itemsets were proposed.

FCA and FIM are two research fields that are closely related to each other [20]. Naturally, they address similar problems, e.g. selecting important concepts versus finding interesting patterns in data. Moreover, they inspire each other (Iceberg concept lattice which is the set of all frequent concepts connected with the subconcept-superconcept relation [23]).

Finding the set of all intents (of formal concepts) is equivalent to finding the set of all closed frequent itemsets using a minimum support equal to zero [20]. Nonetheless, there is no experimental comparison between algorithms for computing formal concepts and algorithms for mining frequent closed itemsets. The aim of this paper is to provide such comparison on real-world data.

2 Compared Algorithms

The problem of generating formal concepts and/or a concept lattice has been well studied and many algorithms have been proposed [8], [15], [17], [21], [22]. A comparative performance study of algorithms for building concept lattices can be found in [10] and [16]. In this paper we will focus only on those algorithms which compute the set of all formal concepts (frequent closed itemsets) only. Therefore, we do not compare our results with [10] and [16].

The fastest algorithms for computing formal concepts are FCbO [14] and In-Close [2] which are based on the CbO algorithm [15]. In the competition between FCA algorithms at ICCS 2009[1] FCbO took the first place and the runner-up was In-Close. The improvement of In-Close algorithm [3] was developed in response to the competition to outperform FCbO, but our results show that FCbO still performs better. A parallel variant of FCbO was also proposed [14], however, we consider only the serial version in this paper.

Implementations of algorithms for mining closed frequent itemsets were experimentally compared[2] and presented at FIMI'03 and FIMI'04 workshops [12]. The best of the tested algorithms ([5], [13], [18], [19], [24], [26]) was FP-Close [13] although it gave a segmentation fault for 4 out of 14 data sets.

In this paper, we provide an experimental comparison of 10 algorithms on real-world data sets whose implementations are publicly available, two of them compute formal concepts (FCbO and In-Close2) and the remaining 8 generate closed frequent itemsets ([5], [6], [7], [13], [18], [19], [24]).

3 Experimental Evaluation

We have carried out a number of experiments for several real-world data sets to compare FCA and FIM algorithms that are publicly available. The characteristics of selected data sets [4], [11] are shown in the table 1.

Table 1. The characteristics of data sets.

Dataset	# Transactions	# Items	Density (%)	Small/Big
Accidents	340183	468	33.8	Big
Car Evaluation	1728	25	28	Small
Connect	67557	129	43	Big
Kosarak	990002	41270	8.1	Big
Mushroom	8124	119	23	Small
Retail	88162	16469	10.3	Big
Tic-tac-toe	958	29	34	Small

The experiments were conducted on the computing node with 16 cores equipped with 24 GB RAM memory running GNU/Linux openSUSE 12.1.

The measured times are CPU times. Each algorithm was run three times for each data set and the given minimum support threshold value to get the most accurate results. All reported times are the average times of the three

---

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

[2] http://fimi.ua.ac.be/experiments/

runs. The output was turned off, i.e. the results of algorithms (intents of formal concepts/frequent closed itemsets) were neither written to a file nor to the screen.

Some of the algorithms were originally developed to mine closed frequent itemsets. For Apriori, Carpenter and Eclat we have set the flag -tc to mine closed frequent itemsets. Similarly, we have used the flag -fci for Mafia.

The input file of In-Close2 is in the cxt (formal context) format while the input of other algorithms is in the standard FIMI format - each line represents a list of attributes of an object/a list of items in a transaction. The disadvantage of In-Close2 is that unlike other algorithms it also computes extents of formal concepts (in addition to their intents).

Some algorithms had problems on certain data sets. For mushroom, Apriori gets killed, Carpenter outputs an incorrect number of closed frequent itemsets (238827), DCI_Closed does not calculate the result in a reasonable time (we have stopped the computation after a few hours). In-Close2 gives an incorrect number of formal concepts (59343) for tic-tac-toe.

For kosarak, FPClose is either aborted due to the invalid pointer or gives segmentation fault for the support 0.8% and supports less than or equal to 0.6%. For retail with *minsup* = 0, Apriori gets killed, Carpenter gives an incorrect number of closed frequent itemsets (2186693) and DCI_Closed is aborted. In-Close2 gives segmentation fault for the supports lower or equal to 60% on accidents and for all supports except for 90% on connect.

Table 2. Performance of algorithms for mining closed frequent itemsets with the minimum support equal to 0 (CPU time in seconds).

# Formal concepts	Car Evaluation 12640	Mushroom 238710	Tic-tac-toe 59505
Afopt	0.13	5.083	1.006
Apriori	0.04	-	0.25
Carpenter	0.71	-	1.646
DCI_Closed	0.023	-	0.05
Eclat	0.02	0.976	0.143
FCbO	0.02	0.803	0.13
FPClose	0.056	1.586	0.36
In-Close2	0.043	2.583	-
LCM	0.02	1.363	0.086
Mafia	0.243	39.746	2.64

We have compared the performance of the algorithms for mining intents of all formal concepts, i.e. closed frequent itemsets using *minsup* = 0 (typical task in FCA) on small data sets. The results are depicted in the table 2. Arguably, FCbO is the best algorithm for the given task, it is the fastest algorithm for car and mushroom and the third fastest for tic-tac-toe. LCM and Eclat perform well on these data sets, too. Considering also the results on retail with *minsup* = 0, LCM is the fastest algorithm and the runner-up are Eclat and FPClose.

