CLA 2011

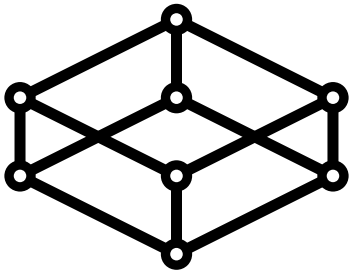
Proceedings of the Eighth International Conference on
Concept Lattices and Their Applications

CLA Conference Series

http://cla.inf.upol.cz

INRIA Nancy – Grand Est and LORIA, France

The Eighth International Conference on
Concept Lattices and Their Applications



CLA 2011

Nancy, France
October 17–20, 2011

Edited by

Amedeo Napoli
Vilem Vychodil

CLA 2011, October 17–20, 2011, Nancy, France.

Technical Editors:
Jan Outrata, jan.outrata@upol.cz
Vilem Vychodil, vychodil@acm.org

Page count: xii+419
Impression: 100
Edition: 1st
First published: 2011

Printed version published by INRIA Nancy – Grand Est and LORIA, France
ISBN 978–2–905267–78–8

Organization

CLA 2011 was organized by the INRIA Nancy – Grand Est and LORIA

Steering Committee

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

Program Chairs

Amedeo Napoli	INRIA NGE/LORIA, Nancy, France
Vilem Vychodil	Palacky University, Olomouc, Czech Republic

Program Committee

Jaume Baixeries	Polytechnical University of Catalonia
Jose Balcazar	University of Cantabria and UPC Barcelona, Spain
Radim Belohlavek	Palacky University, Olomouc, Czech Republic
Karell Bertet	University of La Rochelle, France
François Brucker	University of Marseille, France
Claudio Carpineto	Fondazione Ugo Bordoni, Roma, Italy
Jean Diatta	Université de la Réunion, France
Felix Distel	TU Dresden, Germany
Florent Domenach	University of Nicosia, Cyprus
Mireille Ducassé	IRISA Rennes, France
Alain Gély	University of Metz, France
Cynthia Vera Glodeanu	TU Dresden, Germany
Marianne Huchard	LIRMM, Montpellier, France
Vassilis G. Kaburlasos	TEI, Kavala, Greece
Stanislav Krajci	University of P.J. Safarik, Kosice, Slovakia
Sergei O. Kuznetsov	State University HSE, Moscow, Russia
Léonard Kwuida	Zurich University of Applied Sciences, Switzerland
Mondher Maddouri	URPAH, University of Gafsa, Tunisie
Rokia Missaoui	UQO, Gatineau, Canada
Lhouari Nourine	LIMOS, University of Clermont Ferrand, France

Sergei Obiedkov	State University HSE, Moscow, Russia
Manuel Ojeda-Aciego	University of Malaga, Spain
Jan Outrata	Palacky University, Olomouc, Czech Republic
Pascal Poncelet	LIRMM, Montpellier, France
Uta Priss	Napier University, Edinburgh, United Kingdom
Olivier Raynaud	LIMOS, University of Clermont Ferrand, France
Camille Roth	EHESS, Paris, France
Stefan Schmidt	TU Dresden, Germany
Baris Sertkaya	SAP Research Center, Dresden, Germany
Henry Soldano	Université of Paris 13, France
Gerd Stumme	University of Kassel, Germany
Petko Valtchev	Université du Québec à Montréal, Canada

Additional Reviewers

Mikhail Babin	State University HSE, Moscow, Russia
Daniel Borchmann	TU Dresden, Germany
Peggy Cellier	IRISA Rennes, France
Sebastien Ferre	IRISA Rennes, France
Nathalie Girard	University of La Rochelle, France
Alice Hermann	IRISA Rennes, France
Mehdi Kaytoue	INRIA NGE/LORIA, Nancy, France
Petr Krajca	Palacky University, Olomouc, Czech Republic
Christian Meschke	TU Dresden, Germany
Petr Osicka	Palacky University, Olomouc, Czech Republic
Violaine Prince	LIRMM, Montpellier, France
Chedy Raissy	INRIA NGE/LORIA, Nancy, France
Yoan Renaud	LIRIS, Lyon, France
Heiko Reppe	TU Dresden, Germany
Lucie Urbanova	Palacky University, Olomouc, Czech Republic
Jean Villerd	ENSAIA, Nancy, France

Organization Committee

Mehdi Kaytoue (chair)	INRIA NGE/LORIA, Nancy, France
Elias Egho	INRIA NGE/LORIA, Nancy, France
Felipe Melo	INRIA NGE/LORIA, Nancy, France
Amedeo Napoli	INRIA NGE/LORIA, Nancy, France
Chedy Raïssi	INRIA NGE/LORIA, Nancy, France
Jean Villerd	ENSAIA, Nancy, France

Table of Contents

Short Papers

Preface

The Eighth International Conference "Concept Lattices and Applications (CLA 2011)" is held in Nancy, France from October 17th until October 20th 2011. CLA 2011 is aimed at providing to everyone interested in Formal Concept Analysis and more generally in Concept Lattices or Galois Lattices, students, professors, researchers and engineers, a global and an advanced view of some of the last research trends and applications in this field. As the diversity of the selected papers shows, there is a wide range of theoretical and practical research directions, around data and knowledge processing, e.g. data mining, knowledge discovery, knowledge representation, reasoning, pattern recognition, together with logic, algebra and lattice theory.

This volume includes the selected papers and the abstracts of the 7 invited talks. This year there were initially 47 submissions from which 27 papers were accepted as full papers and 3 papers as posters. We would like to thank here the authors for their work, often of very good quality, the members of the program committee and the external reviewers who did a great job as this can be seen in their reviews. This is one witnesses of the growing quality and importance of CLA, highlightening its leading position in the field.

Next, this year is a little bit special while the bicentennial of the birth of Evariste Galois (1811–1832) is celebrated, particularly in France. Evariste Galois has something to do with Concept Lattices as they are based on a so-called "Galois connection". Among the invited speakers, some of them will discuss of these fundamental aspects of Concept Lattices. Moreover, this is also the occasion of thanking the seven invited speakers who, at least we hope that, will meet the wishes of the attendees.

We would like to thank firstly our first sponsors, namely the CNRS GDR I3 and Institut National Polytechnique de Lorraine (INPL). Then we would like to thank the steering committee of CLA for giving us the occasion of leading this edition of CLA, the conference participants for their participation and support, and people in charge of the organization, especially Anne-Lise Charbonnier, Nicolas Alcaraz and Mehdi Kaytoue, whose help was very precious in many occasions.

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

October 2011

Amedeo Napoli
Vilem Vychodil
Program Chairs of CLA 2011

Mathematical Morphology, Lattices, and Formal Concept Analysis

Isabelle Bloch

Telecom ParisTech, CNRS LTCI, Paris, France

Abstract. Lattice theory has become a popular mathematical framework in different domains of information processing, and various communities employ its features and properties, e.g. in knowledge representation, in logics, automated reasoning and decision making, in image processing, in information retrieval, in soft computing, in formal concept analysis. Mathematical morphology is based adjunctions, on the algebraic framework of posets, and more specifically of complete lattices, which endows it with strong properties and allows for multiple applications and extensions. In this talk we will summarize the main concepts of mathematical morphology and show their instantiations in different settings, where a complete lattice can be built, such as sets, functions, partitions, fuzzy sets, bipolar fuzzy sets, formal logics . . . We will detail in particular the links between morphological operations and formal concept analysis, thus initiating links between two domains that were quite disconnected until now, which could therefore open new interesting perspectives.

Random concept lattices

Richard Emilion

MAPMO, University of Orléans, France

Abstract. After presenting an algorithm providing concepts and frequent concepts, we will study the random size of concept lattices in the case of a Bernoulli(p) context. Next, for random lines which are independent and identically distributed or more generally outcomes of a Markov chain, we will show the almost everywhere convergence of the random closed intents towards deterministic intents. Finally we will consider the problem of predicting the number of concepts before choosing any algorithm.

Galois and his Connections—A retrospective on the 200th birthday of Evariste Galois

Marcel Erné

University of Hannover, Germany

Abstract. A frequently used tool in mathematics is what Oystein Ore called "Galois connections" (also "Galois connexions", "Galois correspondences" or "dual adjunctions"). These are pairs (φ, ψ) of maps between ordered sets in opposite direction so that $x \leq \psi(y)$ is equivalent to $y \leq \varphi(x)$. This concept looks rather simple but proves very effective. The primary gain of such "dually adjoint situations" is that the ranges of the involved maps are dually isomorphic: thus, Galois connections present two faces of the same medal.

Many concrete instances are given by what Garrett Birkhoff termed "polarities": these are nothing but Galois connections between power sets. In slightly different terminology, the fundamental observation of modern Formal Concept Analysis is that every "formal context", that is, any triple (J, M, I) where I is a relation between (the elements of) J and M, gives rise to a Galois connection (assigning to each subset of one side its "polar", "extent" or "intent" on the other side), such that the resulting two closure systems of polars are dually isomorphic; more surprising is the fact that, conversely, every dual isomorphism between two closure systems arises in a unique fashion from a relation between the underlying sets. In other words: the complete Boolean algebra of all relations between J and M is isomorphic to that of all Galois connections between $\P J$ and $\P M$, and also to that of all dual isomorphisms between closure systems on J and M, respectively.

The classical example is the Fundamental Theorem of Galois Theory, establishing a dual isomorphism between the complete lattice of all intermediate fields of a Galois extension and that of the corresponding automorphism groups, due to Richard Dedekind and Emil Artin. In contrast to that correspondence, which does not occur explicitly in Galois' succinct original articles, a few other closely related Galois connections may be discovered in his work (of course not under that name). Besides these historical forerunners, we discuss a few other highlights of mathematical theories where Galois connections enter in a convincing way through certain "orthogonality" relations, and show how the Galois approach considerably facilitates the proofs. For example, each of the following important structural isomorphisms arises from a rather simple relation on the respective ground sets:

- the dual isomorphism between the subspace lattice of a finite-dimensional linear space and the left ideal lattice of its endomorphism ring
- the duality between algebraic varieties and radical ideals
- the categorical equivalence between ordered sets and Alexandroff spaces
- the representation of complete Boolean algebras as systems of polars.

Canonical extensions, Duality theory, and Formal Concept Analysis

Mai Gehrke

LIAFA CNRS – University of Paris 7, France

Abstract. The theory of canonical extensions, developed by Jonsson and Tarski in the setting of Boolean algebras with operators, provides an algebraic approach to duality theory. Recent developments in this theory have revealed that in this algebraic guise duality theory is no more complicated outside than within the setting of Boolean algebras or distributive lattices. This has opened the possibility of exporting the highly developed machinery and knowledge available in the classical setting (e.g. in modal logic) to the completely general setting of partially ordered and non-distributive lattice ordered algebras. Duality theory in this setting is a special instance of the connection between formal contexts and concept lattices and thus allows methods of classical algebraic logic to be imported into FCA.

This will be an introductory talk on the subject of canonical extensions with the purpose of outlining the relationship between the three topics of the title.

Galois connections and residuation: origins and developments II

Bruno Leclerc

CAMS – École des Hautes Études en Sciences Sociales, Paris, France

Abstract. From the seventies, the uses of Galois connections (and residuated/residual maps) multiplied in applied fields. Indeed Galois connections have been several times rediscovered for one or another purpose, for instance in fuzzy set theory or aggregation problems. In this talk, we illustrate the diversity of such applications. Of course, the many developments in Galois lattices and Formal Concept Analysis, with their relation with Data Mining, will be only briefly evoked. Besides these developments, one finds, among other uses, alternative methods to study a correspondence (binary relation) between two sets, models of classification and preferences, fitting and aggregation problems.

Galois connections and residuation: origins and developments I

Bernard Monjardet

Centre d'Economie de la Sorbonne (University of Paris 1) and CAMS (Centre Analyse et Mathmatique Sociale), France

Abstract. The equivalent notions of Galois connexions, and of residual and residuated maps occur in a great varieties of "pure" as well as "applied" mathematical theories. They explicitly appeared in the framework of lattice theory and the first of these talks is devoted to the history of their appearance and of the revealing of their links in this framework. So this talk covers more or less the period between 1940 (with the notion of polarity defined in the first edition of Birkoff's book Lattice theory) and 1972 (with Blyth and Janowitz's book Residuation theory), a period containing fundamental works like Öre's 1944 paper Galois connexions or Croisot's 1956 paper Applications résiduées.

Metrics, Betweeness Relations, and Entropies on Lattices and Applications

Dan Simovici

Department of Computer Science, University of Massachusetts at Boston, USA

Abstract. We discuss an algebraic axiomatization of the notion of entropy in the framework of lattices as well as characterizations of metric structures induced by such entropies. The proposed new framework takes advantage of the partial orders defined on lattices, in particular the semimodular lattice of partitions of a finite set to allow multiple applications in data mining: data discretization, recommendation systems, classification, and feature selection.

Vertical decomposition of a lattice using clique separators

Anne Berry, Romain Pogorelcnik, Alain Sigayret

LIMOS UMR CNRS 6158**
Ensemble Scientifique des Cézeaux Université Blaise Pascal, F-63 173 Aubière, France.
berry@isima.fr, romain.pogorelcnik@isima.fr, sigayret@isima.fr

Abstract. A concept (or Galois) lattice is built on a binary relation; this relation can be represented by a bipartite graph. We explain how we can use the graph tool of clique minimal separator decomposition to decompose some bipartite graphs into subgraphs in linear time; each subgraph corresponds to a subrelation. We show that the lattices of these subrelations easily yield the lattice of the global relation. We also illustrate how this decomposition is a tool to help displaying the lattice.

Keywords: lattice decomposition, clique separator decomposition, lattice drawing

1 Introduction

In many algorithms dealing with hard problems, a divide-and-conquer approach is helpful in practical applications. Computing the set of concepts associated with a given context (or the set of maximal rectangles associated with a binary relation) is time-consuming, as there may be an exponential number of concepts. It would be interesting to decompose the lattice into smaller sublattices. What we propose here is to decompose the relation into smaller subrelations, compute the lattice of each subrelation, and then use these lattices to reconstruct the lattice of the global relation.

For this, we use a graph decomposition, called "clique separator decomposition", introduced by Tarjan [9], and refined afterwards (see [3] for an extensive introduction to this decomposition). The general principal is roughly the following: repeatedly find a set of vertices which are pairwise adjacent (called a clique) and whose removal disconnects the graph (called a separator), then copy this clique separator into the different connected components obtained. When the decomposition is completed, a set of subgraphs is obtained, inconveniently called 'atoms': each subgraph is a maximal subgraph containing no clique separator.

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

In a previous work [2], we used graph algorithms on the complement of the bipartite graph associated with the relation. In this paper, we will apply this decomposition directly to the bipartite graph itself.

It turns out upon investigation that the subgraphs obtained divide not only the graph, but in a very similar fashion divide the matrix of the relation, the set of concepts and the lattice. When the relation has a clique separator of size two, the lattice, as we will explain further on, is divided along a vertical axis by an atom and a co-atom which correspond to the two vertices of the separator. Thus not only can the concepts be computed on the subrelations, but the Hasse diagram of the lattice can be drawn better, as no edge need cross this vertical line.

Moreover, in a bipartite graph, this decomposition can be implemented with a better worse-case time complexity than in the general case, as the clique separators can be of size one (in this case they are called articulation points) or of size two. In both cases, the entire decomposition can be computed in linear time, *i.e.* in the size of the relation, thanks to the works of [6] and [3]. Although some graphs do not have a clique separator, when there is one, the decomposition is thus a useful and non-expensive pre-processing step.

The paper is organized as follows: we will first give some more preliminaries in Section 2. In Section 3, we explain how a bipartite graph is decomposed. In Section 4, we show how to reconstruct the global lattice from the concepts obtained on the subrelations. In Section 5, we discuss using vertical decomposition as a tool for layout help. Finally, in Section 6, we conclude with some general remarks.

2 Preliminaries

We will first recall essential definitions and properties.

All the graphs will be undirected and finite. For a graph $G = (V, E)$, V is the vertex set and E is the edge set. For $xy \in E, x \neq y$, x and y are said to be *adjacent*; we say that x *sees* y. A graph is *connected* if, for every pair $\{x, y\}$ of vertices, there is a path from x to y. When a graph is not connected, the maximal connected subgraphs are called the *connected components*. For $C \subset V$, $G(C)$ denotes the subgraph induced by C. In a graph $G = (V, E)$, the *neighborhood* of a vertex $x \in V$ is the set $N_G(x) = \{y \in V, x \neq y | xy \in E\}$. $N_G(x)$ is denoted $N(x)$ when there is no ambiguity.

A *clique* is set of vertices which induces a complete graph, *i.e.* with all possible edges.

A *bipartite graph* $G = (X + Y, E)$, where $+$ stands for disjoint union, is built on two vertex sets, X and Y, with no edge between vertices of X and no edge between vertices of Y. A *maximal biclique* of a bipartite graph $G = (X + Y, E)$ is a subgraph $G(X' + Y')$ with all possible edges between the vertices of X' and the vertices of Y'.

A *relation* $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$ on a set $\mathcal{O}$ of objects and a set $\mathcal{A}$ of attributes is associated with a *bipartite graph* $G{=}(\mathcal{O} + \mathcal{A}, E)$, which we will denote $Bip(\mathcal{R})$;

thus, for $x \in \mathcal{O}$ and $y \in \mathcal{A}$, (x, y) is in $\mathcal{R}$ iff xy is an edge of G. The maximal rectangles of the relation correspond exactly to the maximal bicliques (maximal complete bipartite subgraphs) of $Bip(\mathcal{R})$ and to the elements (the concepts) of *concept lattice* $\mathcal{L}(\mathcal{R})$ associated with context $(\mathcal{O}, \mathcal{A}, \mathcal{R})$. If $O_1 \times A_1$ and $O_2 \times A_2$ are concepts of $\mathcal{L}(\mathcal{R})$ then $O_1 \times A_1 \preceq O_2 \times A_2$ iff $O_1 \subseteq O_2$ iff $A_1 \supseteq A_2$; the corresponding bicliques on vertex sets $O_1 + A_1$ and $O_2 + A_2$ of $Bip(\mathcal{R})$ are comparable the same way.

An *atom* (resp. *co-atom*) of $\mathcal{L}(\mathcal{R})$ is a concept covering the minimum element (resp. covered by the maximum element). In the bipartite graph $Bip(\mathcal{R})$, the neighborhoods are defined as follows: for $x \in \mathcal{O}$, $N(x) = \mathcal{R}(x) = \{y \in \mathcal{A} | (x, y) \in \mathcal{R}\}$, and for $x \in \mathcal{A}$, $N(x) = \mathcal{R}^{-1}(x) = \{y \in \mathcal{O} | (y, x) \in \mathcal{R}\}$.

A *separator* in a connected graph is a set of vertices, the removal of which disconnects the graph. A *clique separator* is a separator which is a clique. Clique separator decomposition [9] of a graph $G = (V, E)$ is a process which repeatedly finds a clique separator S and copies it into the connected components of $G(V - S)$. When only minimal separators are used (see [3] for extensive general definitions), the decomposition is unique and the subgraphs obtained in the end are exactly the maximal subgraphs containing no clique separator [8], [3]. In a bipartite graph, the clique minimal separators are of size one or two. A separator of size one is called a *articulation point*. A clique separator $S = \{x, y\}$ of size two is minimal if there are two components C_1 and C_2 of $G(V - S)$ such that x and y both have at least one neighbor in C_1 as well as in C_2.

3 Decomposing the bipartite graph and the relation

In the rest of the paper, we will use the bipartite graph $Bip(\mathcal{R})$ defined by a relation $\mathcal{R} \subseteq \mathcal{O} \times \mathcal{A}$. Figure 1 shows an example of a relation with the corresponding bipartite graph.

	a	b	c	d	e	f	g	h	i
1	×		×	×	×				
2		×	×			×		×	×
3	×	×				×			
4				×	×				
5							×	×	
6		×					×		

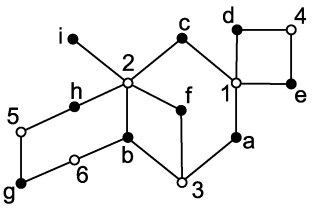


Fig. 1. A relation and the corresponding bipartite graph

In this section, we will first discuss connectivity issues, then illustrate and give our process to decompose the bipartite graph.

3.1 Decomposing the bipartite graph into connected components

When the bipartite graph $Bip(\mathcal{R})$ is not connected, our process can be applied separately (or in parallel) to each connected component. The lattice obtained is characteristic: when the top and bottom elements are removed from the Hasse diagram, the resulting diagram is a set of disjoint lattices, with a one-to-one correspondence between the connected components of $Bip(\mathcal{R})$ and the lattices obtained.

Figure 2 shows such a disconnected bipartite graph, its relation, and the corresponding lattice.

Note that trivially, if a connected component has only one vertex, this means that the corresponding row or column of the relation is empty: such a component corresponds to a lattice with only one element.

In the rest of the paper, we will consider only relations whose bipartite graph is connected.

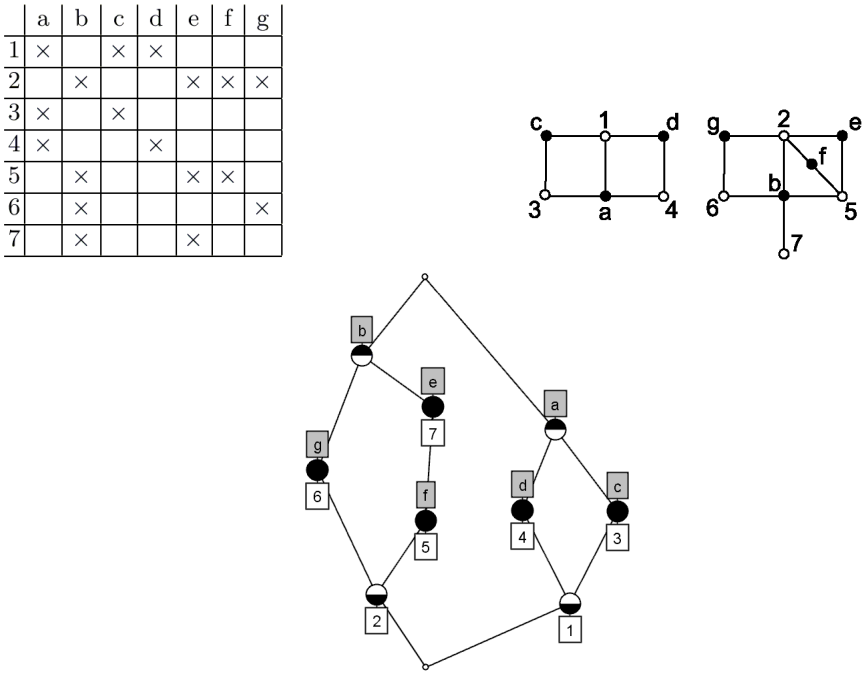


Fig. 2. A disconnected bipartite graph, its relation, and the corresponding characteristic lattice.

3.2 Illustrating the two decomposition steps

In order to make the process we use as clear as possible, we will first describe what happens when one decomposition step is applied for each of the two decompositions involved (using clique separators of size one or of size two).

It is important to understand, however, that to ensure a good (linear) time complexity, each of the two decompositions is computed globally in a single linear-time pass.

Step with an articulation point

The removal of an articulation point $\{x\}$ in a connected bipartite graph G results in components C_1, ..., C_k, which correspond to a partition $V = C_1 + ... + C_k + \{x\}$ of $Bip(\mathcal{R})$. After a decomposition step using $\{x\}$, x is preserved, with its local neighborhood, in each component, so that G is replaced by k subgraphs $G(C_1 \cup \{x\})$, ..., $G(C_k \cup \{x\})$.

Example 1. In the input graph of Figure 3, vertex 1 defines an articulation point that induces two connected components $\{2, 3, a, b, c\}$ and $\{4, d, e\}$. The decomposition step results into subgraphs $G(\{1, 2, 3, a, b, c\})$ and $G(\{1, 4, d, e\})$.

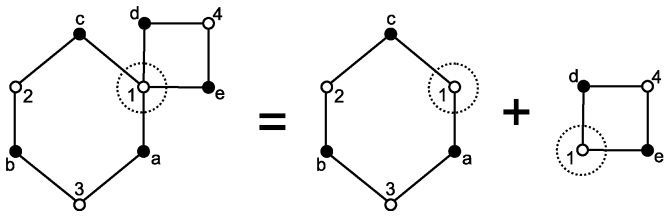


Fig. 3. Decomposition by articulation point {1}.

Step with a separator of size two

When the removal of a clique minimal separator $\{x, y\}$ in a connected bipartite graph G results into components C_1,..., C_k, corresponding to a partition $V = C_1 + ... + C_k + \{x, y\}$. The decomposition step replaces G with $G(C_1 \cup \{x, y\})$, ..., $G(C_k \cup \{x, y\})$.

Example 2. In the input graph of Figure 4, $\{2, b\}$ is a clique minimal separator of size two that induces two connected components $\{1, 2, 3, a, b, c, f\}$ and $\{2, 5, 6, b, g, h\}$.

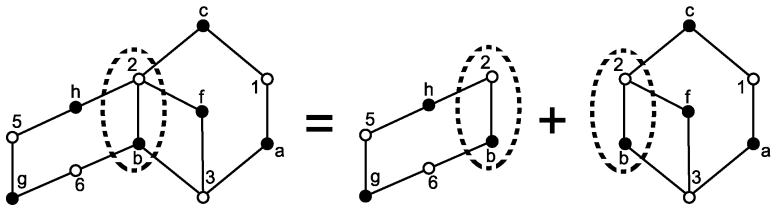


Fig. 4. Decomposition by clique separator {2,b}

3.3 Ordering the steps

A clique minimal separator of size two may include an articulation point. Thus it is important to complete the decomposition by the articulation points first, and then go on to decompose the obtained subgraphs using their clique separators of size two.

Example 3. In the input graph of Figure 5, $\{2\}$ is an articulation point included in clique minimal separator $\{2, b\}$. The decomposition will begin with $\{2\}$, inducing components $\{2, i\}$ and $\{1, 2, 3, 5, 6, a, b, c, f, g, h\}$. As there remains no articulation point in these resulting components, the second component will be then decomposed by $\{2, b\}$ into $\{1, 2, 3, a, b, c, f\}$ and $\{2, 5, 6, b, g, h\}$.

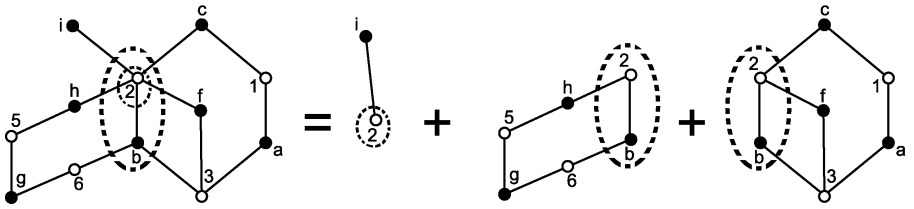


Fig. 5. Articulation point {2} is processed before clique separator {2,b}

After the bipartite graph decomposition is completed, we will obtain subgraphs with no remaining clique minimal separator, and the corresponding subrelations with their associated lattices.

Example 4. Figure 6 shows that the input graph of Figure 1 is decomposable into four bipartite subgraphs: $G_1 = G(\{1, 2, i\})$, $G_2 = G(\{2, 5, 6, b, g, h\})$, $G_3 = G(\{1, 2, 3, a, b, c, f\})$ and $G_4 = G(\{1, 4, d, e\})$.

Note that in the general case all subgraphs obtained have at least two vertices, since at least one vertex of a separator is copied into a component which has at least one vertex.

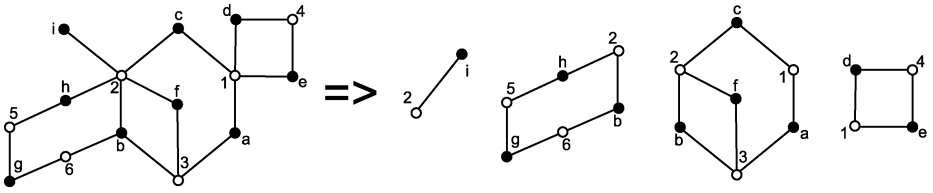


Fig. 6. Complete decomposition of a bipartite graph

3.4 The global decomposition process

To obtain the entire decomposition of a connected bipartite graph, we will thus first decompose the graph using all articulation points, and then decompose each of the subgraphs obtained using all its clique separators of size 2.

The articulation points (clique minimal separators of size one) can be found by a simple depth-first search [6], as well as the corresponding decomposition of the graph (called decomposition into biconnected components).

The search for clique separators of size two corresponds to a more complicated algorithm, described in [5]: all separators of size 2 are obtained, whether they are cliques or not.

Once this list of separators is obtained, it is easy to check which are joined by an edge. The desired decomposition can then be obtained easily.

In both cases, the set of clique separators is output.

Both algorithms run in linear time, so the global complexity is in $O(|\mathcal{R}|)$ to obtain both the set of subgraphs and the set of clique separators of the original graph.

3.5 Sub-patterns defined in the matrix

When the clique separators involved do not overlap and each defines exactly two connected components, this decomposition of the bipartite graph partitions the graph and the underlying relation. This results in a significant pattern of their binary matrix. As the components obtained are pairwise disconnected, the matrix can be reorganized in such a way that zeroes are gathered into blocks. Two components may appear as consecutive blocks, linked by a row corresponding to the articulation point that has been used to split them, or linked by one cell giving the edge between the two vertices of a size two clique minimal separator.

In the general case, this pattern can occur in only some parts of the matrix, and different patterns can be defined according to the different separators which the user chooses to represent.

Example 5. The first of the two matrices below corresponds to our running example from Figure 1 and has been reorganized, following the decomposition, which results in the second matrix. Notice how {1} is an articulation point, so

row 1 is shared by blocs $231 \times bacf$ and $14 \times de$; and how $\{2, b\}$ is a clique separator of size two, so cell $[2, b]$ is the intersection of blocs $562 \times ghb$ and $231 \times bacf$. $[2, i]$ is not integrated into the pattern, because separator $\{2, b\}$ of $Bip(\mathcal{R})$ defines 3 connected components: $\{2, 5, 6, b, g, h\}$, $\{i\}$ and $\{1, 3, 4, a, c, d, e, f\}$.

	a	b	c	d	e	f	g	h	i
1	×		×	×	×				
2		×	×			×		×	×
3	×	×				×			
4				×	×				
5							×	×	
6		×					×		

	i	g	h	b	a	c	f	d	e
5		×	×						
6		×		×					
2	×		×	×		×	×		
3				×	×		×		
1					×	×		×	×
4								×	×

We will now describe a process to organize the lines and columns of the matrix with such patterns. We will construct a meta-graph (introduced in [7] as the '*atom graph*'), whose vertices represent the subgraphs obtained by our decomposition, and where there is an edge between two such vertices if the two subgraphs which are the endpoints have a non-empty intersection which is a clique minimal separator separating the corresponding two subgraphs in the original bipartite graph. In this meta-graph, choose a chordless path; the succession of subgraphs along this path will yield a succession of rectangles in the matrix which correspond to a pattern.

Example 6. Figure 7 gives the meta-graph for our running example from Figure 1. Chordless path $(\{2, 5, 6, b, g, h\}, \{1, 2, 3, a, b, c, \}, \{1, 4, d, e\})$ was chosen for the patterning. Another possible chordless path would be $(\{2, i\}, \{1, 2, 3, a, b, c, \}, \{1, 4, d, e\})$. Finding a chordless path in a graph can be done in linear time; the meta-graph has less than $min(|\mathcal{A}|, |\mathcal{O}|)$ elements, so finding such a path costs less than $(min(|\mathcal{A}|, |\mathcal{O}|))^2$.

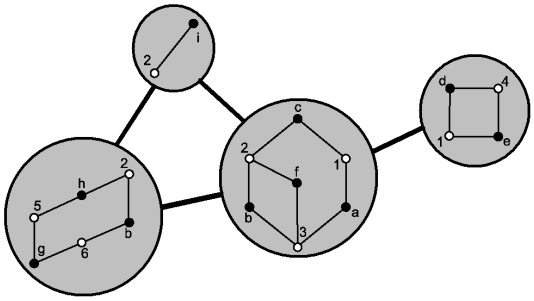


Fig. 7. Meta-graph for graph from Figure 1

3.6 Decomposing the lattice

We will now examine how the set of concepts is modified and partitioned into the subgraphs obtained. As clique minimal separators are copied in all the components induced, most of the concepts will be preserved by the decomposition. Furthermore, only initial concepts including a vertex of a clique minimal separator may be affected by the decomposition.

Definition 1. *We will say that a maximal biclique is a star maximal biclique if it contains either exactly one object or exactly one attribute. This single object or attribute will be called the center of the star.*

Lemma 1. *A star maximal biclique* $\{x\} \cup N(x)$ *of* $Bip(\mathcal{R})$ *is an atomic concept of* $\mathcal{L}(\mathcal{R})$ *(atom or co-atom), unless* x *is universal in* $Bip(\mathcal{R})$*. More precisely,* $\{x\} \times N(x)$ *is a atom if* $x \in \mathcal{O}$ *and* $N(x) \neq \mathcal{A}$*, or* $N(x) \times \{x\}$ *is a co-atom if* $x \in \mathcal{A}$ *and* $N(x) \neq \mathcal{O}$*.*

Proof. Let $\{x\} \cup N(x)$ be a star maximal biclique of $Bip(\mathcal{R})$. As a maximal biclique, it corresponds to a concept of $\mathcal{L}(\mathcal{R})$. Suppose the star has $x \in \mathcal{O}$ as center . As a star, it contains no other element of $\mathcal{O}$; as a biclique, it includes all $N(x) \subseteq \mathcal{A}$, and no other element of $\mathcal{A}$ by maximality. The corresponding concept is $\{x\} \times N(x)$ which is obviously the first concept from bottom to top including x. As the biclique is maximal, and as x is not universal, this concept cannot be the bottom of $\mathcal{L}(\mathcal{R})$ but only an atom. A similar proof holds for $x \in \mathcal{A}$ and co-atomicity.

We will now give the property which describes how the maximal bicliques are dispatched or modified by the decomposition. In the next Section, we will give a general theorem and its proof, from which these properties can be deduced.

Property 1. Let $G = (X + Y, E)$ be a bipartite graph, let S be a clique minimal separator of G which decomposes G into subgraphs G_1, ..., G_k. Then:

1. $\forall x \in S$, $\{x\} \cup N_G(x)$ is a star maximal biclique of G.
2. $\forall x \in S$, $\{x\} \cup N_G(x)$ is not a maximal biclique of any G_i.
3. $\forall x \in S$, $\{x\} \cup N_{G_i}(x)$ is a biclique of G_i, but it is maximal in G_i iff it is not strictly contained in any other biclique of G_i.
4. All the maximal bicliques of G which are not star bicliques with any $x \in S$ as a center are partitioned into the corresponding subgraphs.

With the help of Lemma 1, this property may be translated in terms of lattices. Given a relation $\mathcal{R}$, its associated graph G, its lattice $\mathcal{L}(\mathcal{R})$, and a decomposition step of G into some G_is by articulation point $\{x\}$:

If $x \in \mathcal{O}$ (resp. $\in \mathcal{A}$) is an articulation point of G, $\{x\} \times N_G(x)$ (resp. $N_G(x) \times \{x\}$) is a concept of $\mathcal{L}(\mathcal{R})$. After the decomposition step, in each subgraph G_i of G, either this concept becomes $\{x\} \times N_{G_i}(x)$, or this concept disappears from G_i; this latter case occurs when there is in G_i some $x' \in \mathcal{O}$, the introducer of which appears after the introducer of x in $\mathcal{L}(\mathcal{R})$, from bottom to top (resp. from top

to bottom if $x, x' \in \mathcal{A}$). Every other concept will appear unchanged in exactly one lattice associated with a subgraph G_i.

The same holds for each vertex of a size two clique minimal separator.

Example 7. Figure 8 illustrates a decomposition step with articulation point $\{1\}$ using the graph from Figure 3. Concept $\{1,4\} \times \{d,e\}$ disappears from the first component $\{1,2,3,a,b,c\}$, but remains in the second component $\{1,4,d,e\}$.

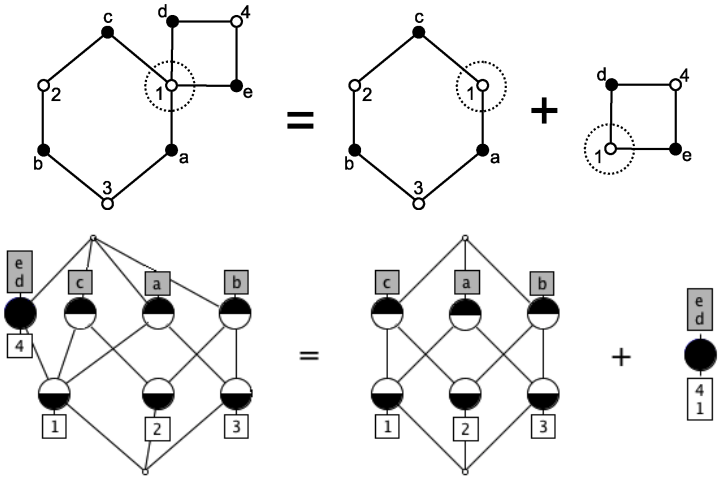


Fig. 8. Example of lattice decomposition using articulation point $\{1\}$.

Example 8. Figure 9 illustrates a decomposition step with clique separator $\{2,b\}$ using the graph from Figure 4. Concept $\{2\} \times N(2)$ is duplicated into components $\{2,5,6,b,g,h\}$ and $\{1,2,3,a,b,c,f\}$; concept $N(b) \times \{b\}$ will appear as $\{2,6\} \times \{b\}$ in the first component, but not in the second one, as $\{2,3,b\}$ is a biclique included in maximal biclique $\{2,3,b,f\}$ of G.

Remark 1. The smaller lattices obtained can not be called sublattices of the initial lattice as some of their elements may not be the same: for example, in Figure 9, $\{2\} \times \{b,c,f\}$ is an element of the third smaller lattice $\mathcal{L}(G_3)$ but is not an element of the initial lattice $\mathcal{L}(G)$.

4 Reconstructing the lattice

We will now explain how, given the subgraphs obtained by clique decomposition, as well as the corresponding subrelations and subsets of concepts, we can reconstruct the set of concepts of the global input bipartite graph. We will then go on to explain how to obtain the Hasse diagram of the reconstructed lattice.

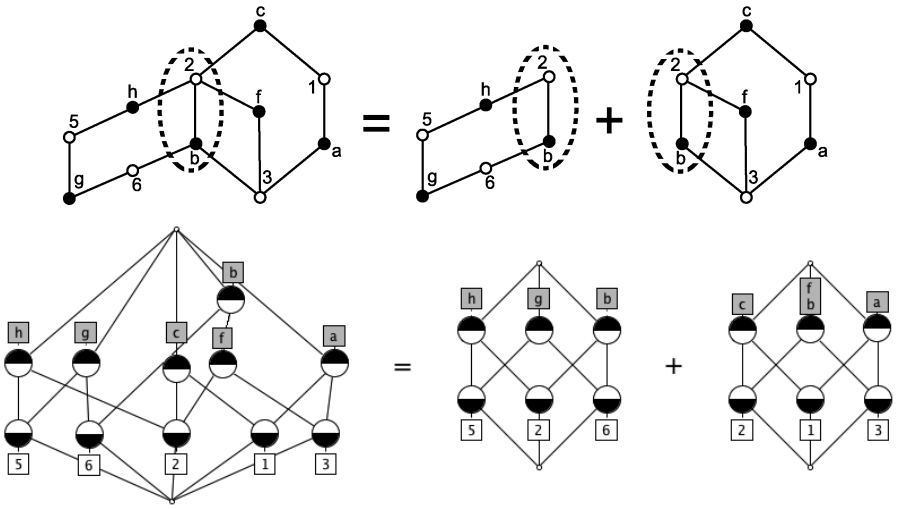


Fig. 9. Example of lattice decomposition using clique separator $\{2, b\}$.

4.1 Reconstructing the set of concepts

We will use the following Theorem, which describes the concepts of the global lattice.

Theorem 1. *Let $G = (X + Y, E)$ be a bipartite graph, let $\Sigma = \{s_1, ...s_h\}$ be the set of all the vertices which belong to a clique separator of G, let $G_1, ...G_k$ be the set of subgraphs obtained by the complete corresponding clique separator decomposition. Then:*

1. *For every $s \in \Sigma$, $\{s\} \cup N_G(s)$ is a star maximal biclique of G.*
2. *Any maximal biclique of a subgraph G_i which is not a star with a vertex of Σ as center is also a maximal biclique of G.*
3. *There are no other maximal bicliques in G: $\forall s \in \Sigma$, no other star maximal biclique of G_i with center s is a star maximal biclique of G, and these are the only maximal bicliques of some graph G_i which are not also maximal bicliques in G.*

Proof.

1. For every $s \in \Sigma$, $\{s\} \cup N_G(s)$ is a star maximal biclique of G:
   Case 1: s is an articulation point, let G_i, G_j be two graphs which s belongs to; s must be adjacent to some vertex y_i in G_i and to some vertex y_j in G_j. Suppose $\{s\} \cup N_G(s)$ is not a maximal biclique: there must be a vertex z in G which sees y_i and y_j, but then $\{s\}$ cannot separate y_i from y_j, a contradiction.
   Case 2: s is not an articulation point, let s' be a vertex of S such that

$\{s, s'\}$ is a clique separator of G, separating G_i from G_j. s must as above see some vertex y_i in G_i and some vertex y_j in G_j. Suppose $\{s\} \cup N_G(s)$ is not maximal: there must be some vertex w in G which sees all of $N_G(s)$, but w must see y_i and y_j, so $\{s, s'\}$ cannot separate G_i from G_j.

2. Let B be a non-star maximal biclique of G_i, containing $o_1, o_2 \in \mathcal{O}$ and $a_1, a_2 \in \mathcal{A}$. Suppose B is not maximal in G: there must be a vertex y in $G - B$ which augments B. Let y be in G_j, *wlog* $y \in \mathcal{A}$: y must see o_1 and o_2. Since G_i is a maximal subgraph with no clique separator, $G_i + \{y\}$ must have a clique separator. Therefore $N(y)$ must be a clique separator of this subgraph, but this is impossible, since y sees two non-adjacent vertices of G_i.
3. Any star maximal biclique B of G_i whose center is not in Σ is also a star maximal biclique of G: suppose we can augment B in G.
   Case 1: v sees an extra vertex w; $G_i + \{w\}$ contains as above a clique separator, which is impossible since $N(w) = v$ and $v \notin S$.
   Case 2: A vertex z of G_j is adjacent to all of $N(v)$: again, $G + \{z\}$ contains a clique separator, so $N(z)$ is a clique separator, but that is impossible since $N(z)$ contains at least two non-adjacent vertices.
4. For $s \in \Sigma$, no star maximal biclique of G_i is a star maximal biclique of G: let B be a star maximal biclique of G_i, with $s \in \Sigma$ as center. $s \in \Sigma$, so s belongs to some clique separator which separates G_i from some graph G_j. s must see a vertex y_j in G_j, so $B + \{y_j\}$ is a larger star including B: B cannot be maximal in G.

Example 9. We illustrate Theorem 1 using graph G from Figure 6, whose decomposition yields subgraphs G_1, ..., G_4, with $G_1 = G(\{1, 2, i\})$, $G_2 = G(\{2, 5, 6, b, g, h\})$, $G_3 = G(\{1, 2, 3, a, b, c, f\})$ and $G_4 = G(\{1, 4, d, e\})$. Finally, $\Sigma = \{1, 2, b\}$. The corresponding lattices are shown in Figure 10, and their concepts are presented in the table below. In this table, braces have been omitted; symbol $\Rightarrow$ represents a concept of the considered subgraph G_i which is identical to a concept of G (there can be only one $\Rightarrow$ per row); the other concepts of the subgraphs will not be preserved in G while recomposing.

$\mathcal{L}(G)$	$\mathcal{L}(G_1)$	$\mathcal{L}(G_2)$	$\mathcal{L}(G_3)$	$\mathcal{L}(G_4)$	star max. biclique of G ?
$1 \times acde$			$1 \times ac$		yes
$2 \times bcfhi$	$2 \times i$	$2 \times bh$	$2 \times bcf$		yes
$3 \times abf$			$\Rightarrow$		
$14 \times de$				$\Rightarrow$	
$5 \times gh$		$\Rightarrow$			
$6 \times bg$		$\Rightarrow$			
$13 \times a$			$\Rightarrow$		
$236 \times b$		$26 \times b$			yes
$12 \times c$			$\Rightarrow$		
$23 \times bf$			$\Rightarrow$		
$56 \times g$		$\Rightarrow$			
$25 \times h$		$\Rightarrow$			

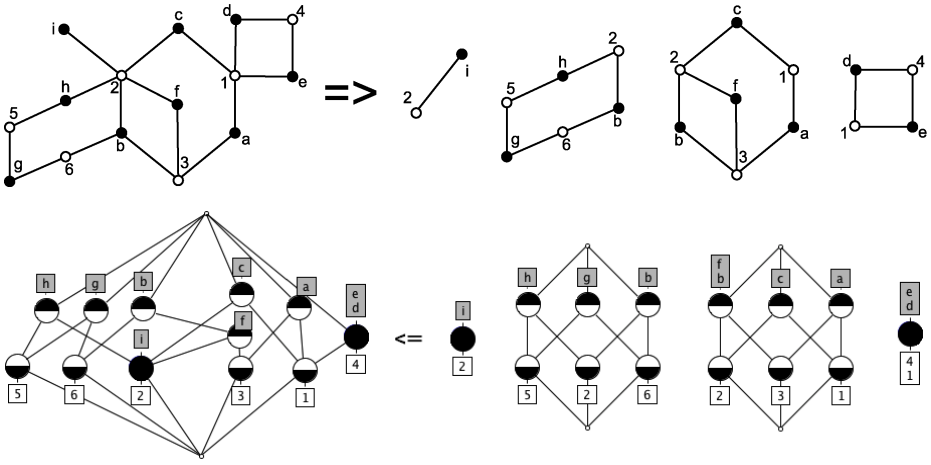


Fig. 10. Reconstruction of a lattice

According to Theorem 1, the steps to reconstruct the maximal concepts of the global lattice from the concepts of the smaller lattices are:

1. Compute Σ, the set of attributes and objects involved in a clique minimal separator. (In our example, $\Sigma = \{1, 2, b\}$.)
2. Compute the maximal star bicliques for all the elements of Σ. (In our example, we will compute star maximal bicliques $1 \times acde$, $2 \times bcfhi$ and $26 \times b$.)
3. For each smaller lattice, remove from the set of concepts the atoms or co-atoms corresponding to elements of Σ; maintain all the other concepts as concepts of the global lattice. (In our example, for $\mathcal{L}(G_3)$, we will remove $1 \times ac$ and $2 \times bcf$, and maintain $3 \times abf$, $13 \times a$, $12 \times c$ and $23 \times bf$ as concepts of $\mathcal{L}(G)$.)

Step 1 requires $O(|\mathcal{R}|)$ time. Step 2 can be done while computing the smaller lattices; Step 3 costs constant time per concept. Thus the overall complexity of the reconstruction is in $O(|\mathcal{R}|)$ time.

4.2 Reconstructing the edges of the Hasse diagram

According to Theorem 1, the maximal bicliques which are not star maximal bicliques with a vertex of Σ as center are preserved; therefore, the corresponding edges between the elements of the lattice are also preserved. In the process

described below, we will refer to labels in the lattice as being the 'reduced' labels, such as the ones used in our lattice figures throuhout this paper.

To define the edges of the Hasse diagram of lattice $\mathcal{L}(G)$, we will, for each smaller lattice $\mathcal{L}(G_i)$:

- find each atom (or co-atom) which corresponds to an element σ of Σ (such as 2 or b for $\mathcal{L}(G_3)$ in our example).
- If σ shares its label with some non-elements of Σ, remove all elements of Σ from the label. (In our example for $\mathcal{L}(G_3)$, bf becomes f). If σ does not share its label with some non-elements of Σ, remove the atom or co-atom. (In our example for $\mathcal{L}(G_3)$, remove element 2).
- Maintain the remaining edges as edges of $\mathcal{L}(G)$.
- Compute the neighborhood in $\mathcal{L}(G)$ of each atom or co-atom which corresponds to an element of Σ.

All this can be done in polynomial time: there are at most $|\mathcal{A}| + |\mathcal{O}|$ vertices in Σ, and the corresponding edges can be added in $O((|\mathcal{A}| + |\mathcal{O}|)^2|\mathcal{R}|)$.

5 Vertical decomposition as a layout tool

When there is a size two clique separator in the bipartite graph which divides the graph into two components, the concepts which are not involved in the separator can be displayed on the two sides of the separator, thus helping to minimize the number of line crossings in the Hasse diagram.

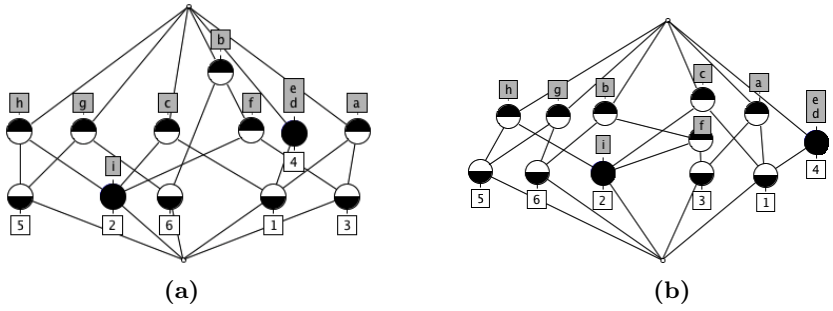


Fig. 11. (a) Lattice constructed by Concept Explorer using the minimal intersection layout option (8 crossings). **(b)** Lattice re-drawn using the information on clique separators (5 crossings).

To illustrate this, we have used our running example with 'Concept Explorer' [1], which is a very nice and user-friendly tool for handling lattices. Notice however how clique separator $\{1, d\}$ is better displayed when put at the right extremity.

Figure 11 shows the lattice as proposed by Concept Explorer, and then redrawn with insight on the clique separators of the bipartite graph.

The same technique of course also applies when there is a succession of such clique separators.

Let us add that if moreover both lattices are planar, as discussed in [4], merging the two lattices obtained using the clique separator as central will preserve planarity.

6 Conclusion and perspectives

We have used a graph method, clique minimal separator decomposition, to provide simple tools which can help reduce the time spent computing the elements of a lattice, as well as improve the drawing of its Hasse diagram.

When there is no clique separator in the bipartite graph, it could be interesting to investigate restricting the relation to a subgraph or partial subgraph which does have one.

We leave open the question of characterizing, without computing the relation, the lattices whose underlying bipartite graph has a clique minimal separator.

Acknowledgments

The authors sincerely thank all the referees for their useful suggestions and questions.

References

1. Concept Explorer. Downloadable at http://sourceforge.net/projects/conexp/, version 1.3 (Java), 20/12/2009.
2. Berry A., Sigayret A.: *Representing a concept lattice by a graph.* Discrete Applied Mathematics, 144(1-2):27–42, 2004.
3. Berry A., Pogorelcnik R., Simonet G.: *An introduction to clique minimal separator decomposition.* Algorithms, 3(2):197–215, 2010.
4. Eschen E.M., Pinet N., Sigayret A.: *Consecutive-ones: handling lattice planarity efficiently.* CLA'07, Montpellier (Fr), 2007.
5. Hopcroft J. E., Tarjan R. E.: *Dividing a graph into triconnected components.* SIAM J. Comput., 2(3):135–158, 1973.
6. Hopcroft J. E., Tarjan R. E.: *Efficient algorithms for graph manipulation [H] (Algorithm 447).* Commun. ACM, 16(6):372–378, 1973.
7. Kaba B., Pinet N., Lelandais G., Sigayret A., Berry A.: Clustering gene expression data using graph separators. *In Silico Biology*, 7(4-5):433–52, 2007.
8. Leimer H.-G.: *Optimal decomposition by clique separators.* Discrete Mathematics, 113(1-3):99–123, 1993.
9. Tarjan R. E.: *Decomposition by clique separators.* Discrete Mathematics, 55(2):221–232, 1985.

Building up Shared Knowledge with Logical Information Systems

Mireille Ducassé[1], Sébastien Ferré[2], and Peggy Cellier[1]

[1] IRISA-INSA de Rennes, France,
{ducasse, cellier}@irisa.fr
[2] IRISA-University of Rennes 1, France
ferre@irisa.fr

Abstract. Logical Information Systems (LIS) are based on Logical Concept Analysis, an extension of Formal Concept Analysis. This paper describes an application of LIS to support group decision. A case study gathered a research team. The objective was to decide on a set of potential conferences on which to send submissions. People individually used Abilis, a LIS web server, to preselect a set of conferences. Starting from 1041 call for papers, the individual participants preselected 63 conferences. They met and collectively used Abilis to select a shared set of 42 target conferences. The team could then sketch a publication planning. The case study provides evidence that LIS cover at least three of the collaboration patterns identified by Kolfschoten, de Vreede and Briggs. Abilis helped the team to build a more complete and relevant set of information (Generate/Gathering pattern); to build a shared understanding of the relevant information (Clarify/Building Shared Understanding); and to quickly reduce the number of target conferences (Reduce/Filtering pattern).

1 Introduction

Group work represents a large amount of time in professional life while many people feel that much of that time is wasted. Lewis [13] argues that this amount of time is even going to increase because problems are becoming more complex and are meant to be solved in a distributed way. Each involved person has a local and partial view of the problem, no one embraces the whole required knowledge. Lewis also emphasizes that it is common that *"groups fail to adequately define a problem before rushing to judgment"*. Building up shared knowledge in order to gather relevant distributed knowledge of a problem is therefore a crucial issue.

Logical Information Systems (LIS) are based on Logical Concept Analysis (LCA), an extension of Formal Concept Analysis (FCA). In a previous work [5], Camelis, a single-user logical information system, has been shown useful to support serene and fair meetings. This paper shows how Abilis, a LIS web server that implements OnLine Analytical Processing (OLAP [3]) features, can be applied to help build shared knowledge among a group of skilled users.

The presented case study gathered a research team to decide on a publication strategy. Starting from 1041 call for papers, each team member on his own

 Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

preselected a set of conferences matching his own focus of interest. The union of individual preselections still contained 63 conferences. Then, participants met for an hour and a half and collectively built a shared set of 42 target conferences. For each conference, the team shared a deep understanding of why it was relevant. The team could sketch a publication planning in a non-conflictual way.

Kolfschoten, de Vreede and Briggs have classified collaboration tasks into 16 collaboration patterns [12]. The contribution of this paper is to give evidences that LIS can significantly support three of these patterns which are important aspects of decision making, namely *Generate/Gathering*, *Clarify/Building Shared Understanding* and *Reduce/Filtering.* Firstly, the navigation and filtering capabilities of LIS were helpful to detect inconsistencies and missing knowledge. The updating capabilities of LIS enabled participants to add objects, features and links between them on the fly. As a result the group had a more complete and relevant set of information (Generate/Gathering pattern). Secondly, the compact views provided by LIS and the OLAP features helped participants embrace the whole required knowledge. The group could therefore build a shared understanding of the relevant information which was previously distributed amongst the participants (Clarify/Building Shared Understanding pattern). Thirdly, the navigation and filtering capabilities of LIS were relevant to quickly converge on a reduced number of target conferences (Reduce/Filtering pattern).

In the following, Section 2 briefly introduces logical information systems. Section 3 describes the case study. Section 4 gives detailed arguments to support the claim that logical information systems help build up shared knowledge. Section 5 discusses related work.