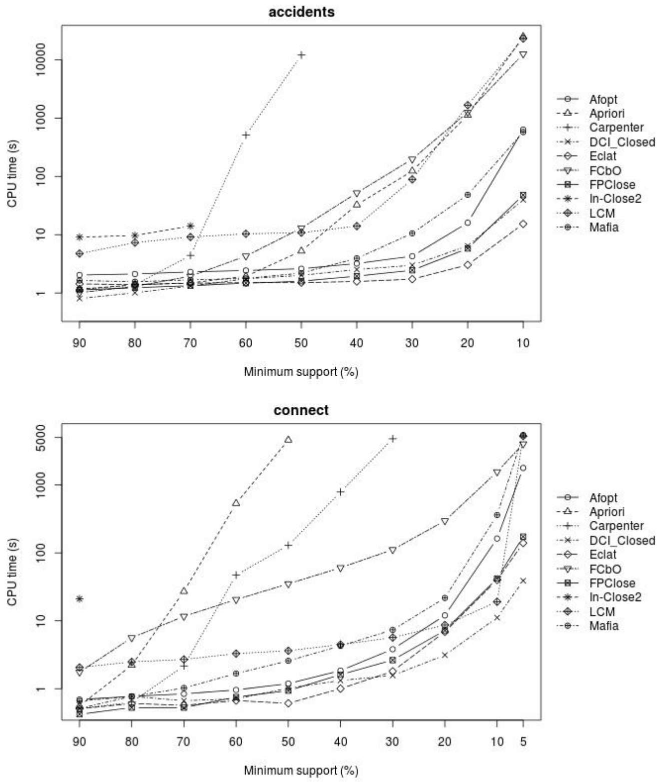


Fig. 1. Performance of algorithms on *accidents* and *connect* data sets for various minimum support values (CPU time in seconds are measured).

Other tests were performed on big data sets and the performance of algorithms was tested for various values of support. Figures 1 and 2 show the timings for the algorithms on the accidents, connect, kosarak and retail data sets. The performance of the FCbO algorithm is average on big data sets except for kosarak. Eclat, DCI_Closed and FPClose are good choice in the case of dense data sets (accidents, connect). However, the runtime of most algorithms increases dramatically with decreasing minimum support on these data sets. Afopt, DCI_Closed and LCM are suitable for sparse data sets (kosarak, retail) although the Afopt algorithm is not able to handle the retail data set for $minsup = 0$. The timings for In-Close2 on kosarak are not included, because the computation took several hours just for high values of support .

For kosarak, in our experimental testing FPClose failed while FPClose was the fastest algorithm for low as well as high values of support in [12]. Our results on other data sets correspond to some extent to the results in [12].

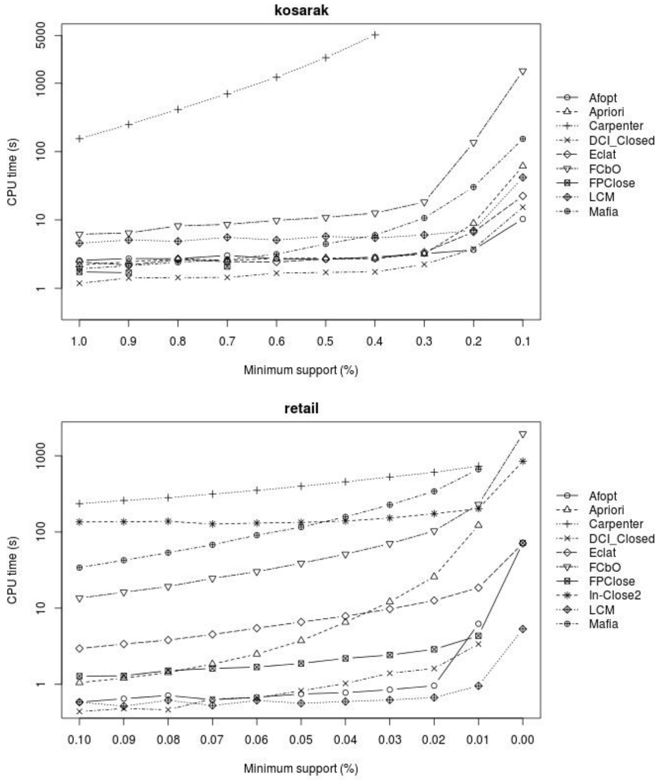


Fig. 2. Performance of algorithms on *kosarak* and *retail* data sets for various minimum support values (CPU time in seconds are measured).

4 Conclusion

We have experimentally compared algorithms for computing intents of formal concepts and algorithms for mining closed frequent itemsets on real-world data. Our experimental testing has no clear winner for different data sets and minimum support threshold setting. In our opinion, DCI_Closed behaves well although it had some problems on the mushroom data set and the retail data set with $minsup = 0$. On small data sets, the fastest algorithm is FCbO.