2 Logical Information Systems

Logical Information Systems (LIS) [7] belong to a paradigm of information retrieval that combines querying and navigation. They are formally based on a logical generalization of Formal Concept Analysis (FCA) [8], namely Logical Concept Analysis (LCA) [6]. In LCA, logical formulas are used instead of sets of attributes to describe objects. LCA and LIS are generic in that the logic is not fixed, but is a parameter of those formalisms. Logical formulas are also used to represent queries and navigation links. The concept lattice serves as the navigation structure: every query leads to a concept, and every navigation link leads from one concept to another. The query can be modified in three ways: by formula edition, by navigation (selecting features in the index in order to modify the query) or by examples. Annotations can be performed in the same interface. Camelis[3] has been developed since 2002; a web interface, Abilis [4], has recently been added. It incorporates display paradigms based on On-Line Analytical Processing (OLAP). Instead of being presented as a list of objects, an extent can be partitioned as an OLAP cube, namely a multi-dimensional array [1].

[3] see `http://www.irisa.fr/LIS/ferre/camelis/`
[4] http://ledenez.insa-rennes.fr/abilis/

3 The Case Study

The reported case study gathered 6 participants, including the 3 authors, 4 academics and 2 PhD students. All participants were familiar with LIS, 4 of them had not previously used a LIS tool as a decision support system. The objective was to identify the publishing strategy of the team: in which conferences to submit and why. This has not been a conflictual decision, the group admitted very early that the set of selected conferences could be rather large provided that there was a good reason to keep each of them.

One person, the facilitator, spent an afternoon organizing the meeting and preparing the raw data as well as a logical context according to the objective. She collected data about conference call for papers of about a year, related to themes corresponding to the potential area of the team, from WikiCFP, a semantic wiki for Calls For Papers in science and technology fields [5]. There were 1041 events: conferences, symposiums, workshops but also special issues of journals.

Then every participant, on its own, spent between half an hour to two hours to browse the context, update it if necessary and preselect a number of conferences (Section 3.1). The group met for one hour and a half. It collaborately explored the data and selected a restricted set of conferences (Section 3.2). After the meeting, every participant filled a questionnaire. The context used for the case study can be freely accessed [6].

3.1 Distributed Individual Preselection and Update

When the context was ready, every participant was asked to preselect a set of conferences that could be possible submission targets. The instruction was to be as liberal as wanted and in case of doubt to label the conference as a possible target.

During this phase, each of the academics preselected 20 to 30 conferences and each of the PhD students preselected around 10 conferences. Each participant had his own "basket". There were overlappings, altogether 63 conferences were preselected. Participants also introduced new conferences and new features, for example, the ranking of the Australian CORE association [7] (Ranking), and the person expected to be a possible first author for the target conference (Main author).

Figure 1 shows a state of Abilis during the preselection phase. LIS user interfaces give a *local view* of the concept lattice, centered on a focus concept. The local view is made of three parts: (1) the query (top left), (2) the extent (bottom right), and (3) the index (bottom left). The query is a logical formula that typically combines attributes (e.g., `Name`), patterns (e.g., `contains "conference"`), and Boolean connectors (`and`, `or`, `not`). The extent is the set of objects that are matched by the query, according to logical subsumption. The extent identifies

[5] http://www.wikicfp.com/cfp/
[6] http://ledenez.insa-rennes.fr/abilis/, connect as guest, load `Call for papers`.
[7] http://core.edu.au/index.php/categories/conference rankings

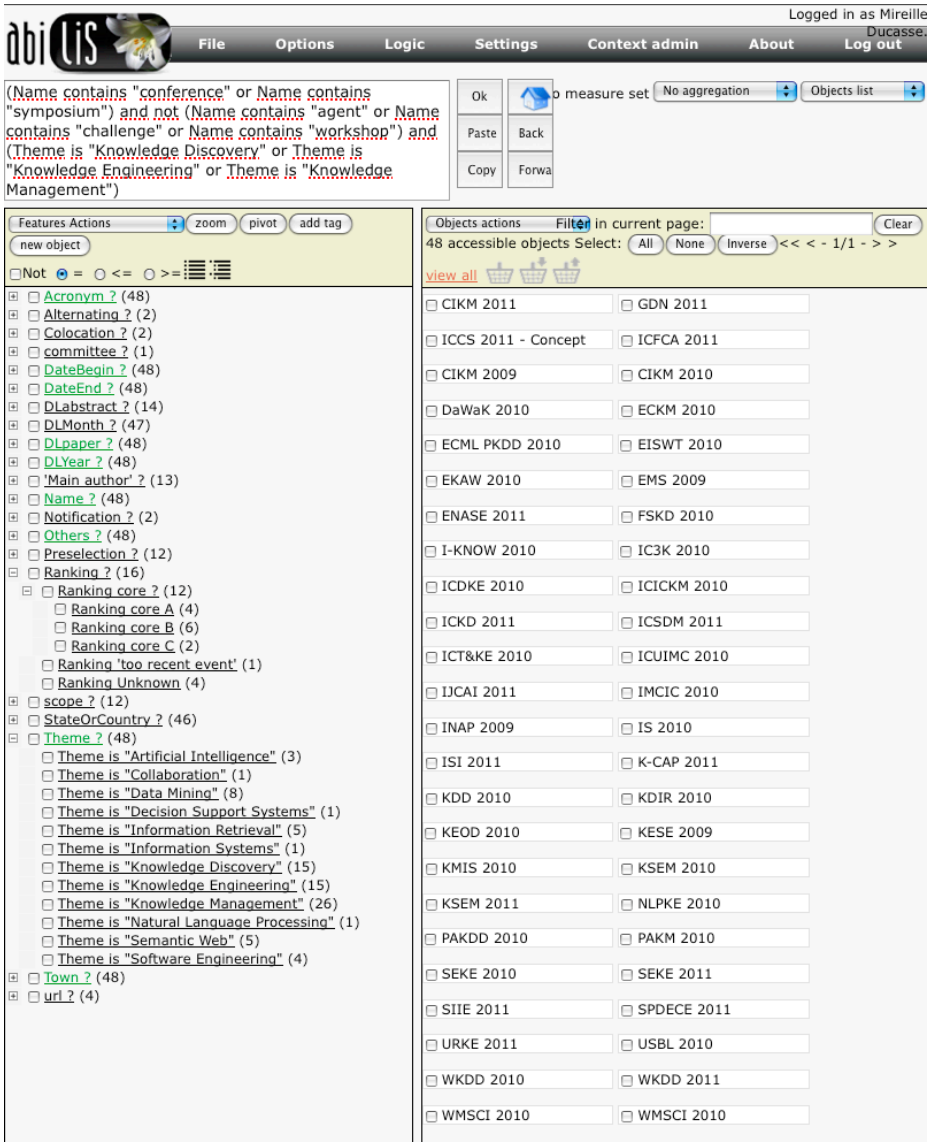


Fig. 1. Snapshot of Abilis during preselection: a powerful query

the focus concept. Finally, the index is a set of features, taken from a finite subset of the logic, and is restricted to features associated to at least one object in the extent. The index plays the role of a summary or inventory of the extent, showing which kinds of objects there are, and how many of each kind there are (e.g., in Figure 1, 8 objects in the extent have `data mining` as a theme). In the index, features are organized as a taxonomy according to logical subsumption.

The query area (top left) shows the current selection criteria: `(Name contains "conference" or Name contains "symposium") and not (Name contains "agent" or Name contains "challenge" or Name contains "workshop") and (Theme is "Knowledge Discovery" or Theme is "Knowledge Engineering" or Theme is "Knowledge Management")`. Note that the query had been obtained solely by clicking on features of the index (bottom left). Let us describe how it had been produced. Initially there were 1041 objects. Firstly, opening the `Name ?` feature, the participant had noticed that names could contain "conference" or "symposium" but also other keywords such as "special issue". He decided to concentrate on conferences and symposiums by clicking on the two features and then on the `zoom` button. The resulting query was `(Name contains "conference" or Name contains "symposium")` and there were 495 objects in the extent. However, the displayed features under `Name ?` showed that there were still objects whose name in addition to "conference" or "symposium" also contained "agent", "challenge" or "workshop". He decided to filter them out by clicking on the three features then on the `Not` button then on the `zoom` button. The resulting query was `(Name contains "conference" or Name contains "symposium") and not (Name contains "agent" or Name contains "challenge" or Name contains "workshop")` and there were 475 objects in the extent. He opened the `Theme ?` feature, clicked on the three subfeatures containing "Knowledge", then on the `zoom` button. The resulting query is the one displayed on Figure 1 and there are 48 objects in the displayed extent.

In the extent area (bottom right), the 48 acronyms of the selected conferences are displayed. In the index area, one can see which of the features are filled for these objects. The displayed features have at least 1 object attached to them. The number of objects actually attached to them is shown in parentheses. For example, only 14 of the preselected conferences have an abstract deadline. All of them have an acronym, a date of beginning, a date of end, a date for the paper deadline, a name, some other (not very relevant) information, as well as at least a theme and a town. The features shared by all selected objects have that number in parentheses (48 in this case). For the readers who have a color printout, these features are in green. The other features are attached to only some of the objects. For example, only 16 objects have a ranking attached to them: 4 core A, 6 core B, 2 core C, 1 'too recent event', 4 unknown (to the Core ranking).

One way to pursue the navigation could be, for example, to click on `Ranking ?` to select the conferences for which the information is filled. Alternatively, one could concentrate on the ones for which the ranking is not filled, for example

to fill in this information on the fly for the conferences which are considered interesting.

Another way to pursue the navigation could be, for example, to notice that under the `Theme ?` feature, there are more than the selected themes. One can see that among the selected conferences, one conference is also relevant to the `Decision Support Systems` theme. One could zoom into it, this would add `and Theme is "Decision Support Systems"` to the query ; the object area would then display the relevant conference (namely GDN2011).

3.2 Collaborative Data Exploration, Update and Selection

The group eventually had a physical meeting where the current state of the context was constantly displayed on a screen.

Using the navigation facilities of Abilis, the conferences were examined by decreasing ranking. Initially, the group put in the selection all the A and A+ preselected conferences. After some discussions, it had, however, been decided that Human Computer Interaction (HCI) was too far away from the core of the team's research. Subsequently, the HCI conferences already selected were removed from the selection. For the conferences of rank B, the team decided that most of them were pretty good and deserved to be kept in the selection. For the others, the group investigated first the conferences without ranking and very recent, trying to identify the ones with high selection rate or other good reasons to put them in the selection. Some of the arguments have been added into the context. Some others were taken into account on the fly to select some conferences but they seemed so obvious at the time of the discussion that they were not added in the context.

Figure 2 shows the selection made by the group at a given point. In the extent area, on the right hand side, the selected objects are partitioned according to the deadline month and the anticipated main author thanks to the OLAP like facilities of Abilis [1]. Instead of being presented as a list of objects, an extent can be partitioned as an OLAP cube, namely a multi-dimensional array. Assuming object features are valued attributes, each attribute can play the role of a dimension, whose values play the role of indices along this dimension. Users can freely interleave changes to the query and changes to the extent view.

The query is `SelectionLIS02Dec and scope international`. Note that the partition display is consistent with the query. When the group added `and scope international` to the query, the national conferences disappeared from the array.

Some conferences, absent from WikiCFP have been entered on the fly at that stage (for example, ICCS 2011 - Concept). Not all the features had been entered for all of them. In particular, one can see in the feature area that only 28 out of 29 had been preselected. Nevertheless, the group judged that the deadline month, the potential main author and the ranking were crucial for the decision process and added them systematically. It is easy to find which objects do not have a feature using `Not` and `zoom`, and then to attach features to them.

Fig. 2. Snapshot of Abilis during collaborative data exploration: a partition deadline month/mainAuthor

One can see that there are enough opportunities for each participant to publish round the year. One can also see at a glance where compromises and decisions will have to be made. For example, PC will probably not be in a position to publish at IDA, ISSTA, KDD and ICCS the same year. Thanks to this global view PC can discuss with potential co-authors what the best strategy could be.

A follow up to the meeting was that participants made a personal publication planning, knowing that their target conferences were approved by the group.

4 Discussion

In this section, we discuss how the reported case study provides evidences that LIS help keep the group focused (Section 4.1) and that LIS also help build up shared knowledge (Section 4.2). As already mentioned, participants filled up a questionnaire after the meeting. In the following, for each item, we introduce the arguments, we present a summary of relevant parts of participant feedbacks, followed by an analysis of the features of LIS that are crucial for the arguments.

4.1 Logical Information Systems Help Keep the Group Focused

It is recognized that an expert facilitator can significantly increase the efficiency of a meeting (see for example [2]). A study made by den Hengst and Adkins [10] investigated which facilitation functions were found the most challenging by facilitators around the world. It provides evidences that facilitators find that *"the most difficult facilitation function in meeting procedures is keeping the group outcome focused."*

In our case study, all participants reported that they could very easily stay focused on the point currently discussed thanks to the query and the consistency between the three views.

As the objective was to construct a selection explicitly identified in Abilis by a feature, the objective of the meeting was always present to everybody and straightforward to bring back in case of digression. Furthermore, even if the context contained over a thousand conferences, thanks to the navigation facilities of LIS, only relevant information was displayed at a given time. Therefore, there was no "noise" and no dispersion of attention, the displayed information was always closely connected to the focus of the discussion.

4.2 Logical Information Systems Help Build Up Shared Knowledge

Kolfschoten, de Vreede and Briggs have identified 6 collaboration patterns: *Generate, Reduce, Clarify, Organize, Evaluate, and Consensus Building* [12]. We discuss in the following three of their 16 sub-patterns for which all participants agreed that they are supported by Abilis in its current stage. For the other sub-patterns, the situation did not demand much with respect to them. For example, the decision to make was not conflictual, the set of selected conferences could be rather large, there was, therefore, not much to experiment about "consensus

building." The descriptions of the patterns in italic are from Kolfschoten, de Vreede and Briggs.

Generate/Gathering: move from having fewer to having more complete and relevant information shared by the group.

Before and during the meeting, information has been added to the shared knowledge repository of the group, namely the logical context. A new theme, important for the team and missing from WikiCFP, has been added: Decision Support Systems. New conferences have been added into the context either by individual participants in the preselection phase or by the group during the selection phase. New features were added. For example, it soon appeared that some sort of conference rankings was necessary. The group added by hand, for the conferences that were selected, the ranking of the Australian Core association. Some conferences were added subsequently, sometimes the ranking was not added at once.

All participants acknowledged that the tool helped the group to set up a set of features which was relevant and reflecting the group's point of view.

The crucial characteristics of LIS for this aspect are those which enable integrated navigation and update.

Firstly, the possibility to update the context while navigating in it enables participants to enhance it on the fly adding small pieces of relevant information at a time. Secondly, for each feature, Abilis displays the number of objects which have it. It is therefore immediate to detect when a feature is not systematically filled. The query `Not <feature>` selects the objects that do not have the feature. Users can then decide if they want to update them. Thanks to the query, as soon as an object is updated, it disappears from the extent. Users can immediately see what remains to be updated. Thirdly, updating the context does not divert from the initial objective. Indeed, the `Back` button allows users to go back to previous queries. Fourthly, the three views (query, features, objects) are always consistent and provide a "global" understanding of the relevant objects. Lastly, in the shared web server, participants can see what information the others had entered. Hence each participant can inspire the others.

For the last aspect, the facilitator inputs were decisive. Participants reported that they did not invent much, they imitated and adapted from what the facilitator had initiated. This is consistent with the literature on group decision and negotiation which emphasizes the key role of facilitators [2].

Clarify/Building Shared Understanding: Move from having less to more shared understanding of the concepts shared by the group and the words and phrases used to express them.

Participants, even senior ones, discovered new conferences. Some were surprised by the ranking of conferences that they had previously overlooked. Participants had a much clearer idea of who was interested in what.

All participants found that the tool helped them understand the points of view of the others.

The crucial characteristics of LIS for this aspect are those which enable to grasp a global understanding at a glance. Firstly, the query, as discussed earlier, helps keep the group focused. Secondly, the consistency between the 3 views helps participants to grasp the situation. Thirdly, irrelevant features are not in the index, the features in the index thus reflect the current state of the group decision. Fourthly, the partitions *à la* OLAP sort the information according to the criteria under investigation. Lastly, the shared web server enables participants to know before the meeting what the others have entered.

Reduce/Filtering: move from having many concepts to fewer concepts that meet specific criteria according to the group members.

Both at preselection time and during the meeting, participants could quickly strip down the set of conferences of interest according to the most relevant criteria.

All participants said that the filtering criteria were relevant and reflecting the group's point of view. They also all thought that the group was satisfied with the selected set of conferences.

The crucial characteristics of LIS for this aspect are those of the *navigation core* of LIS. Firstly, the features of the index propose filtering criteria. They are dynamically computed and they are relevant for the current selection of objects. Secondly, the query with its powerful logic capabilities enables participants to express sophisticated selections. Thirdly, the navigation facilities enable participants to build powerful queries, even without knowing anything about the syntax. Lastly, users do not have to worry about the consistency of the set of selected objects. The view consistency of Abilis guaranties that all conferences fulfilling the expressed query are indeed present.

This aspect is especially important. As claimed by Davis et al. [4], convergence in meetings is a slow and painful process for groups. Vogel and Coombes [16] present an experiment that supports the hypothesis that *groups selecting ideas from a multicriteria task formulation will converge better than groups working on a single criteria formulation,* where convergence is defined as *moving from many ideas to a focus on a few ideas that are worthy of further attention.* Convergence is very close to the Reduce/Filtering collaboration pattern. They also underline that *people try to minimize the effects of information overload by employing conscious or even unconscious strategies of heuristics in order to reduce information load,* where information overload is defined as *having too many things to do at once.*

With their powerful navigation facilities, LIS enable to address a large number of criteria and objects with a limited information overload. Indeed, one can concentrate on local aspects. The global consistency is maintained automatically by the concept lattice.

5 Related work

Abilis in its current stage does not pretend to match up to operational group support systems (GSS) which have a much broader scope. LIS, however, could be

integrated in some of the modules of GSS. For example, $Meetingworks^{TM}$ [13], one of the most established GSS, is a modular toolkit that can be configured to support a wide variety of group tasks. Its "Organize" module proposes a tree structure to help analyze and sort ideas. That structure looks much like the index of LIS. It can be edited by hand and some limited selection is possible. The navigation capabilities of LIS based on the concept lattice are, however, more powerful.

Concept analysis has been applied to numerous social contexts, such as social networks [15], computer-mediated communication [9] and domestic violence detection [14]. Most of those applications are intended to be applied *a posteriori*, in order to get some understanding of the studied social phenomena. On the contrary, we propose to use Logical Concept Analysis in the course and as a support of the social phenomena itself. In our case, the purpose is to support a collaborative decision process. Our approach is to other social applications, what information retrieval is to data mining. Whereas data mining automatically computes a global and static view on *a posteriori* data, information retrieval (i.e. navigation in and update of the concept lattice) presents the user with a local and dynamic view on *live* data, and only guides users in their choice.

A specificity of LIS is the use of logics. This has consequences both on the queries that can be expressed, and on the feature taxonomy. The use of logics allows to express inequalities on numerical attributes, disjunctions and negations in queries. In pure FCA, only conjunctions of Boolean attributes can be expressed. Previous sections have shown how disjunction and negation are important to express selection criteria. In the taxonomy, criteria are organized according to the logical subsumption relation between them in pure FCA, criteria would be presented as a long flat list. Logics help to make the taxonomy more concise and readable by grouping and hierarchizing together similar criteria. The taxonomy can be dynamically updated by end-users.

6 Conclusion

In this paper we have shown that a Logical Information System web server could be used to support a group decision process consisting of 1) data preparation 2) distributed individual preselection and update and 3) collaborative data exploration, update and selection. We have presented evidences that the navigation and filtering capabilities of LIS were relevant to quickly reduce the number of target conferences. Secondly, the same capabilities were also helpful to detect inconsistencies and missing knowledge. The updating capabilities of LIS enabled participants to add objects, features and links between them on the fly. As a result the group had a more complete and relevant set of information. Thirdly, the group had built a shared understanding of the relevant information.

Acknowledgments The authors thank Pierre Allard and Benjamin Sigonneau for the development and maintenance of Abilis. They thank Pierre Allard, Annie Foret and Alice Hermann for attending the experiment and giving many insightful feedbacks.

References

1. Allard, P., Ferré, S., Ridoux, O.: Discovering functional dependencies and association rules by navigating in a lattice of OLAP views. In: Kryszkiewicz, M., Obiedkov, S. (eds.) Concept Lattices and Their Applications. pp. 199–210. CEUR-WS (2010)
2. Briggs, R.O., Kolfschoten, G.L., de Vreede, G.J., Albrecht, C.C., Lukosch, S.G.: Facilitator in a box: Computer assisted collaboration engineering and process support systems for rapid development of collaborative applications for high-value tasks. In: HICSS. pp. 1–10. IEEE Computer Society (2010)
3. Codd, E., Codd, S., Salley, C.: Providing OLAP (On-line Analytical Processing) to User-Analysts: An IT Mandate. Codd & Date, Inc, San Jose (1993)
4. Davis, A., de Vreede, G.J., Briggs, R.: Designing thinklets for convergence. In: AMCIS 2007 Proceedings (2007), http://aisel.aisnet.org/amcis2007/358
5. Ducassé, M., Ferré, S.: Fair(er) and (almost) serene committee meetings with logical and formal concept analysis. In: Eklund, P., Haemmerlé, O. (eds.) Proceedings of the International Conference on Conceptual Structures. Springer-Verlag (July 2008), lecture Notes in Artificial Intelligence 5113
6. Ferré, S., Ridoux, O.: A logical generalization of formal concept analysis. In: Mineau, G., Ganter, B. (eds.) International Conference on Conceptual Structures. pp. 371–384. No. 1867 in Lecture Notes in Computer Science, Springer (Aug 2000)
7. Ferré, S., Ridoux, O.: An introduction to logical information systems. Information Processing & Management 40(3), 383–419 (2004)
8. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Heidelberg (1999)
9. Hara, N.: Analysis of computer-mediated communication: Using formal concept analysis as a visualizing methodology. Journal of Educational Computing Research 26(1), 25–49 (2002)
10. den Hengst, M., Adkins, M.: Which collaboration patterns are most challenging: A global survey of facilitators. In: HICSS. p. 17. IEEE Computer Society (2007)
11. Kilgour, D.M., Eden, C.: Handbook of Group Decision and Negotiation, Advances in Group Decision and Negotiation, vol. 4. Springer Netherlands (2010)
12. Kolfschoten, G.L., de Vreede, G.J., Briggs, R.O.: Collaboration engineering. In: Kilgour and Eden [11], chap. 20, pp. 339–357
13. Lewis, L.F.: Group support systems: Overview and guided tour. In: Kilgour and Eden [11], chap. 14, pp. 249–268
14. Poelmans, J., Elzinga, P., Viaene, S., Dedene, G.: A case of using formal concept analysis in combination with emergent self organizing maps for detecting domestic violence. In: Perner, P. (ed.) Advances in Data Mining. Applications and Theoretical Aspects, Lecture Notes in Computer Science, vol. 5633, pp. 247–260. Springer Berlin / Heidelberg (2009), http://dx.doi.org/10.1007/978-3-642-03067-3_20, 10.1007/978-3-642-03067-3_20
15. Roth, C., Bourgine, P.: Lattice-based dynamic and overlapping taxonomies: The case of epistemic communities. Scientometrics 69, 429–447 (2006), http://dx.doi.org/10.1007/s11192-006-0161-6, 10.1007/s11192-006-0161-6
16. Vogel, D., Coombes, J.: The effect of structure on convergence activities using group support systems. In: Kilgour and Eden [11], chap. 17, pp. 301–311

Comparing Performance of Algorithms for Generating the Duquenne–Guigues Basis

Konstantin Bazhanov and Sergei Obiedkov

Higher School of Economics, Moscow, Russia,
kostyabazhanov@mail.ru, sergei.obj@gmail.com

Abstract. In this paper, we take a look at algorithms involved in the computation of the Duquenne–Guigues basis of implications. The most widely used algorithm for constructing the basis is Ganter's Next Closure, designed for generating closed sets of an arbitrary closure system. We show that, for the purpose of generating the basis, the algorithm can be optimized. We compare the performance of the original algorithm and its optimized version in a series of experiments using artificially generated and real-life datasets. An important computationally expensive subroutine of the algorithm generates the closure of an attribute set with respect to a set of implications. We compare the performance of three algorithms for this task on their own, as well as in conjunction with each of the two versions of Next Closure.

1 Introduction

Implications are among the most important tools of formal concept analysis (FCA) [9]. The set of all attribute implications valid in a formal context defines a closure operator mapping attribute sets to concept intents of the context (this mapping is surjective). The following two algorithmic problems arise with respect to implications:

1. Given a set $\mathcal{L}$ of implications and an attribute set A, compute the closure $\mathcal{L}(A)$.
2. Given a formal context $\mathbb{K}$, compute a set of implications equivalent to the set of all implications valid in $\mathbb{K}$, i.e., the *cover* of valid implications.

The first of these problems has received considerable attention in the database literature in application to functional dependencies [14]. Although functional dependencies are interpreted differently than implications, the two are in many ways similar: in particular, they share the notion of semantic consequence and the syntactic inference mechanism (Armstrong rules [1]). A linear-time algorithm, LinClosure, has been proposed for computing the closure of a set with respect to a set of functional dependencies (or implications) [3], i.e., for solving the first of the two problems stated above. However, the asymptotic complexity estimates may not always be good indicators for relative performance of algorithms in practical situations. In Sect. 3, we compare LinClosure with two

 Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

other algorithms—a "naïve" algorithm, Closure [14], and the algorithm proposed in [20]—both of which are non-linear. We analyze their performance in several particular cases and compare them experimentally on several datasets.

For the second problem, an obvious choice of the cover is the Duquenne–Guigues, or canonical, basis of implications, which is the smallest set equivalent to the set of valid implications [11]. Unlike for the other frequently occurring FCA algorithmic task, the computation of all formal concepts of a formal context [12], only few algorithms have been proposed for the calculation of the canonical basis. The most widely-used algorithm was proposed by Ganter in [10]. Another, attribute-incremental, algorithm for the same problem was described in [17]. It is claimed to be much faster than Ganter's algorithm for most practical situations. The Concept Explorer software system [21] uses this algorithm to generate the Duquenne–Guigues basis of a formal context. However, we do not discuss it here, for we choose to concentrate on the computation of implications in the lectic order (see Sect. 4). The lectic order is important in the interactive knowledge-acquisition procedure of attribute exploration [8], where implications are output one by one and the user is requested to confirm or reject (by providing a counterexample) each implication.

Ganter's algorithm repeatedly computes the closure of an attribute set with respect to a set of implications; therefore, it relies heavily on a subprocedure implementing a solution to the first problem. In Sect. 4, we describe possible optimizations of Ganter's algorithm and experimentally compare the original and optimized versions in conjunction with each of the three algorithms for solving the first problem. A systematic comparison with the algorithm from [17] is left for further work.

2 The Duquenne–Guigues Basis of Implications

Before proceeding, we quickly recall the definition of the Duquenne–Guigues basis and related notions.

Given a *(formal) context* $\mathbb{K} = (G, M, I)$, where G is called a set of *objects*, M is called a set of *attributes*, and the binary relation $I \subseteq G \times M$ specifies which objects have which attributes, the derivation operators $(\cdot)^I$ are defined for $A \subseteq G$ and $B \subseteq M$ as follows:

$$A' = \{m \in M \mid \forall g \in A : gIm\} \qquad B' = \{g \in G \mid \forall m \in B : gIm\}$$

In words, A' is the set of attributes common to all objects of A and B' is the set of objects sharing all attributes of B. The double application of $(\cdot)'$ is a closure operator, i.e., $(\cdot)''$ is extensive, idempotent, and monotonous. Therefore, sets A'' and B'' are said to be *closed*. Closed object sets are called *concept extents* and closed attribute sets are called *concept intents* of the formal context $\mathbb{K}$.

In discussing the algorithms later in the paper, we assume that the sets G and M are finite.

An *implication* over M is an expression $A \to B$, where $A, B \subseteq M$ are attribute subsets. It *holds* in the context if $A' \subseteq B'$, i.e., every object of the context that has all attributes from A also has all attributes from B.

An attribute subset $X \subseteq M$ *respects* (or is a *model* of) an implication $A \to B$ if $A \not\subseteq X$ or $B \subseteq X$. Obviously, an implication holds in a context (G, M, I) if and only if $\{g\}'$ respects the implication for all $g \in G$.

A set $\mathcal{L}$ of implications over M defines the closure operator $X \mapsto \mathcal{L}(X)$ that maps $X \subseteq M$ to the smallest set respecting all the implications in $\mathcal{L}$:

$$\mathcal{L}(X) = \bigcap \{Y \mid X \subseteq Y \subseteq M, \forall (A \to B) \in \mathcal{L} : A \not\subseteq Y \text{ or } B \subseteq Y\}.$$

We discuss algorithms for computing $\mathcal{L}(X)$ in Sect. 3. Note that, if $\mathcal{L}$ is the set of all valid implications of a formal context, then $\mathcal{L}(X) = X''$ for all $X \subseteq M$.

Two implication sets over M are *equivalent* if they are respected by exactly the same subsets of M. Equivalent implication sets define the same closure operator. A *minimum cover* of an implication set $\mathcal{L}$ is a set of minimal size among all implication sets equivalent to $\mathcal{L}$. One particular minimum cover described in [11] is defined using the notion of a pseudo-closed set, which we introduce next.

A set $P \subseteq M$ is called *pseudo-closed* (with respect to a closure operator $(\cdot)''$) if $P \neq P''$ and $Q'' \subset P$ for every pseudo-closed $Q \subset P$.

In particular, all minimal non-closed sets are pseudo-closed. A pseudo-closed attribute set of a formal context is also called a *pseudo-intent*.

The *Duquenne–Guigues* or *canonical basis* of implications (with respect to a closure operator $(\cdot)''$) is the set of all implications of the form $P \to P''$, where P is pseudo-closed. This set of implications is of minimal size among those defining the closure operator $(\cdot)''$. If $(\cdot)''$ is the closure operator associated with a formal context, the Duquenne–Guigues basis is a minimum cover of valid implications of this context. The computation of the Duquenne–Guigues basis of a formal context is hard, since even recognizing pseudo-intents is a coNP-complete problem [2], see also [13, 7]. We discuss algorithms for computing the basis in Sect. 4.

3 Computing the Closure of an Attribute Set

In this section, we compare the performance of algorithms computing the closure of an attribute set X with respect to a set $\mathcal{L}$ of implications. Algorithm 1 [14] checks every implication $A \to B \in \mathcal{L}$ and enlarges X with attributes from B if $A \subseteq X$. The algorithm terminates when a fixed point is reached, that is, when the set X cannot be enlarged any further (which always happens at some moment, since both $\mathcal{L}$ and M are assumed finite).

The algorithm is obviously quadratic in the number of implications in $\mathcal{L}$ in the worst case. The worst case happens when exactly one implication is applied at each iteration (but the last one) of the **repeat** loop, resulting in $|\mathcal{L}|(|\mathcal{L}|+1)/2$ iterations of the **for all** loop, each requiring $O(|M|)$ time.

Example 1. A simple example is when $X = \{1\}$ and the implications in $\mathcal{L} = \{\{i\} \to \{i+1\} \mid i \in \mathbb{N}, 0 < i < n\}$ for some n are arranged in the descending order of their one-element premises.

Algorithm 1 CLOSURE$(X, \mathcal{L})$

Input: An attribute set $X \subseteq M$ and a set $\mathcal{L}$ of implications over M.
Output: The closure of X w.r.t. implications in $\mathcal{L}$.

```
repeat
    stable := true
    for all A → B ∈ L do
        if A ⊆ X then
            X := X ∪ B
            stable := false
            L := L \ {A → B}
until stable
return X
```

In [3], a linear-time algorithm, LINCLOSURE, is proposed for the same problem. Algorithm 2 is identical to the version of LINCLOSURE from [14] except for one modification designed to allow implications with empty premises in $\mathcal{L}$. LINCLOSURE associates a counter with each implication initializing it with the size of the implication premise. Also, each attribute is linked to a list of implications that have it in their premises. The algorithm then checks every attribute m of X (the set whose closure must be computed) and decrements the counters for all implications linked to m. If the counter of some implication $A \to B$ reaches zero, attributes from B are added to X. Afterwards, they are used to decrement counters along with the original attributes of X. When all attributes in X have been checked in this way, the algorithm stops with X containing the closure of the input attribute set.

It can be shown that the algorithm is linear in the length of the input assuming that each attribute in the premise or conclusion of any implication in $\mathcal{L}$ requires a constant amount of memory [14].

Example 2. The worst case for LINCLOSURE occurs, for instance, when $X \subset \mathbb{N}$, $M = X \cup \{1, 2, \ldots, n\}$ for some n such that $X \cap \{1, 2, \ldots, n\} = \varnothing$ and $\mathcal{L}$ consists of implications of the form

$$X \cup \{i \mid 0 < i < k\} \to \{k\}$$

for all k such that $1 \leq k \leq n$. During each of the first $|X|$ iterations of the **for all** loop, the counters of all implications will have to be updated with only the last iteration adding one attribute to X using the implication $X \to \{1\}$. At each of the subsequent $n - 1$ iterations, the counter for every so far "unused" implication will be updated and one attribute will be added to X. The next, $(|X| + n)$th, iteration will terminate the algorithm.

Note that, if the implications in $\mathcal{L}$ are arranged in the superset-inclusion order of their premises, this example will present the worst case for Algorithm 1 requiring n iterations of the main loop. However, if the implications are arranged in the subset-inclusion order of their premises, one iteration will be sufficient.

Inspired by the mechanism used in LINCLOSURE to obtain linear asymptotic complexity, but somewhat disappointed by the poor performance of the

Algorithm 2 LinClosure$(X, \mathcal{L})$

Input: An attribute set $X \subseteq M$ and a set $\mathcal{L}$ of implications over M.
Output: The closure of X w.r.t. implications in $\mathcal{L}$.

```
for all A → B ∈ L do
    count[A → B] := |A|
    if |A| = 0 then
        X := X ∪ B
    for all a ∈ A do
        add A → B to list[a]
update := X
while update ≠ ∅ do
    choose m ∈ update
    update := update \ {m}
    for all A → B ∈ list[m] do
        count[A → B] = count[A → B] − 1
        if count[A → B] = 0 then
            add := B \ X
            X := X ∪ add
            update := update ∪ add
return X
```

algorithm relative to Closure, which was revealed in his experiments, Wild proposed a new algorithm in [20]. We present this algorithm (in a slightly more compact form) as Algorithm 3. The idea is to maintain implication lists similar to those used in LinClosure, but get rid of the counters. Instead, at each step, the algorithm combines the implications in the lists associated with attributes not occurring in X and "fires" the remaining implications (i.e., uses them to enlarge X). When there is no implication to fire, the algorithm terminates with X containing the desired result.

Wild claims that his algorithm is faster than both LinClosure and Closure, even though it has the same asymptotic complexity as the latter. The worst case for Algorithm 3 is when $\mathcal{L} \setminus \mathcal{L}_1$ contains exactly one implication $A \to B$ and $B \setminus X$ contains exactly one attribute at each iteration of the **repeat ... until** loop. Example 1 presents the worst case for 3, but, unlike for Closure, the order of implications in $\mathcal{L}$ is irrelevant. The worst case for LinClosure (see Example 2) is also the worst case for Algorithm 3, but it deals with it, perhaps, in a more efficient way using n iterations of the main loop compared to $n + |X|$ iterations of the main loop in LinClosure.

Experimental Comparison

We implemented the algorithms in C++ using Microsoft Visual Studio 2010. For the implementation of attribute sets, as well as sets of implications in Algorithm 3, we used dynamic bit sets from the Boost library [6]. All the tests described in the following sections were carried out on an Intel Core i5 2.67 GHz computer with 4 Gb of memory running under Windows 7 Home Premium x64.

Algorithm 3 WILD'S CLOSURE$(X, \mathcal{L})$

Input: An attribute set $X \subseteq M$ and a set $\mathcal{L}$ of implications over M.
Output: The closure of X w.r.t. implications in $\mathcal{L}$.

```
for all m ∈ M do
    for all A → B ∈ L do
        if m ∈ A then
            add A → B to list[m]
repeat
    stable := true
    L1 := ⋃_{m∈M\X} list[m]
    for all A → B ∈ L \ L1 do
        X := X ∪ B
        stable := false
    L := L1
until stable
return X
```

Figure 1 shows the performance of the three algorithms on Example 1. Algorithm 2 is the fastest algorithm in this case: for a given n, it needs n iterations of the outer loop—the same as the other two algorithms, but the inner loop of Algorithm 2 checks exactly one implication at each iteration, whereas the inner loop of Algorithm 1 checks $n - i$ implications at the ith iteration. Although the inner loop of Algorithm 3 checks only one implication at the ith iteration, it has to compute the union of $n - i$ lists in addition.

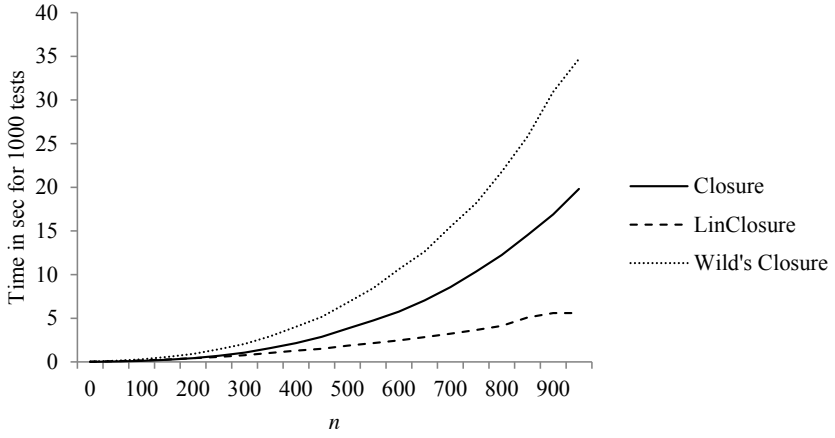


Fig. 1. The performance of Algorithms 1–3 for Example 1.

Figure 2 shows the performance of the algorithms on Example 2. Here, the behavior of Algorithm 2 is similar to that of Algorithm 1, but Algorithm 2 takes more time due to the complicated initialization step.

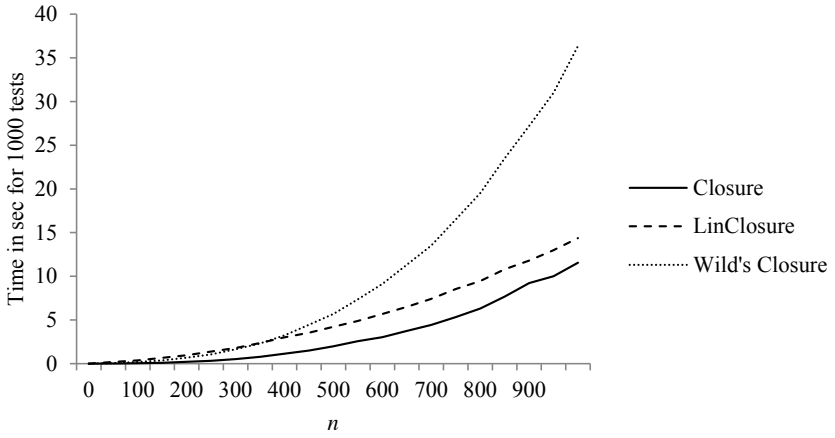


Fig. 2. The performance of Algorithms 1–3 for Example 2 with implications in $\mathcal{L}$ arranged in the superset-inclusion order of their premises and $|X| = 50$.

Interestingly, Algorithm 1 works amost twice as fast on Example 2 as it does on Example 1. This may seem surprising, since it is easy to see that the algorithm performs essentially the same computations in both cases, the difference being that the implications of Example 1 have single-element premises. However, this turns out to be a source of inefficiency: at each iteration of the main loop, all implications but the last fail to fire, but, for each of them, the algorithm checks if their premises are included in the set X. Generally, when $A \not\subseteq X$, this can be established easier if A is large, for, in this case, A is likely to contain more elements outside X. This effect is reinforced by the implementation of sets as bit strings: roughly speaking, to verify that $\{i\} \not\subseteq \{1\}$, it is necessary to check all bits up $\{i\}$, whereas $\{i \mid 0 < i < k\} \not\subseteq \{k+1\}$ can be established by checking only one bit (assuming that bits are checked from left to right). Alternative data structures for set implementation might have less dramatic consequences for performance in this setting. On the other hand, the example shows that performance may be affected by issues not so obviously related to the structure of the algorithm, thus, suggesting additional paths to obtain an optimal behavior (e.g., by rearranging attributes or otherwise preprocessing the input data).

We have experimented with computing closures using the Duquenne–Guigues bases of formal contexts as input implication sets. Table 1 shows the results for randomly generated contexts. The first two columns indicate the size of the attribute set and the number of implications, respectively. The remaining three columns record the time (in seconds) for computing the closures of 1000 ran-

domly generated subsets of M by each of the three algorithms. Table 3 presents similar results for datasets taken from the UCI repository [5] and, if necessary, transformed into formal contexts using FCA scaling [9].[1] The contexts are described in Table 2, where the last four columns correspond to the number of objects, number of attributes, number of intents, and number of pseudo-intents (i.e., the size of the canonical basis) of the context named in the first column.

Table 1. Performance on randomly generated tests (time in seconds per 1000 closures)

		Algorithm		
$\|M\|$	$\|\mathcal{L}\|$	1	2	3
30	557	0.0051	0.2593	0.0590
50	1115	0.0118	0.5926	0.1502
100	380	0.0055	0.2887	0.0900
100	546	0.0086	0.4229	0.1350
100	2269	0.0334	1.5742	0.5023
100	3893	0.0562	2.6186	0.8380
100	7994	0.1134	5.3768	1.7152
100	8136	0.1159	5.6611	1.8412

Table 2. Contexts obtained from UCI datasets

Context	$\|G\|$	$\|M\|$	# intents	# pseudo-intents
Zoo	101	28	379	141
Postoperative Patient	90	26	2378	619
Congressional Voting	435	18	10644	849
SPECT	267	23	21550	2169
Breast Cancer	286	43	9918	3354
Solar Flare	1389	49	28742	3382
Wisconsin Breast Cancer	699	91	9824	10666

In these experiments, Algorithm 1 was the fastest and Algorithm 2 was the slowest, even though it has the best asymptotic complexity. This can be partly explained by the large overhead of the initialization step (setting up counters and implication lists). Therefore, these results can be used as a reference only when the task is to compute one closure for a given set of implications. When

[1] The breast cancer domain was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia (now, Slovenia). Thanks go to M. Zwitter and M. Soklic for providing the data.

a large number of closures must be computed with respect to the same set of implications, Algorithms 2 and 3 may be more appropriate.

Table 3. Performance on the canonical bases of contexts from Table 2 (time in seconds per 1000 closures)

Context	Algorithm 1	2	3
Zoo	0.0036	0.0905	0.0182
Postoperative Patient	0.0054	0.2980	0.0722
Congressional Voting	0.0075	0.1505	0.0883
SPECT	0.0251	0.9848	0.2570
Breast Cancer	0.0361	1.7912	0.5028
Solar Flare	0.0370	2.1165	0.6317
Wisconsin Breast Cancer	0.1368	8.4984	2.4730

4 Computing the Basis in the Lectic Order

The best-known algorithm for computing the Duquenne–Guigues basis was developed by Ganter in [10]. The algorithm is based on the fact that intents and pseudo-intents of a context taken together form a closure system. This makes it possible to iteratively generate all intents and pseudo-intents using Next Closure (see Algorithm 4), a generic algorithm for enumerating closed sets of an arbitrary closure operator (also proposed in [10]). For every generated pseudo-intent P, an implication $P \to P''$ is added to the basis. The intents, which are also generated, are simply discarded.

Algorithm 4 Next Closure(A, M, $\mathcal{L}$)

Input: A closure operator $X \mapsto \mathcal{L}(X)$ on M and a subset $A \subseteq M$.
Output: The lectically next closed set after A.

```
for all m ∈ M in reverse order do
    if m ∈ A then
        A := A \ {m}
    else
        B := L(A ∪ {m})
        if B \ A contains no element < m then
            return B
return ⊥
```

Next Closure takes a closed set as input and outputs the next closed set according to a particular *lectic* order, which is a linear extension of the subset-

inclusion order. Assuming a linear order $<$ on attributes in M, we say that a set $A \subseteq M$ is *lectically smaller* than a set $B \subseteq M$ if

$$\exists b \in B \setminus A \; \forall a \in A (a < b \Rightarrow a \in B).$$

In other words, the lectically largest among two sets is the one containing the smallest element in which they differ.

Example 3. Let $M = \{a < b < c < d < e < f\}$, $A = \{a, c, e\}$ and $B = \{a, b, f\}$. Then, A is lectically smaller than B, since the first attribute in which they differ, b, is in B. Note that if we represent sets by bit strings with smaller attributes corresponding to higher-order bits (in our example, $A = 101010$ and $B = 110001$), the lectic order will match the usual less-than order on binary numbers.

To be able to use Next Closure for iterating over intents and pseudo-intents, we need access to the corresponding closure operator. This operator, which we denote by $^\bullet$, is defined via the Duquenne–Guigues basis $\mathcal{L}$ as follows.[2] For a subset $A \subseteq M$, put

$$A^+ = A \cup \bigcup \{P'' \mid P \to P'' \in \mathcal{L}, P \subset A\}.$$

Then, $A^\bullet = A^{+ + \cdots +}$, where $A^{\bullet +} = A^\bullet$; i.e., $^\bullet$ is the transitive closure of $^+$.

The problem is that $\mathcal{L}$ is not available when we start; in fact, this is precisely what we want to generate. Fortunately, for computing a pseudo-closed set A, it is sufficient to know only implications with premises that are proper subsets of A. Generating pseudo-closed sets in the lectic order, which is compatible with the subset-inclusion order, we ensure that, at each step, we have at hand the required part of the basis. Therefore, we can use any of the three algorithms from Sect. 3 to compute $A^\bullet$ (provided that the implication $A^\bullet \to A''$ has not been added to $\mathcal{L}$ yet). Algorithm 5 uses Next Closure to generate the canonical basis. It passes Next Closure the part of the basis computed so far; Next Closure may call any of the Algorithms 1–3 to compute the closure, $\mathcal{L}(A \cup \{m\})$, with respect to this set of implications.

After Next Closure computes $A^\bullet$, the implication $A^\bullet \to A''$ may be added to the basis. Algorithm 5 will then pass $A^\bullet$ as the input to Next Closure, but there is some room for optimizations here. Let i be the maximal element of A and j be the minimal element of $A'' \setminus A$. Consider the following two cases:

$j < i$: As long as $m > i$, the set $\mathcal{L}(A^\bullet \cup \{m\})$ will be rejected by Next Closure, since it will contain j. Hence, it makes sense to skip all $m > i$ and continue as if $A^\bullet$ had been rejected by Next Closure. This optimization has already been proposed in [17].

$i < j$: It can be shown that, in this case, the lectically next intent or pseudo-intent after $A^\bullet$ is A''. Hence, A'' could be used at the next step instead of $A^\bullet$.

Algorithm 6 takes these considerations into account.

[2] We deliberately use the same letter $\mathcal{L}$ for an implication set and the closure operator it defines.

Algorithm 5 CANONICAL BASIS(M, $''$)

Input: A closure operator $X \mapsto X''$ on M, e.g., given by a formal context (G, M, I).
Output: The canonical basis for the closure operator.

```
L := ∅
A := ∅
while A ≠ M do
    if A ≠ A'' then
        L := L ∪ {A → A''}
    A := NEXT CLOSURE(A, M, L)
return L
```

Algorithm 6 CANONICAL BASIS(M, $''$), an optimized version

Input: A closure operator $X \mapsto X''$ on M, e.g., given by a formal context (G, M, I).
Output: The canonical basis for the closure operator.

```
L := ∅
A := ∅
i := the smallest element of M
while A ≠ M do
    if A ≠ A'' then
        L := L ∪ {A → A''}
    if A'' \ A contains no element < i then
        A := A''
        i := the largest element of M
    else
        A := {m ∈ A | m ≤ i}
    for all j ≤ i ∈ M in reverse order do
        if j ∈ A then
            A := A \ {j}
        else
            B := L(A ∪ {j})
            if B \ A contains no element < j then
                A := B
                i := j
                exit for
return L
```

Experimental Comparison

We used Algorithms 5 and 6 for constructing the canonical bases of the contexts involved in testing the performance of the algorithms from Sect. 3, as well as the context $(M, M, \neq)$ with $|M| = 18$, which is special in that every subset of M is closed (and hence there are no valid implications). Both algorithms have been tested in conjunction with each of the three procedures for computing closures (Algorithm 1–3). The results are presented in Table 4 and Fig. 3. It can be seen that Algorithm 6 indeed improves on the performance of Algorithm 5. Among the three algorithms computing the closure, the simpler Algorithm 1 is generally more efficient, even though, in our implementation, we do not perform the initialization step of Algorithms 2 and 3 from scratch each time we need to compute a closure of a new set; instead, we reuse the previously constructed counters and implication lists and update them incrementally with the addition of each new implication. We prefer to treat these results as preliminary: it still remains to see whether the asymptotic behavior of LinClosure will give it an advantage over the other algorithms on larger contexts.

Table 4. Time (in seconds) for building the canonical bases of artificial contexts

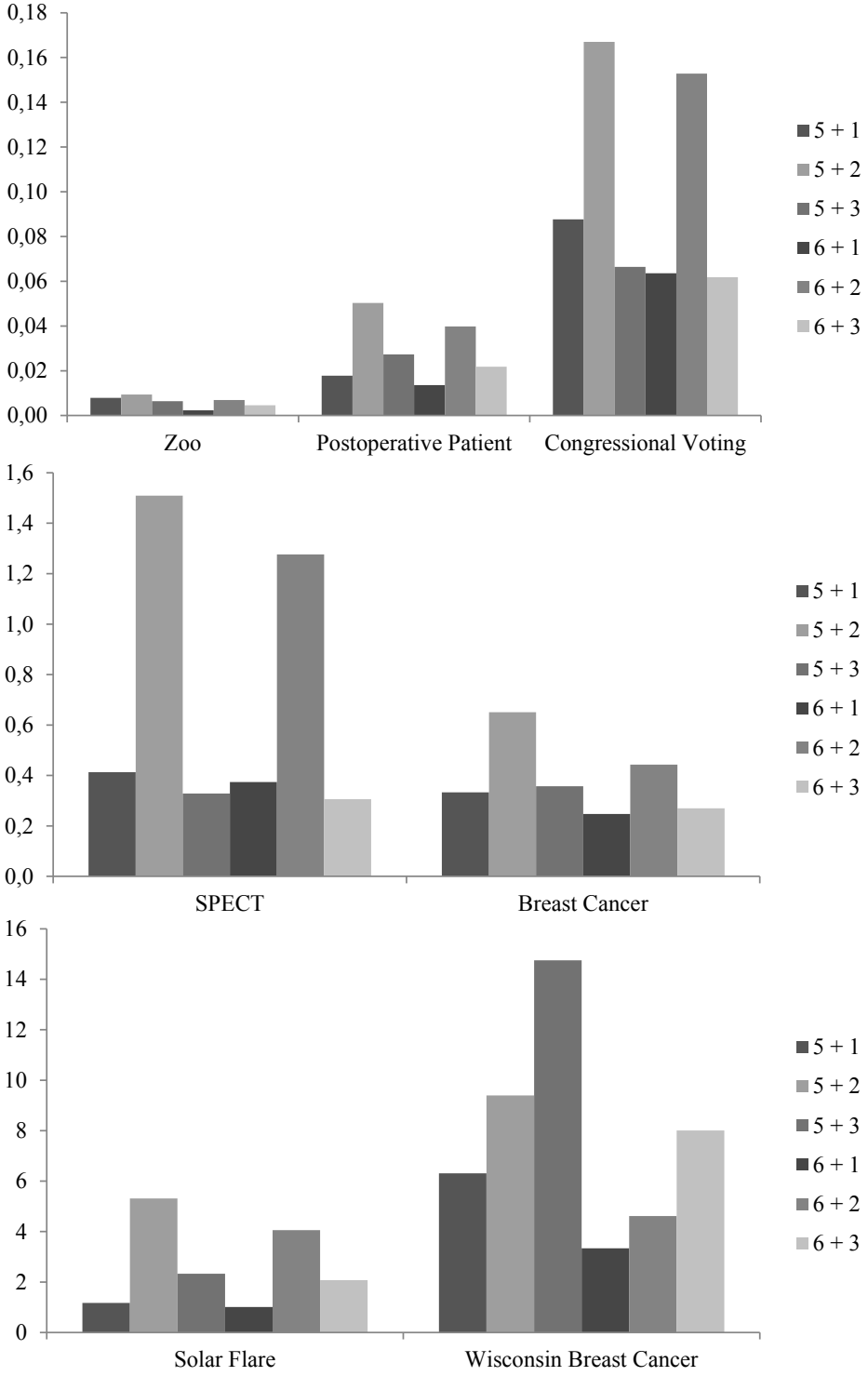
Context	# intents	# pseudo-intents	Algorithm 5 + 1	5 + 2	5 + 3	6 + 1	6 + 2	6 + 3
$100 \times 30, 4$	307	557	0.0088	0.0145	0.0119	0.0044	0.0065	0.0059
$10 \times 100, 25$	129	380	0.0330	0.0365	0.0431	0.0073	0.0150	0.0169
$100 \times 50, 4$	251	1115	0.0442	0.0549	0.0617	0.0138	0.0152	0.0176
$10 \times 100, 50$	559	546	0.0542	0.1312	0.1506	0.0382	0.0932	0.0954
$20 \times 100, 25$	716	2269	0.3814	0.3920	0.7380	0.1219	0.1312	0.2504
$50 \times 100, 10$	420	3893	1.1354	0.7291	1.6456	0.1640	0.1003	0.2299
$900 \times 100, 4$	2472	7994	4.6313	2.7893	6.3140	1.5594	0.8980	2.0503
$20 \times 100, 50$	12394	8136	7.3097	8.1432	14.955	5.1091	6.0182	10.867
$(M, M, \neq)$	262144	0	0.1578	0.3698	0.1936	0.1333	0.2717	0.1656

5 Conclusion

In this paper, we compared the performance of several algorithms computing the closure of an attribute set with respect to a set of implications. Each of these algorithms can be used as a (frequently called) subroutine while computing the Duquenne–Guigues basis of a formal context. We tested them in conjunction with Ganter's algorithm and its optimized version.

In our future work, we plan to extend the comparison to algorithms generating the Duquenne–Guigues basis in a different (non-lectic) order, in particular, to incremental [17] and divide-and-conquer [19] approaches, probably, in conjunction with newer algorithms for computing the closure of a set [16]. In addition,

Fig. 3. Time (in seconds) for building the canonical bases of contexts from Table 2



we are going to consider algorithms that generate other implication covers: for example, direct basis [15, 20, 4] or proper basis [18]. They can be used as an intermediate step in the computation of the Duquenne–Guigues basis. If the number of intents is much larger than the number of pseudo-intents, this two-step approach may be more efficient than direct generation of the Duquenne–Guigues basis with Algorithms 5 or 6, which produce all intents as a side effect.

Acknowledgements

The second author was supported by the Academic Fund Program of the Higher School of Economics (project 10-04-0017) and the Russian Foundation for Basic Research (grant no. 08-07-92497-NTsNIL_a).