Acknowledgements: This work was partially supported by the research grants VEGA 1/0832/12 and the Center of knowledge and information systems in Košice (ITMS 26220220158).

References

1. R. Agrawal, T. Imielinski, A. Swami: Mining association rules between sets of items in large databases. SIGMOD, 1993.

2. S. Andrews: In-Close, a fast algorithm for computing formal concepts. ICCS 2009.
3. S. Andrews: In-Close2, a High Performance Formal Concept Miner. ICCS 2011.
4. K. Bache, M. Lichman: UCI Machine Learning Repository, 2013, http://archive.ics.uci.edu/ml, University of California, Irvine, School of Information and Computer Sciences.
5. Ch. Borgelt: Efficient Implementations of Apriori and Eclat. IEEE ICDM Workshop on FIMI (2003).
6. Ch. Borgelt, X. Yang, R. Nogales-Cadenas, P. Carmona-Saez, A. Pascual-Montano: Finding Closed Frequent Item Sets by Intersecting Transactions. EDBT 2011.
7. D. Burdick, M. Calimlim, J. Flannick, J. Gehrke, T. Yiu: MAFIA: A Performance Study of Mining Maximal Frequent Itemsets. IEEE ICDM Workshop on FIMI (2003).
8. B. Ganter: Two basic algorithms in concept analysis. (Technical Report FB4-Preprint No. 831). TH Darmstadt, 1984.
9. B. Ganter, R. Wille: Formal concept analysis: Mathematical foundations. Springer, 1999.
10. R. Godin, R. Missaoui, H. Alaoui: Incremental concept formation algorithms based on Galois (concept) lattice. Computational Intelligence, vol. 1(2), 1995.
11. FIMI Repository, http://fimi.ua.ac.be/data/.
12. B. Goethals, M. J. Zaki: Advances in Frequent Itemset Mining Implementations: Report on FIMI03. SIGKDD Explorations, vol. 6 (1), 2004.
13. G. Grahne, J. Zhu: Efficiently Using Prefix-trees in Mining Frequent Itemsets. IEEE ICDM Workshop on FIMI (2003).
14. P. Krajca, J. Outrata, V. Vychodil: Advances in algorithms based on CbO. CLA 2010.
15. S. O. Kuznetsov: A Fast Algorithm for Computing All Intersections of Objects in a Finite Semi-lattice. Automatic Documentation and Mathematical Linguistics, vol. 27 (5), 1993.
16. S. O. Kuznetsov, S. A. Obiedkov: Comparing Performance of Algorithms for Generating Concept Lattices. Journal of Experimental and Theoretical Artificial Intelligence, vol. 14 (2-3), 2002.
17. Ch. Lindig: Fast Concept Analysis. Working with Conceptual Structures – Contributions to ICCS 2000, 2000.
18. G. Liu, H. Lu, J. X. Yu, W. Wei, X. Xiao: AFOPT: An Efficient Implementation of Pattern Growth Approach. IEEE ICDM Workshop on FIMI (2003).
19. C. Lucchese, S. Orlando, R. Perego: DCI_Closed: A Fast and Memory Efficient Algorithm to Mine Frequent Closed Itemsets. IEEE ICDM Workshop on FIMI (2004).
20. B. Martin, P. Euklund: From Concepts to Concept Lattice: A Border Algorithm for Making Covers Explicit. ICFCA, 2008.
21. E. M. Norris: An Algorithm for Computing the Maximal Rectangles in a Binary Relation. Revue Roumaine de Mathématiques Pures et Appliquées, vol. 23(2), 1978.
22. L. Nourine, O. Raynaud: A fast algorithm for building lattices. Information Processing Letters, vol. 71, 1999.
23. G. Stumme, R. Taouil, Y. Bastide, N. Pasquier, L. Lakhal: Computing Iceberg Concept Lattices with Titanic. Journal on Knowledge and Data Engineering, vol. 42(2), 2002.
24. T. Uno, T. Asai, Y. Uchida, H. Arimura: LCM: An Efficient Algorithm for Enumerating Frequent Closed Item Sets. IEEE ICDM Workshop on FIMI (2003).
25. R. Wille: Restructuring lattice theory: An approach based on hierarchies of concepts. Ordered Sets, vol. 83, 1982.
26. M. J. Zaki: Scalable algorithms for association mining. IEEE Transactions on Knowledge and Data Engeneering, vol. 12, 2000.

Author Index

Editors: Manuel Ojeda-Aciego, Jan Outrata

Title: CLA 2013, Proceedings of the Tenth International
Conference on Concept Lattices and Their Applications

Publisher & Print: Laboratory L3i &
Computer Sciences, Image Processing and Intercation Lab
University of La Rochelle
Avenur Michel Crépeau
17042 La Rochelle Cedex 1
France

Place, year, edition: La Rochelle, 2013, 1st

Page count: xii+306

Impression: 50

Archived at: `cla.inf.upol.cz`

Not for sale

ISBN 978–2–7466–6566–8