References

1. Armstrong, W.: Dependency structure of data base ralationship. Proc. IFIP Congress pp. 580–583 (1974)
2. Babin, M.A., Kuznetsov, S.O.: Recognizing pseudo-intents is coNP-complete. In: Kryszkiewicz, M., Obiedkov, S. (eds.) Proceedings of the 7th International Conference on Concept Lattices and Their Applications. pp. 294–301. University of Sevilla, Spain (2010)
3. Beeri, C., Bernstein, P.: Computational problems related to the design of normal form relational schemas. ACM TODS 4(1), 30–59 (March 1979)
4. Bertet, K., Monjardet, B.: The multiple facets of the canonical direct unit implicational basis. Theor. Comput. Sci. 411(22-24), 2155–2166 (2010)
5. Blake, C., Merz, C.: UCI repository of machine learning databases (1998), http://archive.ics.uci.edu/ml
6. Demming, R., Duffy, D.: Introduction to the Boost C++ Libraries. Datasim Education Bv (2010), see http://www.boost.org
7. Distel, F., Sertkaya, B.: On the complexity of enumerating pseudo-intents. Discrete Appl. Math. 159, 450–466 (March 2011)
8. Ganter, B.: Attribute exploration with background knowledge. Theor. Comput. Sci. pp. 215–233 (1999)
9. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin (1999)
10. Ganter, B.: Two basic algorithms in concept analysis. Preprint 831, Technische Hochschule Darmstadt, Germany (1984)
11. Guigues, J.L., Duquenne, V.: Familles minimales d'implications informatives resultant d'un tableau de donnees binaires. Math. Sci. Hum. 95(1), 5–18 (1986)
12. Kuznetsov, S., Obiedkov, S.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental and Theoretical Artificial Intelligence 14(2/3), 189–216 (2002)
13. Kuznetsov, S.O., Obiedkov, S.: Some decision and counting problems of the Duquenne–Guigues basis of implications. Discrete Appl. Math. 156(11), 1994–2003 (2008)
14. Maier, D.: The theory of relational databases. Computer software engineering series, Computer Science Press (1983)

15. Mannila, H., Räihä, K.J.: The design of relational databases. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1992)
16. Mora, A., Aguilera, G., Enciso, M., Cordero, P., de Guzman, I.P.: A new closure algorithm based in logic: SLFD-Closure versus classical closures. Inteligencia Artificial, Revista Iberoamericana de IA 10(31), 31–40 (2006)
17. Obiedkov, S., Duquenne, V.: Attribute-incremental construction of the canonical implication basis. Annals of Mathematics and Artificial Intelligence 49(1-4), 77–99 (April 2007)
18. Taouil, R., Bastide, Y.: Computing proper implications. In Proc. ICCS-2001 International Workshop on Concept Lattices-Based Theory, Methods and Tools for Knowledge Discovery in Databases pp. 290–303 (2001)
19. Valtchev, P., Duquenne, V.: On the merge of factor canonical bases. In: Medina, R., Obiedkov, S. (eds.) ICFCA. Lecture Notes in Computer Science, vol. 4933, pp. 182–198. Springer (2008)
20. Wild, M.: Computations with finite closure systems and implications. In: Computing and Combinatorics. pp. 111–120 (1995)
21. Yevtushenko, S.A.: System of data analysis "Concept Explorer" (in Russian). In: Proceedings of the 7th national conference on Artificial Intelligence KII-2000. pp. 127–134. Russia (2000), http://conexp.sourceforge.net/

Filtering Machine Translation Results with Automatically Constructed Concept Lattices

Yılmaz Kılıçaslan[1] and Edip Serdar Güner[1],

[1] Trakya University, Department of Computer Engineering,
22100 Edirne, Turkey
{yilmazk, eserdarguner}@trakya.edu.tr

Abstract. Concept lattices can significantly improve machine translation systems when applied as filters to their results. We have developed a rule-based machine translator from Turkish to English in a unification-based programming paradigm and supplemented it with an automatically constructed concept lattice. The test results achieved by applying this translation system to a Turkish child story reveals that lattices used as filters to translation results have a promising potential to improve machine translation. We have compared our system with Google Translate on the data. The comparison suggests that a rule-based system can even compete with this statistical machine translation system that stands out with its wide range of users.

Keywords: Concept Lattices, Rule-based Machine Translation, Evaluation of MT systems.

1 Introduction

Paradigms of Machine translation (MT) can be classified into two major categories depending on their focus: result-oriented paradigms and process-oriented ones. Statistical MT focuses on the result of the translation, not the translation process itself. In this paradigm, translations are generated on the basis of statistical models whose parameters are derived from the analysis of bilingual text corpora. Rule-based MT, a more classical paradigm, focuses on the selection of representations to be used and steps to be performed during the translation process.

It is the rule-based paradigm that will be the concern of this paper. We argue for the viability of a rule-based translation model where a concept lattice functions as a filter for its results.

In what follows, we first introduce the classical models for doing rule-based MT, illustrating particular problematic cases with translation pairs between Turkish and English (cf. Section 2). Then, we briefly introduce the basic notions of Formal Concept Analysis (FCA) and touch upon the question of how lattices built using FCA can serve as a bridge between two languages (cf. Section 3). This is followed by the presentation of our translation system (cf. Section 4). Subsequently, we report on and evaluate several experiments which we have performed by feeding our translation system with a Turkish child story text (cf. Section 5). The discussion ends with some remarks and with a summary of the paper (cf. Section 6).

2 Models for Rule-Based Translation

2.1 Direct Translation

The most straightforward MT strategy is the so-called direct translation. Basically, the strategy is to translate each word into its target language counterpart while proceeding word-by-word through the source language text or speech. If the only difference between two languages were due to their lexical choices, this approach could be a very easy way of producing high quality translation. However, languages differ from each other not only lexically but also structurally.

In fact, the direct translation strategy works very well only for very simple cases like the following:

(1) Turkish: ⇨ Direct Translation to English:

Köpek-ler havlar-lar. ⇨ Dogs bark.

dog-pl *bark-3pl*

In this example, the direct translation strategy provides us with a perfect translation of the Turkish sentence (interpreted as a kind-level statement about dogs). But, consider now the following example:

(2) Turkish: Direct Translation to English:

Kadın onu tanı-yor. ⇨ Woman {*him*/*her*/*it*} {is/are} knowing.

woman {*him*/*her*/*it*} *know-cont-3*

Supposing that the referent of the pronoun is a male person, the expected translation for the given Turkish sentence would be the following:

(3) Correct Translation:

The woman knows him.

The direct translation approach fails in this example in the following respects: First, the translation results in a subject-object-verb (SOV) ordering, which does not comply with the canonical SVO ordering in English. SOV is the basic word order in Turkish. Second, the subject does not have the required definite article in the translation. The reason for this is another typological difference between the two languages: Turkish lacks a definite article. Third, the word-by-word translation leaves the English auxiliary verb ambiguous with respect to number, as the Turkish verb does not carry the number information. Fourth, the verb *know* is encoded in the progressive aspect in the translation, which is unacceptable as it denotes a mental state. This anomaly is the result of directly translating the Turkish continuous suffix *–yor* to the English suffix *–ing*. Fifth, the pronoun is left ambiguous with respect to gender in the translation, as Turkish pronouns do not bear this information.

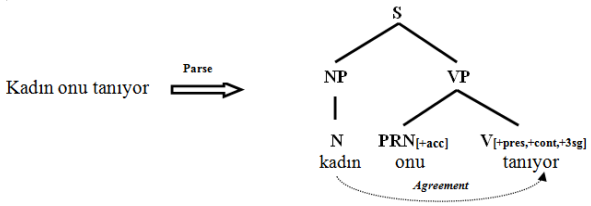
2.2 Transfer Approach

2.2.1 Syntactic Transfer

As Jurafsky and Martin [6] point out, examples like those above suggest that the direct approach to MT is too focused on individual words and that we need to add phrasal and structural knowledge into our MT models to achieve better results. It is through the transfer approach that a rule-based strategy incorporates the structural knowledge into the MT model. In this approach, MT involves three phases: *analysis*, *transfer*, and *generation*. In the analysis phase, the source language text is parsed into a syntactic and/or semantic structure. In the transfer phase, the structure of the source language is transformed to a structure of the target language. The generation phase takes this latter structure as input and turns it to an actual text of the target language.

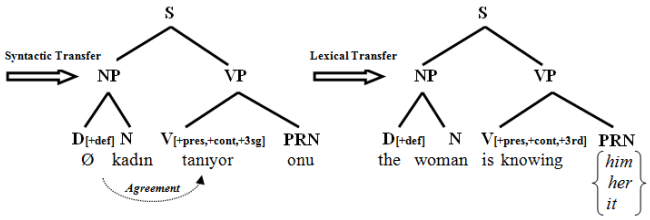
Let us first see how the transfer technique can make use of syntactic knowledge to improve the translation result of the example discussed above. Assuming a simple syntactic paradigm, the input sentence can be parsed into the following structure:

(4)



Once the sentence has been parsed, the resulting tree will undergo a *syntactic transfer* operation to resemble the target parse tree and this will be followed by a lexical transfer operation to generate the target text:

(5)



The syntactic transfer exploits the following facts about English: a singular count noun must have a determiner and the subject agrees in number and person with the verb. Collecting the leaves of the target parse tree, we get the following output:

(6) Translation via Syntactic Transfer:

The woman is knowing {him / her / it}.

This output is free from the first three defects noted with the direct translation. However, the problem of encoding the mental state verb in progressive aspect and the

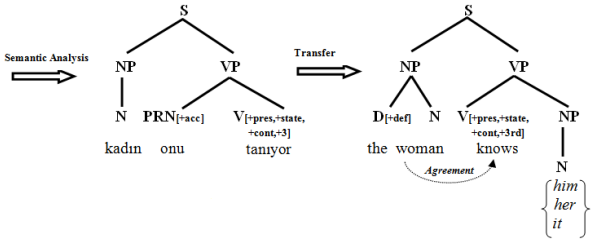
gender ambiguity of the pronoun still await to be resolved. These require *meaning-related knowledge* to be incorporated into the MT model.

2.2.2 Semantic Transfer

The *context-independent* aspect of meaning is called *semantic meaning*. A crucial component of the semantic meaning of a natural language sentence is its *lexical aspect*, which determines whether the situation that the sentence describes is a (*punctual*) *event*, a *process* or a *state*. This information is argued to be inherently encoded in the verb. Obviously, knowing is a mental state and, hence, cannot be realized in the progressive aspect.

We can apply a shallow semantic analysis to our previously obtained syntactic structure, which will give us a tree structure enriched with aspectual information, and thereby achieve a more satisfactory transfer:

(7)



The resulting translation is the following:

(8) Translation via Semantic Transfer:

The woman knows $\left\{\begin{matrix} him \\ her \\ it \end{matrix}\right\}$.

2.3 Interlingua Approach

There are two problems with the transfer model: it requires contrastive knowledge about languages and it requires such knowledge for every pair of languages. If the meaning of the input can be extracted and encoded in a language-independent form and the output can, in turn, be generated out of this form, there will be no need for any kind of contrastive knowledge. A language-independent meaning representation language to be used in such a scheme is usually referred to as an *interlingua*.

A common way to visualize the three approaches to rule-based MT is with *Vauquois triangle* shown below (adopted from [6]):

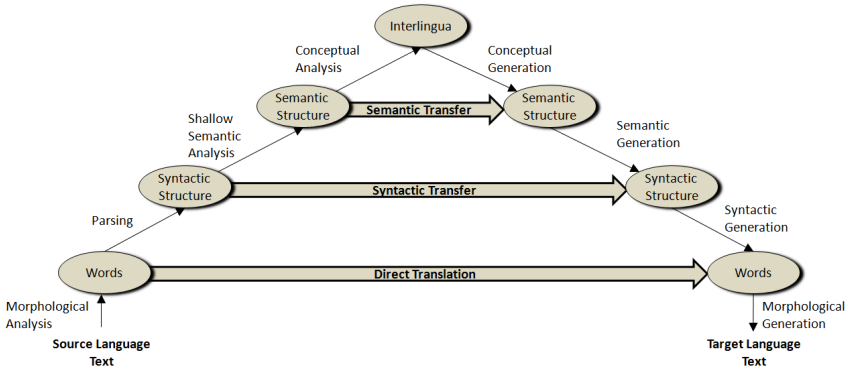


Fig. 1. The Vauquois triangle.

As Jurafsky and Martin point out:

> [t]he triangle shows the increasing depth of analysis required (on both the analysis and generation end) as we move from the direct approach through transfer approaches, to interlingual approaches. In addition, it shows the decreasing amount of transfer knowledge needed as we move up the triangle, from huge amounts of transfer at the direct level (almost all knowledge is transfer knowledge for each word) through transfer (transfer rules only for parse trees or thematic roles) through interlingua (no specific transfer knowledge). (p. 867)

3 Lattice-Based Interlingua Strategy

A question left open above is that of what kind of representation scheme can be used as an interlingua. There are many possible alternatives such as predicate calculus, Minimal Recursion Semantics or an event-based representation. Another interesting possibility is to use lattices built using Formal Concept Analysis (FCA) as meaning representations to this effect.

FCA, developed by Ganter & Wille [5], assumes that data from an application are given by a formal context, a triple (G, M, I) consisting of two sets G and M and a so called incidence relation I between these sets. The elements of G are called the objects and the elements of M are called the attributes. The relation I holds between g and m, $(g, m) \in I$ if and only if the object g has the attribute m. A formal context induces two operators, both of which usually denoted by $'$. One of these operators maps each set of objects A to the set of attributes A' which these objects have in common. The other operator maps each set of attributes B to the set of objects B' which satisfies these attributes. FCA is in fact an attempt to give a formal definition of the notion of a 'concept'. A formal concept of the context (G, M, I) is a pair (A, B) such that $G \supseteq A = A'$ and $M \supseteq B = B'$. A is called the *extent* and B the *intent* of the concept (A, B). The set of all concepts of the context (G, M, I) is denoted by C(G, M, I). This set is ordered by a subconcept – superconcept relation, which is a partial order relation denoted by $\leq$. If (A_1, B_1) and (A_2, B_2) are concepts in C(G, M, I), the former is said to

be a subconcept of the latter (or, the latter a superconcept of the former), i.e., $(A_1, B_1) \leq (A_2, B_2)$, if and only if $A_1 \subseteq A_2$ (which is equivalent to $B_1 \supseteq B_2$). The ordered set $C(G, M, I; \leq)$ is called the concept lattice or (Galois lattice) of the context (G, M, I). A concept lattice can be drawn as a (Hasse) diagram in which concepts are represented by nodes interconnected by lines going down from superconcept nodes to subconcept ones.

Priss [15], rewording an idea first mentioned by Kipke & Wille [8], suggests that once linguistic databases are formalized as concept lattices, the lattices can serve as an interlingua. She explains how a concept lattice can serve as a bridge between two languages with the aid of the figure below (taken from [13]):

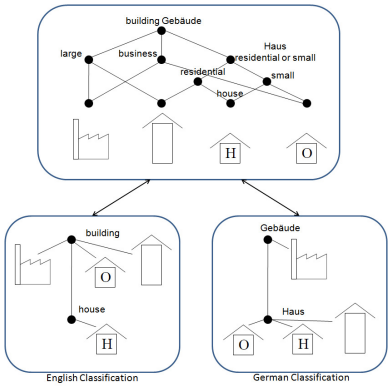


Fig. 2. – A concept lattice as an interlingua.

> [This figure] shows separate concept lattices for English and German words for "building". The main difference between English and German is that in English "house" only applies to small residential buildings (denoted by letter "H"), whereas in German even small office buildings (denoted by letter "O") and larger residential buildings can be called "Haus". Only factories would not normally be called "Haus" in German. The lattice in the top of the figure constitutes an information channel in the sense of Barwise & Seligman [2] between the German and the English concept lattice. ([15] p. 158)

We consider Priss's approach a promising avenue for interlingua-based translation strategies. We suggest that this approach can work not only for isolated words but also even for text fragments. In what follows, we will sketch out a strategy with interlingual concept lattices serving as filters for refining translation results. The strategy proceeds as follows: 1) Compile a concept lattice from a data source like WordNet. 2) Link the nodes of the lattice to their possibly corresponding expressions in the source and target language. 3) Translate the input text into the target language with no consideration of the pragmatic aspects of its meaning. 4) Integrate the concepts derived from the input text into the concept lattice. The main motivation behind this strategy is to refine the translation results to a certain extent by means of pragmatic knowledge structured as formal contexts.

4 A Translation System with Interlingual Concept Lattices

4.1 A Concept Lattice Generator

Concept lattices to be used as machine translation filters should contain concept nodes associated with both functional and substantive words. All languages have a finite number of functional words. Therefore, a manual construction of the lattice fragments that would contain them would be reasonable. However, manually constructing a concept lattice for lexical words would have considerable drawbacks such as the following:

- It is labor intensive.
- It is prone to yielding errors which are difficult to detect automatically.
- It generates incomplete lists that are costly to extend to cover missing information.
- It is not easy to adapt to changes and domain-specific needs.

Taking these potential problems into consideration, we have developed a tool for generating concept lattices for lexical words automatically. As this is an FCA application, it is crucial to decide on which formal context to use before delving its implementation details.

Priss & Old [16] propose to construct concept neighborhoods in WordNet with a formal context where the formal objects are the words of the synsets belonging to all senses of a word, the formal attributes are the words of the hypernymic synsets and the incidence relation is the semantic relation between the synsets and their hypernymic synsets. The neighborhood lattice of a word in WordNet consists of all words that share some senses with that word.[1] Below is the neighborhood lattice their method yields for the word *volume*:

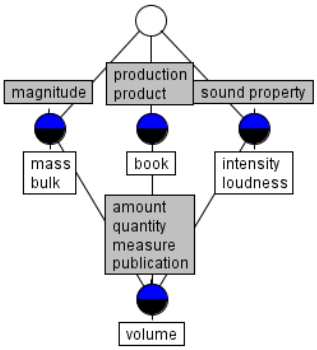


Fig. 3. – Priss and Old's neighborhood lattice for the word *volume*.

[1] As lattices often grow very rapidly to a size too large to be visualized, Wille [18] describes a method for constructing smaller, so-called "neighborhood" lattices.

Consider the bottom node. The concept represented by this node is not a naturally occurring one. Obviously, the adopted formal context causes two distinct natural concepts to collapse into one single formal concept here. The reason is simply that WordNet employs one single word, i.e., *volume*, for two distinct senses, i.e., *publication* and *amount*. This could leave a translation attempt with the task of disambiguating this word. In fact, WordNet marks each sense with a single so-called synset number.

When constructing concept lattices in WordNet, we suggest two amendments to the formal context adopted by Priss and Old. First, the formal objects are to be the synset numbers. Second, the formal attributes are to include also some information compiled from the glosses of the words. The first change allows us to distinguish between the two senses of the word *volume*, as shown in Fig. 4a. But, we are still far from resolving all ambiguities concerning this word, as indicated by the presence of two objects in the leftmost node. The problem is that the hypernymic attributes are not sufficiently informative to differentiate the *3-D space* sense of the word *volume* from its *relative amount* sense. This extra information resides in the glosses of the word and once encoded as attributes it evokes the required effect, as shown in Fig. 4b.

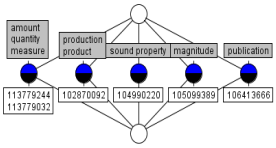


Fig. 4a. – A neighborhood lattice with the objects being synset numbers.

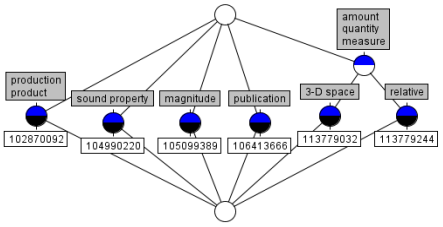


Fig. 4b. – A more fine-grained neighborhood lattice with the objects being synset numbers.

Each gloss, which is most likely a noun phrase, is parsed by means of a shift-reduce parser to extract a set of attributes. Having collected the objects (i.e. the synset numbers) and the associated attributes, the FCA algorithm that comes with the FCALGS library [9] is used for deriving a lattice-based ontology from that collection. FCALGS employs a parallel and recursive algorithm. Apart from its being parallel, it is very similar to Kuznetsov's [10] Close-by-One algorithm.

However, even the lattice in Fig4.b is still defective in at least one respect. The names of the objects denoted are lost. To remedy this problem, we suggest to encode the objects as tuples of synset numbers and sets of names, as illustrated below.

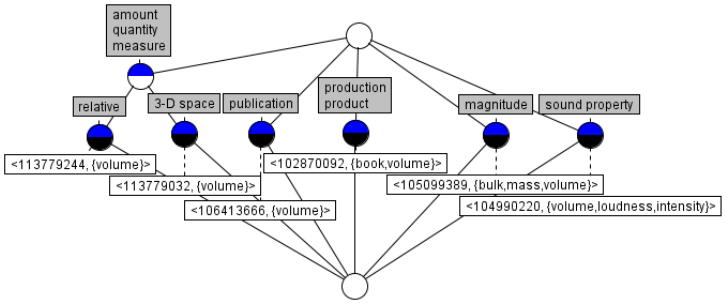


Fig. 5. – A neighborhood lattice including the names of the objects.

Another point to note is that the name of a synset serves as the attribute of a subconcept. For example, 'entity' is the name of the topmost synset. But, as everything is an entity, any subconcept must treat it as an element of its set of attributes.

4.2 A Sense Translator

Each WordNet node is associated with a set of synonymous English words, which is referred to as its synset. Each synset, in effect, denotes a sense in English. Thus, one task to accomplish is to translate synsets into Turkish to the furthest possible extent. We should, of course, keep in mind that some synsets (i.e. some senses encoded in English) may not have a counterpart in the target language. To find the Turkish translation of a particular synset, the Sense Translator first downloads a set of relevant articles via the links given in the disambiguation pages Wikipedia provides for the words in this set. It searches for the hypernyms of the synset in these articles. It assigns each article a score in accordance with the sum of the weighted points of the hypernyms found in this article. More specifically, if a synset has N hypernyms, the K^{th} hypernym starting from the top is assigned $Weight_K = K/N$. Let $Frequency_K$ be the number of occurrences of an item in a given article, then the score of the article is calculated as follows:

$$\textbf{Article Score} = \textbf{Weight}_1 * \textbf{Frequency}_1 + \ldots + \textbf{Weight}_N * \textbf{Frequency}_N. \quad (1)$$

If the article with the highest score has a link to a Turkish article, the title of the article will be the translation of the English word under examination. Otherwise, the word will be left unpaired with a Turkish counterpart. Figure 6 visualizes how the word *cat* in WordNet is translated into its Turkish counterpart, kedi, via Wikipedia.

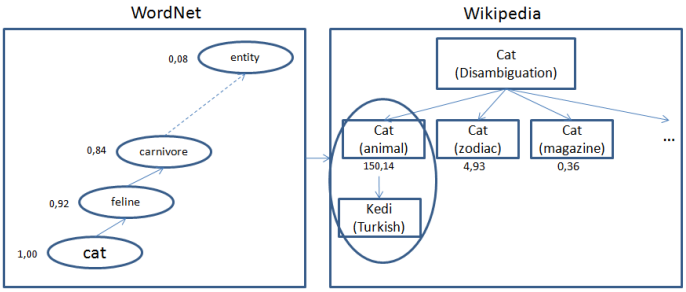


Fig. 6. - Translating the word *cat* into Turkish via Wikipedia.

The Turkish counterparts will be added next to the English names, as shown below:

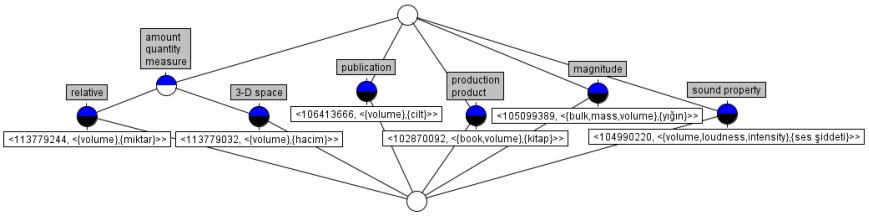


Fig. 7. - A neighborhood lattice including the Turkish counterparts of the English names.

4.3 A Rule-Based Machine Translator

We have designed a transfer-based architecture for Turkish-English translation and implemented the translator in SWI-Prolog which is an open-source implementation of the Prolog programming language. Below is a figure representing the main modules of the translator:

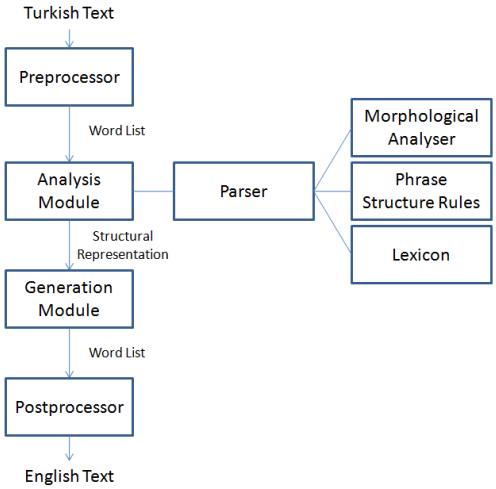


Fig. 8. - The main modules of the rule-based machine translator.

The word list extracted by the Preprocessor is used as an input to the Analysis Module. We have devised a shift-reduce parser in the analysis phase for building up the grammatical structure of expressions. Briefly, a shift-reduce parser uses a bottom-up strategy with an ultimate goal of building trees rooted with a start symbol [1]. The Generation Module first rearranges the constituents using transformation rules. Afterwards, all the structures are lexically transferred into English using a bilingual dictionary.

4.4 Filtering Translation Results with the Concept Lattice

Let us turn to our exemplary sentence introduced in (2) (i.e. *Kadın onu tanıyor*). Failing to take the context of the sentence into account, the rule-based translator generates the result in (8) (i.e. *The woman knows him/her/it*), where the pronoun is left ambiguous with respect to gender.

Our claim is that we can resolve such ambiguities using FCA and thereby refine our translations. To this effect, we propose to generate transient formal concepts for noun phrases. We make the following assumptions. Basically, personal pronouns, determiners and proper names introduce formal objects whereas adjectives and nouns encode formal attributes.

Suppose that our sentence is preceded by (the Turkish paraphrase of) a sentence like '*A man has arrived*'. The indefinite determiner evokes a new formal object, say *obj1*. As the source text is in Turkish, all attributes will be Turkish words. The Turkish counterpart of the word *man* is *adam*. Thus, the transient concept for the subject of this sentence will be ({*obj1*}, {adam}). The task is now to embed this transient concept into the big permanent concept lattice. To do this, a node where the Turkish counterpart of the synset name is 'adam' is searched for. Immediately below this node is placed a new node with its set of objects being {*obj1*} and with no additional attributes. As this is a physical object, the subconcept of this new node has to be the

lowest one. As for the second sentence, the NP *kadın* (the woman) will be associated with the transient concept ({X},{kadın}) and the pronoun *onu* (him/her/it) with the transient concept ({Y},{entity}). X and Y are parameters to be anchored to particular formal objects. In other words, they are anaphoric. It seems plausible to assert that the attributes of an anaphoric object must constitute a (generally proper) subset or hypernym set of the attributes of the object serving as the antecedent. Assume that X is somehow anaphorically linked to an object *obj2*. Now, there are two candidate antecedents for Y. The woman, or the object *obj2*, is barred from being antecedent of the pronoun by a locality principle like one stated in Chomsky's [3] Binding Theory: roughly stated, a pronoun and its antecedent cannot occur in the same clause. There remains one single candidate antecedent, *obj1*. As its attribute set is a hyponym set of {entity}, it can be selected as a legitimate antecedent. The concept node created for the man will also be the one denoted by the pronoun with Y being instantiated with *obj1*. In the concept lattice constructed in WordNet, the concept named as 'man' includes 'male person' in its set of attributes. Hence, the ambiguity is resolved and the pronoun translates into English as 'him'.

It is worth noting that in case there is more than one candidate antecedent, an anaphora resolution technique, especially a statistical one, can be employed to pick out the candidate most likely to be the antecedent. The interested reader is referred to Mitkov [12] for a survey of anaphora resolution approaches in general and to Kılıçaslan et al [7] for anaphora resolution in Turkish.

The gender disambiguation process can also be carried out for common nouns. Consider the following fragment taken from a child story:

(9) Kız kardeş-ler-i Sinderella-ya gül-müş-ler.
girl {brother / sister} -pl-acc Cinderella-dat laugh-pst-3pl
'His/her sisters laughed at Cinderella.'

Kardeş-ler-i ayakkabı-nın ona
{brother / sister} -pl-poss3 shoe-gen {him / her / it}

uy-ma-yacağ-ı-na inan-ıyor-lar-mış.
fit-neg-fut-poss3-dat believe-cont-agr3pl-pst

'His/her sisters/brothers believed that the shoe would not fit him/her.'

Turkish, leaves not only pronouns but also many other words ambiguous with respect to the gender feature. The word 'kardeş' in this example is ambiguous between the translations *sister* and *brother*. This ambiguity will be resolved in favor of the former interpretation in way similar to the disambiguation process sketched out for pronouns above.

In fact, the problem of sense disambiguation is a kind of specification problem. Therefore, it cannot be confined to gender disambiguation. For example, given that we have somehow managed to compile the attributes listed in the column on the left-hand side, our FCA-based system generates the translations listed on the right-hand side:

zehirli, diş 'poisonous, tooth'	*fang*
zehirli, mantar 'poisonous, mushroom'	*toadstool*
sivri, diş 'sharp, tooth'	*fang*
arka, koltuk 'rear, seat'	*rumble*
acemi, asker 'inexperienced, soldier'	*recruit*

It will, of course, be interesting to try to solve other kinds of translation problems with FCA-based techniques. We leave this task to accomplish in the light of further research in the future.

5 Results and Evaluation

In the early years of MT, the quality of an MT system was determined by human judgment. Though specially trained for the purpose, human judges are prone to suffer at least from subjectivity. Besides, this exercise is almost always more costly and time consuming. Some automated evaluation metrics have been developed in order to overcome such problems. Among these are BLEU, NIST, WER and PER.

BLEU [14] and NIST [4] are rather specialized metrics. They are employed by considering the fraction of output n-grams that also appear in a set of human translations (n-gram precision). This allows the acknowledgment of a greater diversity of acceptable MT results.

As for WER (Word Error Rate) and PER (Position-independent Word Error Rate), they are more general purpose measures and they rely on direct correspondence between the machine translation and a single human-produced reference. WER is based on the Levenshtein distance [11] which is the edit distance between a reference translation and its automatic translation, normalized by the length of the reference translation. This metric is formulated as:

$$WER = \frac{S+D+I}{N} \qquad (2)$$

where N is the total number of words in the reference translation, S is the number of substituted words in the automatic translation, D is the number of words deleted from the automatic translation and I is the number of words inserted in the reference not appearing in the automatic translation.

Although WER requires exactly the same order of the words in automatic translation and reference, PER neglects word order completely [17]. It measures the difference in the count of the words occurring in automatic and reference translations. The resulting number is divided by the number of words in the reference. It is worth noting that PER is technically not a distance measure as it uses a position-independent Levenshtein distance where the distance between a sentence and one of its permutations is always taken to be zero.

We used WER to evaluate the performance of our MT system. This is probably the metric most commonly used for similar purposes. As we employed a single human-produced reference, this metric suits well to our evaluation setup. We fed our system

with a Turkish child story involving 91 sentences (970 words).[2] We post-edited the resulting translation in order to generate a reference. When necessary calculations were done in accordance with formula (1), the WER turned out to be 38%.

The next step was to see the extent to which the performance or our MT system could be improved using concept lattices as filters for the raw results. To this effect, we devised several concept lattices like that in figure 3 and filtered the lexical constituents of each automatic translation with them.

A considerable regression in error rate is observed in our system supplemented with concept lattices: the WER score is reduced down to a value around 30%.

One question that comes to mind at this point is that of whether the improvement achieved is statistically significant or not. To get an answer we had recourse to the Wilcoxon Signed-Rank test. This test is used to analyze matched-pair numeric data, looking at the difference between the two values in each matched pair. When applied to the WER scores of the non-filtered and filtered translation results, the test shows that the difference is statistically significant ($p < 0.005$).

Another question is that of whether the results are practically satisfactory. To get some insight to this question, we should employ a baseline system for a comparison on usability. Google Translate, a statistical MT system that stands out with its wide range of users, can serve for this purpose. The WER score obtained employing Google Translate on our data is 34%. Recalling that the WER score of our system supplemented with concept lattices is 30%, we seem to be entitled to argue for the viability of rule-based MT systems. Of course, we need to make this claim tentatively since the size of the data on which the comparisons are made is relatively small. However, it should also be noted that we have employed a limited number of concept lattices of considerably small sizes. It is of no doubt that increasing the number and size of filtering lattices would improve the performance of our MT system.

More importantly, we do not primarily have an NLP concern in this work. Rather, we would like the results to be evaluated from a computational linguistics perspective. Everything aside, the results show that even a toy lattice based ontology can yield statistically significant improvement for an MT system.

6 Conclusion

In this paper, we have illustrated some translation problems caused by some typological divergences between Turkish and English using a particular example. We have gone through the direct translation, syntactic transfer and semantic transfer phases of the rule-based translation model to see what problem is dealt with in what phase. We have seen that a context-dependent pragmatic process is necessary to get to a satisfactory result. Concept lattices appear to be very efficient tools for accomplishing this pragmatic disambiguation task. Supplementing a rule-based MT system with concept lattices not only yields statistically significant improvement on the results of the system but also enables it to compete with a statistical MT system like Google Translate.

[2] This is the story where the example in (9) comes from.

References

1. Aho, A.V., Ullman, J.D.: *The Theory of Parsing, Translation, and Compiling*, Vol. 1., Prentice Hall (1972)
2. Barwise J., Seligman, J.: *Information Flow. The Logic of Distributed Systems*. Cambridge University Press (1997)
3. Chomsky, N.: *Lectures on Government and Binding*, Foris, Dordrecht (1981).
4. Doddington, G.: "Automatic Evaluation of Machine Translation Quality Using N-gram Co-occurrence Statistics". In *Proceedings of HLT 2002 (2nd Conference on Human Language Technology)*. San Diego, California, 128-132 (2002)
5. Ganter, B., Wille, R.: *Formale Begriffsanalyse: Mathematische Grundlagen*. Berlin: Springer (1996)
6. Jurafsky, D., Martin, J. H.: *Speech and Language Processing*, 2nd Edition, Prentice Hall (2009)
7. Kılıçaslan, Y., Güner, E. S., Yıldırım, S.: Learning-based pronoun resolution for Turkish with a comparative evaluation, Computer Speech & Language, Volume 23, Issue 3, p. 311-331 (2009)
8. Kipke, U., Wille, R.: *Formale Begriffsanalyse erläutert an einem Wortfeld*. LDV–Forum, 5 (1987)
9. Krajca, P., Outrata, J., Vychodil, V.: *Parallel Recursive Algorithm for FCA*. In: Belohlavek R., Kuznetsov S. O. (Eds.): Proc. CLA 2008, CEUR WS, 433, 71–82 (2008)
10. Kuznetsov, S.: *Learning of Simple Conceptual Graphs from Positive and Negative Examples*. PKDD 1999, pp. 384–391 (1999)
11. Levenshtein, V. I.: "Binary codes capable of correcting deletions, insertions, and reversals," Tech. Rep. 8. (1966)
12. Mitkov, R.: Anaphora Resolution: The State of the Art. Technical Report, University of Wolverhampton (1999)
13. Old, L. J., Priss, U.: *Metaphor and Information Flow*. In Proceedings of the 12th Midwest Artificial Intelligence and Cognitive Science Conference, pp. 99-104 (2001)
14. Papineni, K., Roukos, S., Ward, T., Zhu, W. J.: "BLEU: a method for automatic evaluation of machine translation" in *ACL-2002: 40th Annual meeting of the Association for Computational Linguistics* pp. 311–318 (2002)
15. Priss, U.: Linguistic Applications of Formal Concept Analysis, Ganter; Stumme; Wille (eds.), *Formal Concept Analysis, Foundations and Applications*, Springer Verlag, LNAI 3626, pp. 149-160 (2005)
16. Priss, U., Old, L. J.: "Concept Neighbourhoods in Lexical Databases.", In Proceedings of the 8th International Conference on Formal Concept Analysis, ICFCA'10, Springer Verlag, LNCS 5986, p. 283-295 (2010)
17. Tillmann C., Vogel, S., Ney, H., Zubiaga A., Sawaf, H.: Accelerated DP based search for statistical translation. In *European Conf. on Speech Communication and Technology*, pages 2667–2670, Rhodes, Greece, September (1997)
18. Wille, R.: *The Formalization of Roget's International Thesaurus.* Unpublished manuscript (1993)

Concept lattices in fuzzy relation equations*

Juan Carlos Díaz and Jesús Medina**

Department of Mathematics. University of Cádiz
Email: {juancarlos.diaz,jesus.medina}@uca.es

Abstract. Fuzzy relation equations are used to investigate theoretical and applicational aspects of fuzzy set theory, e.g., approximate reasoning, time series forecast, decision making and fuzzy control, etc.. This paper relates these equations to a particular kind of concept lattices.

1 Introduction

Recently, multi-adjoint property-oriented concept lattices have been introduced in [16] as a generalization of property-oriented concept lattices [10,11] to a fuzzy environment. These concept lattices are a new point of view of rough set theory [23] that considers two different sets: the set of objects and the set of attributes.

On the other hand, fuzzy relation equations, introduced by E. Sanchez [28], are associated to the composition of fuzzy relations and have been used to investigate theoretical and applicational aspects of fuzzy set theory [22], e.g., approximate reasoning, time series forecast, decision making, fuzzy control, as an appropriate tool for handling and modeling of nonprobabilistic form of uncertainty, etc. Many papers have investigated the capacity to solve (systems) of fuzzy relation equations, e.g., in [1,8,9,25,26].

In this paper, the multi-adjoint relation equations are presented as a generalization of the fuzzy relation equations [24,28]. This general environment inherits the properties of the multi-adjoint philosophy, consequently, e.g., several conjunctors and residuated implications defined on general carriers as lattice structures can be used, which provide more flexibility in order to relate the variables considered in the system.

Moreover, multi-adjoint property-oriented concept lattices and systems of multi-adjoint relation equations have been related in order to obtain results that ensure the existence of solutions in these systems. These definitions and results are illustrated by a toy example to improve the readability and comprehension of the paper.

Among all concept lattice frameworks, we have related the multi-adjoint property-oriented concept lattices to the systems of multi-adjoint relation equations, e.g., the extension and intension operators of this concept lattice can be

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

** Corresponding author.

used to represent multi-adjoint relation equations, and, as a result, the solutions of these systems of relation equations can be related to the concepts of the corresponding concept lattice.

The more important consequence is that this relation provides that the properties given, e.g., in [2–4,12,14,17,18,27] can be applied to obtain many properties of these systems. Indeed, it can be considered that the algorithms presented, e.g., in [5, 6, 15] obtain solutions for these systems.

The plan of this paper is the following: in Section 2 we will recall the multi-adjoint property-oriented concept lattices as well as the basic operators used and some properties; later, in Section 3, an example will be introduced to motivate the multi-adjoint relation equations. Once these equations have been presented, in Section 4 the multi-adjoint property-oriented concept lattices and the systems of multi-adjoint relation equations will be related in order to obtain results which ensure the existence of solutions in these systems; the paper ends with some conclusions and prospects for future work.

2 Multi-adjoint property-oriented concept lattices

The basic operators in this environment are the adjoint triples, which are formed by three mappings: a non-commutativity conjunctor and two residuated implications [13], which satisfy the well-known adjoint property.

Definition 1. *Let* $(P_1, \leq_1)$, $(P_2, \leq_2)$, $(P_3, \leq_3)$ *be posets and* $\&\colon P_1 \times P_2 \to P_3$, $\swarrow\colon P_3 \times P_2 \to P_1$, $\nwarrow\colon P_3 \times P_1 \to P_2$ *be mappings, then* $(\&, \swarrow, \nwarrow)$ *is an* adjoint triple *with respect to* P_1, P_2, P_3 *if:*

1. $\&$ *is order-preserving in both arguments.*
2. $\swarrow$ *and* $\nwarrow$ *are order-preserving on the first argument*[1] *and order-reversing on the second argument.*
3. $x \leq_1 z \swarrow y$ *iff* $x \,\&\, y \leq_3 z$ *iff* $y \leq_2 z \nwarrow x$, *where* $x \in P_1$, $y \in P_2$ *and* $z \in P_3$.

Example of adjoint triples are the Gödel, product and Łukasiewicz t-norms together with their residuated implications.

Example 1. Since the Gödel, product and Łukasiewicz t-norms are commutative, the residuated implications satisfy that $\swarrow^G = \nwarrow_G$, $\swarrow^P = \nwarrow_P$ and $\swarrow^L = \nwarrow_L$. Therefore, the Gödel, product and Łukasiewicz adjoint triples are defined on $[0, 1]$ as:

$$\&_P(x, y) = x \cdot y \qquad z \nwarrow_P x = \min(1, z/x)$$

$$\&_G(x, y) = \min(x, y) \qquad z \nwarrow_G x = \begin{cases} 1 & \text{if } x \leq z \\ z & \textit{otherwise} \end{cases}$$

$$\&_L(x, y) = \max(0, x + y - 1) \qquad z \nwarrow_G x = \min(1, 1 - x + z)$$

[1] Note that the antecedent will be evaluated on the right side, while the consequent will be evaluated on the left side, as in logic programming framework.

In [19] more general examples of adjoint triples are given.

The basic structure, which allows the existence of several adjoint triples for a given triplet of lattices, is the multi-adjoint property-oriented frame.

Definition 2. *Given two complete lattices* $(L_1, \preceq_1)$ *and* $(L_2, \preceq_2)$*, a poset* $(P, \leq)$ *and adjoint triples with respect to* P, L_2, L_1*,* $(\&_i, \swarrow^i, \nwarrow_i)$*, for all* $i = 1, \ldots, l$*, a* multi-adjoint property-oriented frame *is the tuple*

$$(L_1, L_2, P, \preceq_1, \preceq_2, \leq, \&_1, \swarrow^1, \nwarrow_1, \ldots, \&_l, \swarrow^l, \nwarrow_l)$$

Multi-adjoint property-oriented frames are denoted as $(L_1, L_2, P, \&_1, \ldots, \&_l)$. Note that the notation is similar to a multi-adjoint frame [18], although the adjoint triples are defined on different carriers.

The definition of context is analogous to the one given in [18].

Definition 3. *Let* $(L_1, L_2, P, \&_1, \ldots, \&_l)$ *be a multi-adjoint property-oriented frame. A* context *is a tuple* (A, B, R, σ) *such that* A *and* B *are non-empty sets (usually interpreted as attributes and objects, respectively),* R *is a* P*-fuzzy relation* $R\colon A \times B \to P$ *and* $\sigma\colon B \to \{1, \ldots, l\}$ *is a mapping which associates any element in* B *with some particular adjoint triple in the frame.*[2]

From now on, we will fix a multi-adjoint property-oriented frame and context, $(L_1, L_2, P, \&_1, \ldots, \&_l)$, (A, B, R, σ).

Now we define the following mappings $^{\uparrow_\pi}\colon L_2^B \to L_1^A$ and $^{\downarrow^N}\colon L_1^A \to L_2^B$ as

$$g^{\uparrow_\pi}(a) = \sup\{R(a,b) \,\&_{\sigma(b)}\, g(b) \mid b \in B\} \tag{1}$$

$$f^{\downarrow^N}(b) = \inf\{f(a) \nwarrow_{\sigma(b)} R(a,b) \mid a \in A\} \tag{2}$$

Clearly, these definitions[3] generalize the classical possibility and necessity operators [11] and they form an isotone Galois connection [16]. There are two dual versions of the notion of Galois connetion. The most famous Galois connection, where the maps are order-reversing, is properly called *Galois connection*, and the other in which the maps are order-preserving, will be called *isotone Galois connection*. In order to make this contribution self-contained, we recall their formal definitions:

Let $(P_1, \leq_1)$ and $(P_2, \leq_2)$ be posets, and $^{\downarrow}\colon P_1 \to P_2$, $^{\uparrow}\colon P_2 \to P_1$ mappings, the pair $(^{\uparrow}, ^{\downarrow})$ forms a *Galois connection* between P_1 and P_2 if and only if: $^{\uparrow}$ and $^{\downarrow}$ are order-reversing; $x \leq_1 x^{\downarrow\uparrow}$, for all $x \in P_1$, and $y \leq_2 y^{\uparrow\downarrow}$, for all $y \in P_2$.

The one we adopt here is the dual definition: Let $(P_1, \leq_1)$ and $(P_2, \leq_2)$ be posets, and $^{\downarrow}\colon P_1 \to P_2$, $^{\uparrow}\colon P_2 \to P_1$ mappings, the pair $(^{\uparrow}, ^{\downarrow})$ forms an *isotone Galois connection* between P_1 and P_2 if and only if: $^{\uparrow}$ and $^{\downarrow}$ are order-preserving; $x \leq_1 x^{\downarrow\uparrow}$, for all $x \in P_1$, and $y^{\uparrow\downarrow} \leq_2 y$, for all $y \in P_2$.

[2] A similar theory could be developed by considering a mapping $\tau\colon A \to \{1, \ldots, l\}$ which associates any element in A with some particular adjoint triple in the frame.

[3] From now on, to improve readability, we will write $\&_b$, $\nwarrow_b$ instead of $\&_{\sigma(b)}$, $\nwarrow_{\sigma(b)}$.

A concept, in this environment, is a pair of mappings $\langle g, f\rangle$, with $g \in L^B, f \in L^A$, such that $g^{\uparrow_\pi} = f$ and $f^{\downarrow^N} = g$, which will be called *multi-adjoint property-oriented formal concept*. In that case, g is called the *extension* and f, the *intension* of the concept. The set of all these concepts will be denoted as $\mathcal{M}_{\pi N}$ [16].

Definition 4. *A* multi-adjoint property-oriented concept lattice *is the set*

$$\mathcal{M}_{\pi N} = \{\langle g, f\rangle \mid g \in L_2^B, f \in L_1^A \text{ and } g^{\uparrow_\pi} = f, f^{\downarrow^N} = g\}$$

in which the ordering is defined by $\langle g_1, f_1\rangle \preceq \langle g_2, f_2\rangle$ *iff* $g_1 \preceq_2 g_2$ *(or equivalently* $f_1 \preceq_1 f_2$*).*

The pair $(\mathcal{M}_{\pi N}, \preceq)$ is a complete lattice [16], which generalize the concept lattice introduced in [7] to a fuzzy environment.

3 Multi-adjoint relation equations

This section begins with an example that motivates the definition of multi-adjoint relation equations, which will be introduced later.

3.1 Multi-adjoint logic programming

A short summary of the main features of multi-adjoint languages will be presented. The reader is referred to [20, 21] for a complete formulation.

A language $\mathcal{L}$ contains propositional variables, constants, and a set of logical connectives. In this fuzzy setting, the usual connectives are adjoint triples and a number of aggregators.

The language $\mathcal{L}$ is interpreted on a *(biresiduated) multi-adjoint lattice*,[4] $\langle L, \preceq, \swarrow^1, \nwarrow_1, \&_1, \ldots, \swarrow^n, \nwarrow_n, \&_n\rangle$, which is a complete lattice L equipped with a collection of adjoint triples $\langle \&_i, \swarrow^i, \nwarrow_i\rangle$, where each $\&_i$ is a conjunctor intended to provide a *modus ponens*-rule with respect to $\swarrow^i$ and $\nwarrow_i$.

A *rule* is a formula $A \swarrow^i \mathcal{B}$ or $A \nwarrow_i \mathcal{B}$, where A is a propositional symbol (usually called the *head*) and $\mathcal{B}$ (which is called the *body*) is a formula built from propositional symbols $B_1, \ldots, B_n$ $(n \geq 0)$, truth values of L and conjunctions, disjunctions and aggregations. Rules with an empty body are called *facts*.

A *multi-adjoint logic program* is a set of pairs $\langle \mathcal{R}, \alpha\rangle$, where $\mathcal{R}$ is a rule and α is a value of L, which may express the confidence which the user of the system has in the truth of the rule $\mathcal{R}$. Note that the truth degrees in a given program are expected to be assigned by an expert.

Example 2. Let us to consider a multi-adjoint lattice

$$\langle [0,1], \leq, \leftarrow_G, \&_G, \leftarrow_P, \&_P, \wedge_L\rangle$$

[4] Note that a multi-adjoint lattice is a particular case of a multi-adjoint property-oriented frame.

where $\&_G$ and $\&_P$ are the Gödel and product conjunctors, respectively, and $\leftarrow_G$, $\leftarrow_P$ their corresponding residuated implications. Moreover, the Łukasiewicz conjunctor $\wedge_L$ will be used in the program [13].

Given the set of variables (propositional symbols)

$$\Pi = \{\texttt{low\_oil}, \texttt{low\_water}, \texttt{rich\_mixture}, \texttt{overheating}, \texttt{noisy\_behaviour}, \texttt{high\_fuel\_consumption}\}$$

the following set of multi-adjoint rules form a multi-adjoint program, which may represent the behaviour of a motor.

$$\begin{aligned}
\langle \texttt{high_fuel_consumption} &\leftarrow_G \texttt{rich_mixture} \wedge_L \texttt{low_oil}, 0.8\rangle\\
\langle \texttt{overheating} &\leftarrow_G \texttt{low_oil}, 0.5\rangle\\
\langle \texttt{noisy_behaviour} &\leftarrow_P \texttt{rich_mixture}, 0.8\rangle\\
\langle \texttt{overheating} &\leftarrow_P \texttt{low_water}, 0.9\rangle\\
\langle \texttt{noisy_behaviour} &\leftarrow_G \texttt{low_oil}, 1\rangle
\end{aligned}$$

The usual procedural is to measure the levels of "oil", "water" and "mixture" of a specific motor, after that the values for `low_oil`, `low_water` and `rich_mixture` are obtained, which are represented in the program as facts, for instance, the next ones can be added to the program:

$$\begin{aligned}
\langle \texttt{low_oil} &\leftarrow_P \top, 0.2\rangle\\
\langle \texttt{low_water} &\leftarrow_P \top, 0.2\rangle\\
\langle \texttt{rich_mixture} &\leftarrow_P \top, 0.5\rangle
\end{aligned}$$

Finally, the values for the rest of variables (propositional symbols) are computed [20]. For instance, in order to attain the value for $\texttt{overheating}(o, w)$, for a level of oil, o, and water, w, the rules $\langle \texttt{overheating} \leftarrow_G \texttt{low\_oil}, \vartheta_1\rangle$ and $\langle \texttt{overheating} \leftarrow_P \texttt{low\_water}, \vartheta_2\rangle$ are considered and its value is obtained as:

$$\texttt{overheating}(o, w) = (\texttt{low\_oil}(o) \,\&_G\, \vartheta_1) \vee (\texttt{low\_water}(w) \,\&_P\, \vartheta_2) \qquad (3)$$

Now, the problem could be to recompute the weights of the rules from experimental instances of the variables, that is, the values of `overheating`, `noisy_behaviour` and `high_fuel_consumption` are known for particular measures of `low_oil`, `low_water` and `rich_mixture`.

Specifically, given the levels of oil, $o_1, \ldots, o_n$, the levels of water, $w_1, \ldots, w_n$, and the measures of mixture, $t_1, \ldots, t_n$, we may experimentally know the values of the variables: $\texttt{noisy\_behaviour}(t_i, o_i)$, $\texttt{high\_fuel\_consumption}(t_i, o_i)$ and $\texttt{overheating}(o_i, w_i)$, for all $i \in \{1, \ldots, n\}$.

Considering Equation (3), the unknown elements could be ϑ_1 and ϑ_2 instead of $\texttt{overheating}(o, w)$. Therefore, the problem now is to look for the values of ϑ_1 and ϑ_2, which solve the following system obtained after assuming the experimental data for the propositional symbols, $ov_1, o_1, w_1, \ldots, ov_n, o_n, w_n$.

$$\begin{aligned}
\texttt{overheating}(ov_1) &= (\texttt{low_oil}(o_1) \,\&_G\, \vartheta_1) \vee (\texttt{low_water}(w_1) \,\&_P\, \vartheta_2)\\
&\vdots \qquad\qquad \vdots \qquad\qquad\qquad \vdots \qquad\qquad \vdots\\
\texttt{overheating}(ov_n) &= (\texttt{low_oil}(o_n) \,\&_G\, \vartheta_1) \vee (\texttt{low_water}(w_n) \,\&_P\, \vartheta_2)
\end{aligned}$$

This system can be interpreted as a system of fuzzy relation equations in which several conjunctors, $\&_G$ and $\&_P$, are assumed. Moreover, these conjunctors could be neither non-commutative nor associative and defined in general lattices, as permit the multi-adjoint framework.

Next sections introduce when these systems have solutions and a novel method to obtain them using concept lattice theory.

3.2 Systems of multi-adjoint relation equations

The operators used in order to obtain the systems will be the generalization of the sup-$*$-composition, introduced in [29], and inf-$\rightarrow$-composition, introduced in [1]. From now on, a multi-adjoint property-oriented frame, $(L_1, L_2, P, \&_1, \ldots, \&_l)$ will be fixed.

In the definition of a multi-adjoint relation equation an interesting mapping $\sigma\colon U \to \{1, \ldots, l\}$ will be considered, which relates each element in U to an adjoint triple. This mapping will play a similar role as the one given in a multi-adjoint context, defined in the previous section, for instance, this map provides a partition of U in preference sets. A similar theory may be developed for V instead of U.

Let $U = \{u_1, \ldots, u_m\}$ and $V = \{v_1, \ldots, v_n\}$ be two universes, $R \in {L_2}^{U \times V}$ an unknown fuzzy relation, $\sigma\colon U \to \{1, \ldots, l\}$ a map that relates each element in U to an adjoint triple, and $K_1, \ldots, K_n \in P^U$, $D_1, \ldots, D_n \in {L_1}^V$ arbitrarily chosen fuzzy subsets of the respective universes.

A *system of multi-adjoint relation equations with* sup-$\&$*-composition*, is the following system of equations

$$\bigvee_{u \in U} (K_i(u) \,\&_u\, R(u, v)) = D_i(v), \quad i \in \{1, \ldots, n\} \tag{4}$$

where $\&_u$ represents the adjoint conjunctor associated to u by σ, that is, if $\sigma(u) = (\&_s, \swarrow^s, \nwarrow_s)$, for $s \in \{1, \ldots, l\}$, then $\&_u$ is exactly $\&_s$.

If an element v of V is fixed and the elements $K_i(u_j)$, $R(u_j, v)$ and $D_i(v)$ are written as k_{ij}, x_j and d_i, respectively, for each $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, m\}$, then System (4) can particularly be written as

$$\begin{array}{ccc} k_{11} \,\&_{u_1}\, x_1 \vee \cdots \vee k_{1m} \,\&_{u_m}\, x_m = d_1 \\ \vdots \quad \vdots \qquad\qquad \vdots \quad \vdots \\ k_{n1} \,\&_{u_1}\, x_1 \vee \cdots \vee k_{nm} \,\&_{u_m}\, x_m = d_n \end{array} \tag{5}$$

where k_{ij} and d_i are known and x_j must be obtained.

Hence, for each $v \in V$, if we solve System (5), then we obtain a "column" of R (i.e. the elements $R(u_j, v)$, with $j \in \{1, \ldots, m\}$), thus, solving n similar systems, one for each $v \in V$, the unknown relation R is obtained.

Example 3. Assuming Example 2, in this case, we will try to solve the problem about to obtain the weights associated to the rules from particular observed data for the propositional symbols.

The propositional symbols (variables) will be written in short as: $\mathtt{hfc}$, $\mathtt{nb}$, $\mathtt{oh}$, $\mathtt{rm}$, $\mathtt{lo}$ and $\mathtt{lw}$, and the measures of particular cases of the behaviour of the motor will be: h_i, n_i, ov_i, r_i, o_i, w_i, for $\mathtt{hfc}$, $\mathtt{nb}$, $\mathtt{oh}$, $\mathtt{rm}$, $\mathtt{lo}$ and $\mathtt{lw}$, respectively, in each case i, with $i \in \{1, 2, 3\}$.

For instance, the next system associated to $\mathtt{overheating}$ is obtained from the computation provided in Example 2.

$$\begin{aligned}
\mathtt{oh}(ov_1) &= (\mathtt{lo}(o_1) \,\&_G\, \vartheta^{\mathtt{oh}}_{\mathtt{lo}}) \vee (\mathtt{lw}(w_1) \,\&_P\, \vartheta^{\mathtt{oh}}_{\mathtt{lw}}) \\
\mathtt{oh}(ov_2) &= (\mathtt{lo}(o_2) \,\&_G\, \vartheta^{\mathtt{oh}}_{\mathtt{lo}}) \vee (\mathtt{lw}(w_2) \,\&_P\, \vartheta^{\mathtt{oh}}_{\mathtt{lw}}) \\
\mathtt{oh}(ov_3) &= (\mathtt{lo}(o_3) \,\&_G\, \vartheta^{\mathtt{oh}}_{\mathtt{lo}}) \vee (\mathtt{lw}(w_3) \,\&_P\, \vartheta^{\mathtt{oh}}_{\mathtt{lw}})
\end{aligned}$$

where $\vartheta^{\mathtt{oh}}_{\mathtt{lo}}$ and $\vartheta^{\mathtt{oh}}_{\mathtt{lw}}$ are the weights associated to the rules with head $\mathtt{oh}$. Similar systems can be obtained to $\mathtt{high\_fuel\_consumption}$ and $\mathtt{noisy\_behaviour}$.

Assuming the multi-adjoint frame with carrier $L = [0, 1]$ and the Gödel and product triples, these systems are particular systems of multi-adjoint relational equations. The corresponding context is formed by the sets $U = \{\mathtt{rm}, \mathtt{lo}, \mathtt{lw}, \mathtt{rm} \wedge_L \mathtt{lo}\}$, $V = \{\mathtt{hfc}, \mathtt{nb}, \mathtt{oh}\}$; the mapping σ that relates the elements $\mathtt{lo}$, $\mathtt{rm} \wedge_L \mathtt{lo}$ to the Gödel triple, and $\mathtt{rm}$, $\mathtt{lw}$ to the product triple; the mappings $K_1, \ldots, K_n \in P^U$, defined as the values given by the propositional symbols in U on the experimental data, for instance, if $u = \mathtt{lo}$, then $K_1(\mathtt{lo}) = \mathtt{lo}(o_1)$, …, $K_n(\mathtt{lo}) = \mathtt{lo}(o_n)$; and the mappings $D_1, \ldots, D_n \in {L_1}^V$, defined analogously, for instance, if $v = \mathtt{rm}$, then $D_1(\mathtt{rm}) = \mathtt{rm}(r_1)$, …, $D_n(\mathtt{rm}) = \mathtt{rm}(r_n)$.

Finally, the unknown fuzzy relation $R \in {L_2}^{U \times V}$ is formed by the weights of the rules in the program.

In the system above, $\mathtt{oh}$ has been the element $v \in V$ fixed. Moreover, as there do not exist rules with body $\mathtt{rm}$ and $\mathtt{rm} \wedge_L \mathtt{lo}$, that is, the weights for that hypothetical rules are 0, then the terms $(\mathtt{rm}(r_i) \,\&_G\, 0 = 0$ and $(\mathtt{rm}(r_i) \wedge_L \mathtt{lo}(o_i) \,\&_P\, 0 = 0$ do not appear.

Its counterpart is a *system of multi-adjoint relation equations with* inf-$\nwarrow$-*composition*, that is,

$$\bigwedge_{v \in V} (R(u, v) \nwarrow_{u_j} K^*_j(v)) = E_j(u), \quad j \in \{1, \ldots, m\} \tag{6}$$

considered with respect to unknown fuzzy relation $R \in {L_1}^{U \times V}$, and where $K^*_1, \ldots, K^*_m \in P^V$ and $E_1, \ldots, E_m \in {L_2}^U$. Note that $\nwarrow_{u_j}$ represents the corresponding adjoint implication associated to u_j by σ, that is, if $\sigma(u_j) = (\&_s, \swarrow^s, \nwarrow_s)$, for $s \in \{1, \ldots, l\}$, then $\nwarrow_{u_j}$ is exactly $\nwarrow_s$. Remark that in System 6, the implication $\nwarrow_{u_j}$ does not depend of the element u, but of j. Hence, the implications used in each equation of the system are the same.

If an element u of U is fixed, fuzzy subsets $K^*_1, \ldots, K^*_m \in P^V$, $E_1, \ldots, E_m \in L_2^U$ are assumed, such that $K^*_j(v_i) = k_{ij}$, $R(u, v_i) = y_i$ and $E_j(u) = e_j$, for each $i \in \{1, \ldots, n\}$, $j \in \{1, \ldots, m\}$, then System (6) can particularly be written as

$$\begin{array}{rcl}
y_1 \nwarrow_{u_1} k_{11} \wedge \cdots \wedge y_n \nwarrow_{u_1} k_{n1} & = & e_1 \\
\vdots \qquad\qquad \vdots \qquad\qquad \vdots & \vdots & \\
y_1 \nwarrow_{u_m} k_{1m} \wedge \cdots \wedge y_n \nwarrow_{u_m} k_{nm} & = & e_m
\end{array} \tag{7}$$

Therefore, for each $u \in U$, we obtain a "row" of R (i.e. the elements $R(u, v_i)$, with $i \in \{1, \ldots, n\}$), consequently, solving m similar systems, the unknown relation R is obtained.

Systems (5) and (7) have the same goal, searching for the unknown relation R although the mechanism is different.

Analyzing these systems, we have that the left side of these systems can be represented by the mappings $C_K \colon L_2^m \to L_1^n, I_{K^*} \colon L_1^n \to L_2^m$, defined as:

$$C_K(\bar{x})_i = k_{i1} \,\&_{u_1}\, x_1 \vee \cdots \vee k_{im} \,\&_{u_m}\, x_m, \text{ for all } i \in \{1, \ldots, n\} \tag{8}$$

$$I_{K^*}(\bar{y})_j = y_1 \nwarrow_{u_j} k_{1j} \wedge \cdots \wedge y_n \nwarrow_{u_j} k_{nj}, \text{ for all } j \in \{1, \ldots, m\} \tag{9}$$

where $\bar{x} = (x_1, \ldots, x_m) \in L_2^m$, $\bar{y} = (y_1, \ldots, y_n) \in L_1^n$, and $C_K(\bar{x})_i$, $I_{K^*}(\bar{y})_j$ are the components of $C_K(\bar{x})$, $I_{K^*}(\bar{y})$, respectively, for each $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$.

Hence, Systems (5) and (7) can be written as:

$$C_K(x_1, \ldots, x_m) = (d_1, \ldots, d_n) \tag{10}$$

$$I_{K^*}(y_1, \ldots, y_n) = (e_1, \ldots, e_m) \tag{11}$$

respectively.

4 Relation between multi-adjoint property-oriented concept lattices and multi-adjoint relation equation

This section shows that Systems (5) and (7) can be interpreted in a multi-adjoint property-oriented concept lattice. And so, the properties given to the isotone Galois connection $^{\uparrow_\pi}$ and $^{\downarrow^N}$, as well as to the complete lattice $\mathcal{M}_{\pi N}$ can be used in the resolution of these systems.

First of all, the environment must be fixed. Hence, a multi-adjoint context (A, B, S, σ) will be considered, such that $A = V'$, $B = U$, where V' has the same cardinality as V, σ will be the mapping given by the systems and $S \colon A \times B \to P$ is defined as $S(v'_i, u_j) = k_{ij}$. Note that $A = V'$ is related to the mappings K_i, since $S(v'_i, u_j) = k_{ij} = K_i(u_j)$;

Now, we will prove that the mappings defined at the end of the previous section are related to the isotone Galois connection. Given $\mu \in L_2^B$, such that $\mu(u_j) = x_j$, for all $j \in \{1, \ldots, m\}$, the following equalities are obtained, for each $i \in \{1, \ldots, n\}$:

$$\begin{aligned} C_K(\bar{x})_i &= k_{i1} \,\&_{u_1}\, x_1 \vee \cdots \vee k_{im} \,\&_{u_m}\, x_m \\ &= S(v'_i, u_1) \,\&_{u_1}\, \mu(u_1) \vee \cdots \vee S(v'_i, u_m) \,\&_{u_m}\, \mu(u_m) \\ &= \sup\{S(v'_i, u_j) \,\&_{u_j}\, \mu(u_j) \mid j \in \{1, \ldots, m\}\} \\ &= \mu^{\uparrow_\pi}(v'_i) \end{aligned}$$

Therefore, the mapping $C_K \colon L_2^m \to L_1^n$ is equivalent to the mapping $^{\uparrow_\pi} \colon L_2^B \to L_1^A$, where an element $\bar{x}$ in L_2^m can be interpreted as a map μ in L_2^B, such that

$\mu(u_j) = x_j$, for all $j \in \{1, \ldots, m\}$, and the element $C_K(\bar{x})$ as the mapping $\mu^{\uparrow_\pi}$, such that $\mu^{\uparrow_\pi}(v'_i) = C_K(\bar{x})_i$, for all $i \in \{1, \ldots, n\}$.

An analogy can be developed applying the above procedure to mappings I_{K^*} and $^{\downarrow^N}$, obtaining that the mappings $I_{K^*}\colon L_1^n \to L_2^m$ and $^{\downarrow^N}\colon L_1^A \to L_2^B$ are equivalent.

As a consequence, the following result holds:

Theorem 1. *The mappings $C_K\colon L_2^m \to L_1^n, I_{K^*}\colon L_1^n \to L_2^m$, establish an isotone Galois connection. Therefore, $I_{K^*} \circ C_K\colon L_2^m \to L_2^m$ is a closure operator and $C_K \circ I_{K^*}\colon L_1^n \to L_1^n$ is an interior operator.*

As (C_A, I_{K^*}) is an isotone Galois connection, any result about the solvability of one system has its dual counterpart.

The following result explains when these systems can be solved and how a solution can be obtained.

Theorem 2. *System (5) can be solved if and only if $\langle \lambda_{\bar{d}}^{\downarrow^N}, \lambda_{\bar{d}} \rangle$ is a concept of $\mathcal{M}_{\pi N}$, where $\lambda_{\bar{d}}\colon A = \{v_1, \ldots, v_n\} \to L_1$, defined as $\lambda_{\bar{d}}(v_i) = d_i$, for all $i \in \{1, \ldots, n\}$. Moreover, if System (5) has a solution, then $\lambda_{\bar{d}}^{\downarrow^N}$ is the greatest solution of the system.*

Similarly, System (7) can be solved if and only if $\langle \mu_{\bar{e}}, \mu_{\bar{e}}^{\uparrow_\pi} \rangle$ is a concept of $\mathcal{M}_{\pi N}$, where $\mu_{\bar{e}}\colon B = \{u_1, \ldots, u_m\} \to L_2$, defined as $\mu_{\bar{e}}(u_j) = e_j$, for all $j \in \{1, \ldots, m\}$. Furthermore, if System (7) has a solution, then $\mu_{\bar{e}}^{\uparrow_\pi}$ is the smallest solution of the system.

The main contribution of the relation introduced in this paper is not only the above consequences, but a lot of other properties for Systems (5) and (7) that can be stabilized from the results proved, for example, in [2–4, 12, 14, 17, 18, 27].

Next example studies the system of multi-adjoint relation equations presented in Example 3.

Example 4. The aim will be to solve a small system in order to improve the understanding of the method. In the environment of Example 3, the following system will be solved assuming the experimental data: $\mathtt{oh}(ov_1) = 0.5$, $\mathtt{lo}(o_1) = 0.3$, $\mathtt{lw}(w_1) = 0.3$, $\mathtt{oh}(ov_2) = 0.7$, $\mathtt{lo}(o_2) = 0.6$, $\mathtt{lw}(w_2) = 0.8$, $\mathtt{oh}(ov_3) = 0.4$, $\mathtt{lo}(o_3) = 0.5$, $\mathtt{lw}(w_3) = 0.2$.

$$\begin{aligned}
\mathtt{oh}(ov_1) &= (\mathtt{lo}(o_1) \,\&_G\, \vartheta_{\mathtt{lo}}^{\mathtt{oh}}) \vee (\mathtt{lw}(w_1) \,\&_P\, \vartheta_{\mathtt{lw}}^{\mathtt{oh}}) \\
\mathtt{oh}(ov_2) &= (\mathtt{lo}(o_2) \,\&_G\, \vartheta_{\mathtt{lo}}^{\mathtt{oh}}) \vee (\mathtt{lw}(w_2) \,\&_P\, \vartheta_{\mathtt{lw}}^{\mathtt{oh}}) \\
\mathtt{oh}(ov_3) &= (\mathtt{lo}(o_3) \,\&_G\, \vartheta_{\mathtt{lo}}^{\mathtt{oh}}) \vee (\mathtt{lw}(w_3) \,\&_P\, \vartheta_{\mathtt{lw}}^{\mathtt{oh}})
\end{aligned}$$

where $\vartheta_{\mathtt{lo}}^{\mathtt{oh}}$ and $\vartheta_{\mathtt{lw}}^{\mathtt{oh}}$ are the variables.

The context is: $A = V' = \{1, 2, 3\}$, the set of observations, $B = U = \{\mathtt{lo}, \mathtt{lw}\}$, σ associates the propositional symbol `lo` to the Gödel triple and `lw` to the product triple. The relation $S\colon A \times B \to [0, 1]$ is defined in Table 1.

Therefore, considering the mapping $\lambda_{\mathtt{oh}}\colon A \to [0, 1]$ associated to the values of `overheating` in each experimental case, that is $\lambda_{\mathtt{oh}}(1) = 0.5$, $\lambda_{\mathtt{oh}}(2) = 0.7$,

Table 1. Relation S.

	low_oil	low_water
1	0.3	0.3
2	0.6	0.8
3	0.5	0.2

and $\lambda_{\mathtt{oh}}(3) = 0.4$; and the mapping $C_K \colon [0,1]^2 \to [0,1]^3$, defined in Equation (8), the system above can be written as

$$C_K(\vartheta^{\mathtt{oh}}_{\mathtt{lo}}, \vartheta^{\mathtt{oh}}_{\mathtt{lw}}) = \lambda_{\mathtt{oh}}$$

Since, by the comment above, there exists $\mu \in [0,1]^B$, such that $C_K(\vartheta^{\mathtt{oh}}_{\mathtt{lo}}, \vartheta^{\mathtt{oh}}_{\mathtt{lw}}) = \mu^{\uparrow_\pi}$, the goal will be to attain the mapping $\mu \in [0,1]^B$, such that $\mu^{\uparrow_\pi} = \lambda_{\mathtt{oh}}$, which can be found if and only if $((\lambda_{\mathtt{oh}})^{\downarrow^N}, \lambda_{\mathtt{oh}})$ is a multi-adjoint property-oriented concept in the considered context, by Theorem 2.

First of all, we compute $(\lambda_{\mathtt{oh}})^{\downarrow^N}$.

$$\begin{aligned}
(\lambda_{\mathtt{oh}})^{\downarrow^N}(\mathtt{lo}) &= \inf\{\lambda_{\mathtt{oh}}(1) \nwarrow_G S(1,\mathtt{lo}), \lambda_{\mathtt{oh}}(2) \nwarrow_G S(2,\mathtt{lo}), \lambda_{\mathtt{oh}}(3) \nwarrow_G S(3,\mathtt{lo})\} \\
&= \inf\{0.5 \nwarrow_G 0.3, 0.7 \nwarrow_G 0.6, 0.4 \nwarrow_G 0.5\} \\
&= \inf\{1, 1, 0.4\} = 0.4 \\
(\lambda_{\mathtt{oh}})^{\downarrow^N}(\mathtt{lw}) &= \inf\{0.5 \nwarrow_P 0.3, 0.7 \nwarrow_P 0.8, 0.4 \nwarrow_P 0.2\} \\
&= \inf\{1, 0.875, 1\} = 0.875
\end{aligned}$$

Now, the mapping $(\lambda_{\mathtt{oh}})^{\downarrow^N\uparrow_\pi}$ is obtained.

$$\begin{aligned}
(\lambda_{\mathtt{oh}})^{\downarrow^N\uparrow_\pi}(1) &= \sup\{S(1,\mathtt{lo}) \,\&_G (\lambda_{\mathtt{oh}})^{\downarrow^N}(\mathtt{lo}), S(1,\mathtt{lw}) \,\&_P (\lambda_{\mathtt{oh}})^{\downarrow^N}(\mathtt{lw})\} \\
&= \sup\{0.3 \,\&_G 0.4, 0.3 \,\&_P 0.875\} \\
&= \sup\{0.3, 0.2625\} = 0.3 \\
(\lambda_{\mathtt{oh}})^{\downarrow^N\uparrow_\pi}(2) &= \sup\{0.6 \,\&_G 0.4, 0.8 \,\&_P 0.875\} = 0.7 \\
(\lambda_{\mathtt{oh}})^{\downarrow^N\uparrow_\pi}(3) &= \sup\{0.5 \,\&_G 0.4, 0.2 \,\&_P 0.875\} = 0.4
\end{aligned}$$

Therefore, $((\lambda_{\mathtt{oh}})^{\downarrow^N}, \lambda_{\mathtt{oh}})$ is not a multi-adjoint property-oriented concept and thus, the considered system has no solution, although if the experimental value for `oh` had been 0.3 instead of 0.5, the system would have had a solution.

These changes could be considered in several applications where noisy variables exist and their values can be conveniently changed to obtain approximate solutions for the systems. Thus, if the experimental data for `overheating` are $\mathtt{oh}(ov_1) = 0.3$, $\mathtt{oh}(ov_2) = 0.7$ and $\mathtt{oh}(ov_2) = 0.4$, then the original system will have at least one solution and the values $\vartheta^{\mathtt{oh}}_{\mathtt{lo}}$, $\vartheta^{\mathtt{oh}}_{\mathtt{lw}}$ will be 0.4, 0.875, respectively for a solution. Consequently, the truth for the first rule is lower than for the second or it might be thought that it is more determinant in obtaining higher

values for `lw` than for `lo`. Another possibility is to consider that this conclusion about the certainty of the rules is not correct, in which case another adjoint triple might be associate to `lo`.

As a result, the properties introduced in several fuzzy formal concept analysis frameworks can be applied in order to obtain solutions of fuzzy relation equations, as well as in the multi-adjoint general framework.

Furthermore, in order to obtain the solutions of Systems (5) and (7), the algorithms developed, e.g., in [5, 6, 15], can be used.

5 Conclusions and future work

Multi-adjoint relation equations have been presented that generalize the existing definitions presented at this time. In this general environment, different conjunctors and residuated implications can be used, which provide more flexibility in order to relate the variables considered in the system.

A toy example has been introduced in the paper in order to improve its readability and reduce the complexity of the definitions and results.

As a consequence of the results presented in this paper, several of the properties provided, e.g., in [2–4, 12, 14, 17, 18, 27], can be used to obtain additional characteristics of these systems.

In the future, we will apply the results provided in the fuzzy formal concept analysis environments to the general systems of fuzzy relational equations presented here.

References

1. W. Bandler and L. Kohout. Semantics of implication operators and fuzzy relational products. *Int. J. Man-Machine Studies*, 12:89–116, 1980.
2. 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.
3. R. Bělohlávek. Lattices of fixed points of fuzzy Galois connections. *Mathematical Logic Quartely*, 47(1):111–116, 2001.
4. R. Bělohlávek. Concept lattices and order in fuzzy logic. *Annals of Pure and Applied Logic*, 128:277–298, 2004.
5. R. Bělohlávek, B. D. Baets, J. Outrata, and V. Vychodil. Lindig's algorithm for concept lattices over graded attributes. *Lecture Notes in Computer Science*, 4617:156–167, 2007.
6. R. Bělohlávek, B. D. Baets, J. Outrata, and V. Vychodil. Computing the lattice of all fixpoints of a fuzzy closure operator. *IEEE Transactions on Fuzzy Systems*, 18(3):546–557, 2010.
7. Y. Chen and Y. Yao. A multiview approach for intelligent data analysis based on data operators. *Information Sciences*, 178(1):1–20, 2008.
8. 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.

9. A. Di Nola, S. Sessa, W. Pedrycz, and E. Sanchez. *Fuzzy Relation Equations and Their Applications to Knowledge Engineering*. Kluwer, 1989.
10. I. Düntsch and G. Gediga. Approximation operators in qualitative data analysis. In *Theory and Applications of Relational Structures as Knowledge Instruments*, pages 214–230, 2003.
11. G. Gediga and I. Düntsch. Modal-style operators in qualitative data analysis. In *Proc. IEEE Int. Conf. on Data Mining*, pages 155–162, 2002.
12. G. Georgescu and A. Popescu. Non-dual fuzzy connections. *Arch. Math. Log.*, 43(8):1009–1039, 2004.
13. P. Hájek. *Metamathematics of Fuzzy Logic*. Trends in Logic. Kluwer Academic, 1998.
14. 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.
15. C. Lindig. Fast concept analysis. In G. Stumme, editor, *Working with Conceptual Structures-Contributions to ICCS 2000*, pages 152–161, 2000.
16. J. Medina. Towards multi-adjoint property-oriented concept lattices. *Lect. Notes in Artificial Intelligence*, 6401:159–166, 2010.
17. J. Medina and M. Ojeda-Aciego. Multi-adjoint t-concept lattices. *Information Sciences*, 180(5):712–725, 2010.
18. 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.
19. J. Medina, M. Ojeda-Aciego, A. Valverde, and P. Vojtáš. Towards biresiduated multi-adjoint logic programming. *Lect. Notes in Artificial Intelligence*, 3040:608–617, 2004.
20. J. Medina, M. Ojeda-Aciego, and P. Vojtáš. Multi-adjoint logic programming with continuous semantics. In *Logic Programming and Non-Monotonic Reasoning, LPNMR'01*, pages 351–364. Lect. Notes in Artificial Intelligence 2173, 2001.
21. J. Medina, M. Ojeda-Aciego, and P. Vojtáš. Similarity-based unification: a multi-adjoint approach. *Fuzzy Sets and Systems*, 146:43–62, 2004.
22. A. D. Nola, E. Sanchez, W. Pedrycz, and S. Sessa. *Fuzzy Relation Equations and Their Applications to Knowledge Engineering*. Kluwer Academic Publishers, Norwell, MA, USA, 1989.
23. Z. Pawlak. Rough sets. *International Journal of Computer and Information Science*, 11:341–356, 1982.
24. W. Pedrycz. Fuzzy relational equations with generalized connectives and their applications. *Fuzzy Sets and Systems*, 10(1-3):185 – 201, 1983.
25. I. Perfilieva. Fuzzy function as an approximate solution to a system of fuzzy relation equations. *Fuzzy Sets and Systems*, 147(3):363–383, 2004.
26. I. Perfilieva and L. Nosková. System of fuzzy relation equations with inf-$\rightarrow$ composition: Complete set of solutions. *Fuzzy Sets and Systems*, 159(17):2256–2271, 2008.
27. A. M. Radzikowska and E. E. Kerre. A comparative study of fuzzy rough sets. *Fuzzy Sets and Systems*, 126(2):137–155, 2002.
28. E. Sanchez. Resolution of composite fuzzy relation equations. *Information and Control*, 30(1):38–48, 1976.
29. L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning I, II, III. *Information Sciences*, 8–9:199–257, 301–357, 43–80, 1975.

Adaptation knowledge discovery for cooking using closed itemset extraction

Emmanuelle Gaillard, Jean Lieber, and Emmanuel Nauer

LORIA (UMR 7503—CNRS, INRIA, Nancy University)
BP 239, 54506 Vandœuvre-lès-Nancy, France,
First-Name.Last-Name@loria.fr

Abstract. This paper is about the adaptation knowledge (AK) discovery for the TAAABLE system, a case-based reasoning system that adapts cooking recipes to user constraints. The AK comes from the interpretation of closed itemsets (CIs) whose items correspond to the ingredients that have to be removed, kept, or added. An original approach is proposed for building the context on which CI extraction is performed. This approach focuses on a restrictive selection of objects and on a specific ranking based on the form of the CIs. Several experimentations are proposed in order to improve the quality of the AK being extracted and to decrease the computation time. This chain of experiments can be seen as an iterative knowledge discovery process: the analysis following each experiment leads to a more sophisticated experiment until some concrete and useful results are obtained.

Keywords: adaptation knowledge discovery, closed itemset, data preprocessing, case-based reasoning, cooking.

1 Introduction

This paper addresses the adaptation challenge proposed by the *Computer Cooking Contest* (http://computercookingcontest.net/) which consists in adapting a given cooking recipe to specific constraints. For example, the user wants to adapt a strawberry pie recipe, because she has no strawberry. The underlying question is: which ingredient(s) will the strawberries be replaced with?

Adapting a recipe by substituting some ingredients by others requires cooking knowledge and adaptation knowledge in particular. TAAABLE, a case-based reasoning (CBR) system, addresses this problem using an ingredient ontology. This ontology is used for searching which is/are the closest ingredient(s) to the one that has to be replaced. In this approach the notion of "being close to" is given by the distance between ingredients in the ontology. In the previous example, TAAABLE proposes to replace the strawberries by other berries (e.g. raspberries, blueberries, etc.). However, this approach is limited because two ingredients which are close in the ontology are not necessarily interchangeable and because introducing a new ingredient in a recipe may be incompatible with some other ingredient(s) of the recipe or may required to add other ingredients.

This paper extends the approach proposed in [2] for extracting this kind of adaptation knowledge (AK). The approach is based on closed itemset (CI) extraction, in which items are the ingredients that have to be removed, kept, or added for adapting the recipe. This paper introduces two originalities. The first one concerns the way the binary context, on which the CI extraction is performed, is built, by focusing on a restrictive selection of objects according to the objectives of the knowledge discovery process. The second one concerns the way the CIs are filtered and ranked, according to their form. The paper is organised as follows: Section 2 specifies the problem in its whole context and introduces TAAABLE which will integrate the discovered AK in its reasoning process. Section 3 gives preliminaries for this work, introducing CI extraction, case-based reasoning, and related work. Section 4 explains our approach; several experiments and evaluations are described and discussed.

2 Context and motivations

2.1 Taaable

The *Computer Cooking Contest* is an international contest that aims at comparing systems that make inferences about cooking. A candidate system has to use the recipe base given by the contest to propose a recipe matching the user query. This query is a set of constraints such as inclusion or rejection of ingredients, the type or the origin of the dish, and the compatibility with some diets (vegetarian, nut-free, etc.).

TAAABLE [1] is a system that has been originally designed as a candidate of the *Computer Cooking Contest*. It is also used as a brain teaser for research in knowledge based systems, including knowledge discovery, ontology engineering, and CBR. Like many CBR systems, TAAABLE uses an ontology to retrieve recipes that are the most similar to the query. TAAABLE retrieves and creates cooking recipes by adaptation. If there exist recipes exactly matching the query, they are returned to the user; otherwise the system is able to retrieve similar recipes (i.e. recipes that partially match the target query) and adapts these recipes, creating new ones. Searching similar recipes is guided by several ontologies, i.e. hierarchies of classes (ingredient hierarchy, dish type hierarchy and dish origin hierarchy), in order to relax constraints by generalising the user query. The goal is to find the most specific generalisation of the query (the one with the minimal cost) for which recipes exist in the recipe base. Adaptation consists in substituting some ingredients of the retrieved recipes by the ones required by the query.

TAAABLE retrieves recipes using query generalisation, then adapts them by substitution. This section gives a simplified description of the TAAABLE system. For more details about the TAAABLE inference engine, see e.g. [1]. For example, for adapting the *"My Strawberry Pie"* recipe to the `no Strawberry` constraint, the system first generalises `Strawberry` into `Berry`, then specialises `Berry` into, say, `Raspberry`.

2.2 Domain ontology

An ontology $\mathcal{O}$ defines the main classes and relations relevant to cooking. $\mathcal{O}$ is a set of atomic classes organised into several hierarchies (ingredient, dish type, dish origin, etc.). Given two classes `B` and `A` of this ontology, `A` is subsumed by `B`, denoted by $\texttt{B} \sqsupseteq \texttt{A}$, if the set of instances of `A` is included in the set of instances of `B`. For instance, $\texttt{Berry} \sqsupseteq \texttt{Blueberry}$ and $\texttt{Berry} \sqsupseteq \texttt{Raspberry}$.

2.3 Taaable adaptation principle

Let R be a recipe and Q be a query such that R does not exactly match Q (otherwise, no adaptation would be needed). For example, $Q = \texttt{no Strawberry}$ and $R =$ *"My Strawberry Pie"*.The basic ontology-driven adaptation in Taaable follows the generalisation/specialisation principle explained hereafter (in a simplified way). First, R is generalised (in a minimal way) into $\Gamma(R)$ that matches Q. For example, Γ may be the substitution $\texttt{Strawberry} \rightsquigarrow \texttt{Berry}$. Second, $\Gamma(R)$ is specialised into $\Sigma(\Gamma(R))$ that still matches Q. For example, Σ is the substitution $\texttt{Berry} \rightsquigarrow \texttt{Raspberry}$ (the class `Berry` is too abstract for a recipe and must be made precise). This adaptation approach has at least two limits. First, the choice of Σ is at random: there is no reason to choose raspberries instead of blueberries, unless additional knowledge is given. Second, when such a substitution of ingredient is made, it may occur that some ingredients should be added or removed from R. These limits point out the usefulness of additional knowledge for adaptation.

3 Preliminaries

3.1 Itemset extraction

Itemset extraction is a set of data-mining methods for extracting regularities into data, by aggregating object items appearing together. Like FCA [8], itemset extraction algorithms start from a *formal context* K, defined by $K = (G, M, r)$, where G is a set of objects, M is a set of items, and r is the relation on $G \times M$ stating that an object is described by an item [8]. Table 1 shows an example of context, in which recipes are described by the ingredients they require: G is a set of 5 objects (recipes R, R_1, R_2, R_3, and R_4), M is a set of 7 items (ingredients `Sugar`, `Water`, `Strawberry`, etc.).

An *itemset* I is a set of items, and the *support* of I, $support(I)$, is the number of objects of the formal context having every item of I. I is frequent, with respect to a threshold σ, whenever $support(I) \geq \sigma$. I is closed if it has no proper superset J ($I \subsetneq J$) with the same support. For example, $\{\texttt{Sugar}, \texttt{Raspberry}\}$ is an itemset and $support(\{\texttt{Sugar}, \texttt{Raspberry}\}) = 2$ because 2 recipes require both `Sugar` and `Raspberry`. However, $\{\texttt{Sugar}, \texttt{Raspberry}\}$ is not a closed itemset, because $\{\texttt{Sugar}, \texttt{PieCrust}, \texttt{Raspberry}\}$ has the same support. Another, equivalent, definition of closed itemsets can be given on the basis of a closure operator $\cdot''$ defined as follows. Let I be an itemset and I' be the set of objects that have all the items

	Sugar	Water	Strawberry	PieCrust	Cornstarch	CoolWhip	Raspberry	Gelatin	Apple	AppleJuice	Cinnamon	PieShell
R	×	×	×	×	×	×						
R_1	×			×	×		×	×				
R_2	×	×		×			×					
R_3	×			×	×				×	×		
R_4	×	×							×		×	×

Table 1. Formal context representing ingredients used in recipes.

of I: $I' = \{x \in G \mid \forall i \in I, x\ r\ i\}$. In a dual way, let X be a set of objects and X' be the set of properties shared by all objects of X: $X' = \{i \in M \mid \forall x \in X, x\ r\ i\}$. This defines two operators: $\cdot' : I \in 2^M \mapsto I' \in 2^G$ and $\cdot' : X \in 2^G \mapsto X' \in 2^M$. These operators can be composed in an operator $\cdot'' : I \in 2^M \mapsto I'' \in 2^M$. An itemset I is said to be closed if it is a fixed point of $\cdot''$, i.e., $I'' = I$.

In the following, "CIs" stands for closed itemsets, and "FCIs" stands for frequent CIs. For $\sigma = 3$, the FCIs of this context are {`Sugar`, `PieCrust`}, {`Sugar`, `PieCrust`, `Cornstarch`}, {`Sugar`, `Water`}, {`Sugar`}, {`Water`}, {`PieCrust`}, and {`Cornstarch`}.

For the following experiments, the CHARM algorithm [12] that efficiently computes the FCIs is used thanks to CORON a software platform implementing a rich set of algorithmic methods for symbolic data mining [11].

3.2 Case-based reasoning

Case-based reasoning (CBR [10]) consists in answering queries with the help of previous experience units called cases. In TAAABLE, a case is a recipe and a query represents user constraints. In many systems, including TAAABLE, CBR consists in the retrieval of a case from the case base and in the adaptation of the retrieved case in an adapted case that solves the query. Retrieval in TAAABLE is performed by minimal generalisation of the query (cf. section 2.3). Adaptation can be a simple substitution (e.g., substitute strawberry with any berry) but it can be improved thanks to the use of some domain specific AK. This motivates the research on AK acquisition.

3.3 Related work

The AK may be acquired in various way. It may be collected from experts [6], it may be acquired using machine learning techniques [9], or be semi-automatic, using data-mining techniques and knowledge discovery principles [3,4].

This paper addresses automatic AK discovery. Previous works, such as the ones proposed by d'Aquin et al. with the KASIMIR project in the medical do-

main [5], and by Badra et al. in the context of a previous work on TAAABLE [2], are the foundations of our work.

KASIMIR is a CBR system applied to decision support for breast cancer treatment. In KASIMIR, a case is a treatment used for a given patient. The patient is described by characteristics (age, tumour size and location, etc.) and the treatment consists in applying medical instructions. In order to discover AK, cases that are *similar* to the target case are first selected. Then, FCIs are computed on the variations between the target case and the similar cases. FCIs matching a specific form are interpreted for generating AK [5].

Badra et al. use this approach to make cooking adaptations in TAAABLE [2]. Their work aims at comparing pairs of recipes depending on the ingredients they contain. A recipe R is represented by the set of its ingredients: $\texttt{Ingredients}(R)$. For example, the recipe *"My Strawberry Pie"* is represented by

$$\texttt{Ingredients}(\textit{"My Strawberry Pie"}) = \{\texttt{Sugar}, \texttt{Water}, \texttt{Strawberry}, \texttt{PieCrust}, \texttt{Cornstarch}, \texttt{CoolWhip}\}$$

Let (R, R') be a pair of recipes which is selected. According to [2], the representation of a pair is denoted by Δ, where Δ represents the variation of ingredients between R and R'. Each ingredient ing is marked by $-$, $=$, or $+$:

- $ing^- \in \Delta$ if $ing \in \texttt{Ingredients}(R)$ and $ing \notin \texttt{Ingredients}(R')$, meaning that ing appears in R but not in R'.
- $ing^+ \in \Delta$ if $ing \notin \texttt{Ingredients}(R)$ and $ing \in \texttt{Ingredients}(R')$, meaning that ing appears in R' but not in R.
- $ing^= \in \Delta$ if $ing \in \texttt{Ingredients}(R)$ and $ing \in \texttt{Ingredients}(R')$, meaning that ing appears both in R in R'.

Building a formal context about ingredient variations in cooking recipes. Suppose we want to compare the recipe R with the four recipes (R_1, R_2, R_3, R_4) given in Table 1. Variations between $R =$ *"My Strawberry Pie"* and a recipe R_i have the form $\bigwedge_j ing_{i,j}^{mark}$. For example:

$$\Delta_{R,R_1} = \texttt{Sugar}^= \wedge \texttt{Water}^- \wedge \texttt{Strawberry}^- \wedge \texttt{PieCrust}^= \wedge \texttt{Cornstarch}^= \wedge \texttt{CoolWhip}^- \wedge \texttt{Raspberry}^+ \wedge \texttt{Gelatin}^+ \quad (1)$$

According to these variations, a formal context $K = (G, M, I)$ can be built (cf. Table 2, for the running example):

- $G = \{\Delta_{R,R_i}\}_i$
- M is the set of ingredient variations: $M = \{ing_{i,j}^{mark}\}_{i,j}$. In particular, M contains all the conjuncts of Δ_{R,R_1} ($\texttt{Strawberry}^-$, etc., cf.(1)).
- $(g, m) \in I$, if $g \in G$, $m \in M$, and m is a conjunct of g, for example $(\Delta_{R,R_1}, \texttt{Strawberry}^-) \in I$.

	Sugar$^{=}$	Water$^{=}$	Water^{-}	Strawberry^{-}	PieCrust$^{=}$	PieCrust^{-}	Cornstarch$^{=}$	Cornstarch^{-}	CoolWhip^{-}	Raspberry^{+}	Gelatin^{+}	Apple^{+}	AppleJuice^{+}	Cinnamon^{+}	PieShell^{+}
Δ_{R,R_1}	×		×	×	×		×		×	×	×				
Δ_{R,R_2}	×	×		×	×			×	×	×					
Δ_{R,R_3}	×		×	×	×		×		×			×	×		
Δ_{R,R_4}	×	×		×		×		×	×			×		×	×

Table 2. Formal context for ingredient variations in pairs of recipes (R, R_j).

Interpretation. In the formal context, an ingredient marked with + (resp. −) is an ingredient that has to be added (resp. removed). An ingredient marked with = is an ingredient common to R and R_i.

4 Adaptation Knowledge discovery

AK discovery is based on the same scheme as knowledge discovery in databases (KDD [7]). The main steps of the KDD process are data preparation, data-mining, and interpretation of the extracted units of information. Data preparation relies on formatting data for being used by data-mining tools and on filtering operations for focusing on special subsets of objects and/or items, according to the objectives of KDD. Data-mining tools are applied for extracting regularities into the data. These regularities have then to be interpreted; filtering operations may also be performed on this step because of the (often) huge size of the data-mining results or of the noise included in these results. All the steps are guided by an analyst.

The objective of our work is the extraction of some AK useful for adapting a given recipe to a query. The work presented in the following focuses on filtering operations, in order to extract from a formal context encoding ingredient variations between pairs of recipes, the cooking adaptations. The database used as entry point of the process is the Recipe Source database (`http://www.recipesource.com/`) which contains 73795 cooking recipes. For the sake of simplicity, we consider in the following, the problem of adapting R by substituting one or several ingredient(s) with one or several ingredient(s) (but the approach can be generalised for removing more ingredients, and also be used for adding ingredient(s) in a recipe). Three experiments are presented; they address the same adaptation problem: adapting the $R =$ *"My Strawberry Pie"* recipe, with `Ingredients`(*"My Strawberry Pie"*) = {`Sugar`, `Water`, `Strawberry`, `PieCrust`, `Cornstarch`, `CoolWhip`}, to the query `no Strawberry`. In each experiment, a formal context about ingredient variations in recipes is built. Then, FCIs are extracted and filtered for proposing cooking adaptation. The two first

experiments focus on object filtering, selecting recipes which are more and more similar to the *"My Strawberry Pie"* recipe: the first experiment uses recipe from the same type (i.e. pie dish) as *"My Strawberry Pie"* instead of choosing recipes of any type; the second experiment focuses on a more precise filtering based on similarity between the *"My Strawberry Pie"* recipe and recipes used for generating the formal context on ingredient variations.

4.1 A first approach with closed itemsets

As introduced in [2], a formal context is defined, where objects are ordered pairs of recipes (R, R') and properties are ingredients marked with $+$, $=$, $-$ for representing the ingredient variations from R to R'. The formal context which is build is similar to the example given in Table 2. In each pair of recipes, the first element is the recipe $R =$*"My Strawberry Pie"* that must be adapted; the second element is a recipe of the same dish type as R which, moreover, does not contain the ingredient which has to be removed. In our example, it corresponds to *pie dish* recipes which do not contain *strawberry*. This formal context allows to build CIs which have to be interpreted in order to acquire adaptation rules.

Experiment. 3653 pie dish recipes that do not contain strawberry are found in the Recipe Source database. The formal context, with 3653 objects $\times$ 1355 items produces 107,837 CIs (no minimal support is used).

Analysis. Some interesting CIs can be found. For example, $\{\texttt{PieCrust}^-, \texttt{Strawberry}^-, \texttt{Cornstarch}^-, \texttt{CoolWhip}^-, \texttt{Water}^-, \texttt{Sugar}^-\}$ with support of 1657, contains all the ingredients of R with a $-$ mark, meaning that there are 1657 recipes which have no common ingredients with the R recipe. In the same way, $\{\texttt{PieCrust}^-, \texttt{Strawberry}^-, \texttt{Cornstarch}^-, \texttt{CoolWhip}^-, \texttt{Water}^-\}$ with support 2590, means that 2590 recipes share only the `Sugar` ingredient with R because the sugar is the sole ingredient of R which is not included in this CI. The same analysis can be done for $\{\texttt{PieCrust}^-, \texttt{Strawberry}^-, \texttt{Cornstarch}^-, \texttt{CoolWhip}^-, \texttt{Sugar}^-\}$ (support of 1900), for water, etc.

Conclusion. The CIs are too numerous for being presented to the analyst. Only 1996 of the 3653 pie dish without strawberry recipes share at least one ingredient with R. There are too many recipes without anything in common. A first filter can be used to limit the size of the formal context in number of objects.

4.2 Filtering recipes with at least one common ingredient

Experiment. The formal context, with 1996 objects $\times$ 813 items, produces 22,408 CIs (no minimal support is used), ranked by decreasing support.

Results. The top five FCIs are:

- $\{\texttt{Strawberry}^-\}$ with support of 1996;
- $\{\texttt{Strawberry}^-, \texttt{CoolWhip}^-\}$ with support of 1916;
- $\{\texttt{Strawberry}^-, \texttt{PieCrust}^-\}$ with support of 1757;
- $\{\texttt{Strawberry}^-, \texttt{PieCrust}^-, \texttt{CoolWhip}^-\}$ with support of 1679;
- $\{\texttt{Strawberry}^-, \texttt{Cornstarch}^-\}$ with support of 1631.

Analysis. Several observations can be made. The first FCI containing an ingredient marked by + ($\{\texttt{Strawberry}^-, \texttt{Egg}^+\}$, with support of 849) appears only at the 46th position. Moreover, there are 45 FCIs with one ingredient marked by + in the first 100 FCIs, and no FCI with more than one ingredient marked by +. A substituting ingredient *ing* can only be found in CIs containing ing^+ meaning that there exists a recipe containing *ing*, which is not in R. So, FCIs that do not contain the + mark cannot be used for finding a substitution proposition, and they are numerous in the first 100 ones, based on a support ranking (recall that it has been chosen not to consider adaptation by simply removing ingredient).

In the first 100 FCIs, there is only 15 FCIs containing both an ingredient marked by + and an ingredient marked by =. In a FCI I, the = mark on a ingredient *ing* means that *ing* is common to R and to recipe(s) involved by the creation of I. So, an ingredient marked by = guarantees a certain similarity (based on ingredients that are used) between the recipes R and R' compared by $\Delta_{R,R'}$. If a FCI I contains a potential substituting ingredient, marked by +, but does not contain any =, the risk for proposing a cooking adaptation from I is very high, because there is no common ingredient with R in the recipe the potential substituting ingredient comes from.

In the first 100 recipes, the only potential substituting ingredients (so, the ingredients marked by +) are egg, salt, and butter, which are not satisfactory from a cooking viewpoint for substituting the strawberries.

We have conducted similar experiments with other R and queries, and the same observations as above can be made.

Conclusion. From these observations, it can be concluded that the sole ranking based on support is not efficient to find relevant cooking adaptation rules, because the most frequent CIs do no contain potential substituting ingredients and, moreover, have no common ingredient with R.

4.3 Filtering and ranking CIs according to their forms

To extract realistic adaptation, CIs with a maximum of ingredients marked by = are searched. We consider that a substitution is acceptable, if 50% of ingredients of R are preserved and if the adaptation does not introduce too many ingredients; we also limit the number of ingredients introduced to 50% of the initial number of ingredients in R. For the experiment with the R = *"My Strawberry Pie"*, containing initially 6 ingredients, it means that at least 3 ingredients must be preserved and at most 3 ingredients can be added. In term of CIs, it corresponds to CIs containing at least 3 ingredients marked with = and at most 3 ingredients marked with +.

Experiment. Using this filter on CIs produced by the previous experiment reduces the number of CIs to 505. However, because some CIs are more relevant than others, they must be ranked according to several criteria. We use the following rules, given by priority order:

1. A CI must have a + in order to find a potential substituting ingredient.
2. A CI which has more = than another one is more relevant. This criterion promotes the pairs which have a largest set of common ingredient.
3. A CI which has less − than another one is more relevant. This criterion promotes adaptations which remove less ingredients.
4. A CI which has less + than another one is more relevant. This criterion promotes adaptations which add less ingredients.
5. If two CIs cannot be ranked according to the 4 criteria above, the CI the more frequent is considered to be the more relevant.

Results. The 5 first CIs ranked according to the previous criteria are:

- $\{\texttt{Water}^{=}, \texttt{Sugar}^{=}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{=}, \texttt{Salt}^{+}\}$ with support of 5;
- $\{\texttt{Water}^{=}, \texttt{Sugar}^{=}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{=}, \texttt{LemonJuice}^{+}\}$ with support of 4;
- $\{\texttt{Water}^{=}, \texttt{Sugar}^{=}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{=}, \texttt{LemonJuice}^{+}, \texttt{CreamCheese}^{+}\}$ with support of 2;
- $\{\texttt{Water}^{=}, \texttt{Sugar}^{=}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{=}, \texttt{LemonJuice}^{+}, \texttt{WhippingCream}^{+}\}$ with support of 2;
- $\{\texttt{Water}^{=}, \texttt{Sugar}^{=}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{=}, \texttt{LemonJuice}^{+}, \texttt{LemonPeel}^{+}\}$ with support of 2.

Analysis. One can observe that potential substituting ingredients take part of the first 5 CIs and each CIs preserve 4 (of 6) ingredients. The low supports of these CIs confirm that searching frequent CIs is not compatible with our need, which is to extract CIs with a specific form.

Conclusion. Ranking the CIs according to our particular criteria is more efficient than using a support based ranking. This kind of ranking can also be seen as a filter on CIs. However, this approach requires to compute all CIs because the support of interesting CIs is low.

4.4 More restrictive formal context building according to the form of interesting CIs

The computation time can be improved by applying a more restrictive selection of recipe pairs at the formal context building step, decreasing drastically the size of the formal context. Indeed, as the expected form of CIs is known, recipe pairs that cannot produce CIs of the expected form can be removed. This can also be seen as a selection of recipes that are similar enough to R. R' is considered

as enough similar to R if R' has a minimal threshold $\sigma^{=} = 50\%$ of ingredients in common with R (cf. (2)) and if R' has a maximal threshold $\sigma^{+} = 50\%$ of ingredients that are not used in R (cf. (3)). These two conditions expresses for $\Delta_{R,R'}$ the same similarities conditions considered in section 4.3 on CIs.

$$\frac{|\texttt{Ingredients}(R) \cap \texttt{Ingredients}(R')|}{|\texttt{Ingredients}(R)|} \geq \sigma^{=} \tag{2}$$

$$\frac{|\texttt{Ingredients}(R') \setminus \texttt{Ingredients}(R)|}{|\texttt{Ingredients}(R)|} \geq \sigma^{+} \tag{3}$$

Experiment. Among the 1996 pie dish recipes not containing `Strawberry`, only 20 recipes satisfy the two conditions. The formal context, with 20 objects × 40 items, produces only 21 CIs (no minimal support is used).

Results. The 5 first CIs, satisfying the form introduced in the previous section and ranked by decreasing support are:

- $\{\texttt{Water}^{=}, \texttt{Sugar}^{=}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{=}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{RedFoodColoring}^{+}, \texttt{Cherry}^{+}\}$ with support of 1;
- $\{\texttt{Water}^{=}, \texttt{Sugar}^{=}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{-}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{PieShell}+\}$ with support of 6;
- $\{\texttt{Water}^{=}, \texttt{Sugar}^{=}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{-}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{Raspberry}^{+}\}$ with support of 3;
- $\{\texttt{Water}^{=}, \texttt{Sugar}^{-}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{=}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{Apple}^{+}, \texttt{AppleJuice}^{+}\}$ with support of 3;
- $\{\texttt{Water}^{=}, \texttt{Sugar}^{=}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{-}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{Peach}^{+}, \texttt{PieShell}+\}$ with support of 2.

Analysis. According to these CIs the first potential substituting ingredients are: `RedFoodColoring`, `Cherry`, `PieShell`, `Raspberry`, `Apple`, and `Peach`. Each CI preserves 3 or 4 (of 6) ingredients to 6 and two CIs add 2 ingredients.

Conclusion. This approach reduces the computation time without reducing the result quality. Moreover, it gives the best potential adaptation in the first CIs.

4.5 From CIs to adaptation rules

As Taaable must propose a recipe adaptation, CIs containing potentially substituting ingredients must be transformed. Indeed, a CI does not represent a direct cooking adaptation. For example, the third CI of the last experiment contains $\texttt{Raspberry}^{+}$, simultaneously with $\texttt{CoolWhip}^{-}$ and $\texttt{PieCrust}^{-}$. Removing the pie crust (i.e. $\texttt{PieCrust}^{-}$) can look surprising for a pie dish, but one must keep in mind that a CI does not correspond to a real recipe, but to an abstraction of variations between R and a set of recipes. So, producing a complete adaptation requires to get back to the Δ_{R,R_i} for having all the variations of ingredient that will take part to the adaptation. For example, for

	Water$^{=}$	Sugar$^{=}$	Cornstarch$^{=}$	PieCrust^{-}	Strawberry^{-}	CoolWhip^{-}	Raspberry^{+}	PieShell^{+}	Gelatin^{+}	FoodColor^{+}	GCPieCrust^{+}
Δ_{R,R_1}	×	×	×	×	×	×	×		×		×
Δ_{R,R_2}	×	×	×	×	×	×	×	×		×	
Δ_{R,R_3}	×	×	×	×	×	×	×	×			

Table 3. Formal context for ingredient variations in pairs of recipes (R, R_j).

the CI $\{\texttt{Water}^{=}, \texttt{Sugar}^{=}, \texttt{Cornstarch}^{=}, \texttt{PieCrust}^{-}, \texttt{Strawberry}^{-}, \texttt{CoolWhip}^{-}, \texttt{Raspberry}^{+}\}$, the Δ_{R,R_i} (with $i \in [1;3]$) are the ones given by Table 3.

The adaptation rules extracted from these 3 recipe variations are:

- $\{\texttt{CoolWhip}, \texttt{PieCrust}, \texttt{Strawberry}\} \rightsquigarrow \{\texttt{Gelatin}, \texttt{GCPieCrust}, \texttt{Raspberry}\}$;
- $\{\texttt{CoolWhip}, \texttt{PieCrust}, \texttt{Strawberry}\} \rightsquigarrow \{\texttt{FoodColor}, \texttt{PieShell}, \texttt{Raspberry}\}$;
- $\{\texttt{CoolWhip}, \texttt{PieCrust}, \texttt{Strawberry}\} \rightsquigarrow \{\texttt{PieShell}, \texttt{Raspberry}\}$.

For R_2 and R_3, `PieShell` is added in replacement of `PieCrust`; in R_1, `GCPieCrust` plays the role of `PieCrust`. These three recipe variations propose to replace `Strawberry` by `Raspberry`. For R_1 (resp. R_2), `Gelatin` (resp. `FoodColor`) is also added. Finally, the three recipe variations propose to remove the `CoolWhip`.

Our approach guarantees the ingredient compatibility, with the assumption that the recipe base used for the adaptation rule extraction process contains only *good* recipes, i.e. recipes which do not contain ingredient incompatibility. Indeed, as adaptation rules are extracted from real recipes, the good combination of ingredients is preserved. So, when introducing a new ingredient ing_1 (marked by ing_1^+), removing another ingredient ing_2 (marked by ing_2^-) could be required. The reason is that there is no recipe, entailed in the creation of the CI from which the adaptation rules are extracted, using both ing_1 and ing_2. In the same way, adding a supplementary ingredient ing_3 (marked by ing_3^+) in addition of ing_1, is obtained from recipes which use both ing_1 and ing_3.

Applying FCA on these Δ_{R,R_i} produces the concept lattice presented in Fig. 1 in which the top node is the CI retained. This node can be seen as a generic cooking adaptation, and navigating into the lattice will conduct to more specific adaptation. The KDD loop is closed: after having (1) selected and formatting the data, (2) applying a data-mining CI extraction algorithm, and (3) interpreting the results, a new set of data is selected on which a data-mining –FCA– algorithm could then be applied.

We have chosen to return the adaptation rules generated from the 5 first CIs to the user. So, the system proposes results where `Strawberry` could be replaced (in addition of some other ingredient adding or removing) by "`RedFoodColoring` and `Cherry`", by `Raspberry` with optional `Gelatin` or `FoodColor`, by `Peach`

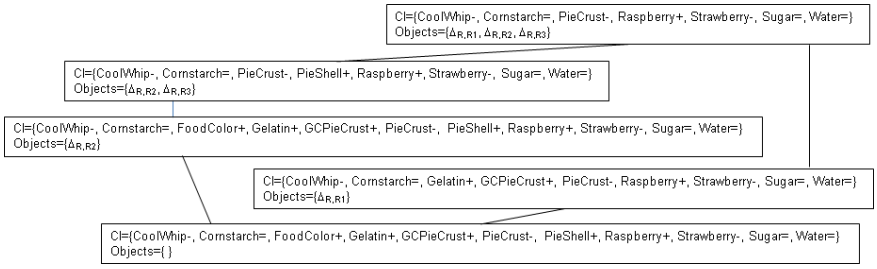


Fig. 1. The lattice computed on the formal context given in Table 3.

with optional `FoodColor` or `LemonJuice`, by "`HeavyCream` and `LemonRind`", or by "`Apple` and `AppleJuice`".

5 Conclusion

This paper shows how adaptation knowledge can be extracted efficiently for addressing a cooking adaptation challenge. Our approach focuses on CIs with a particular form, because the support is not a good ranking measure for this problem. A ranking method based on 5 criteria explicitly specified for this adaptation problem is proposed; the support is used in addition to distinguish CIs which satisfy in the same way the 5 criteria.

Beyond the application domain, this study points out that KD is not only a data-mining issue: the preparation and interpretation steps are also important. Moreover, it highlights the iterative nature of KD: starting from a first experiment with few a priori about the form of the results which are too numerous to be interpreted, it arrives to an experiment with a precise aim that gives results that are easy to interpret as adaptation rules.

It has been argued in the paper that this approach is better than the basic adaptation approach (based on substituting an ingredient by another one, on the basis of the ontology), in that it avoids some ingredient incompatibilities and makes some specialisation choices. However, a careful study remains to be made in order to compare experimentally these approaches.

A short-term future work is to integrate this AK discovery into the online system Taaable, following the principles of opportunistic KD [2].

A mid-term future work consists in using the ontology during the KD process. The idea is to add new items, deduced thanks to the ontology (e.g. the properties $\texttt{Cream}^-$ and $\texttt{Milk}^+$ entail the variation $\texttt{Dairy}^=$). First experiments have already been conducted but they raise interpretation difficulties. Indeed, the extracted CIs contain abstract terms (such as $\texttt{Dairy}^=$ or $\texttt{Flavoring}^+$) that are not easy to interpret.

References

1. F. Badra, R. Bendaoud, R. Bentebitel, P.-A. Champin, J. Cojan, A. Cordier, S. Després, S. Jean-Daubias, J. Lieber, T. Meilender, A. Mille, E. Nauer, A. Napoli, and Y. Toussaint. Taaable: Text Mining, Ontology Engineering, and Hierarchical Classification for Textual Case-Based Cooking. In *ECCBR Workshops, Workshop of the First Computer Cooking Contest*, pages 219–228, 2008.
2. F. Badra, A. Cordier, and J. Lieber. Opportunistic Adaptation Knowledge Discovery. In Lorraine McGinty and David C. Wilson, editors, *8th International Conference on Case-Based Reasoning - ICCBR 2009*, volume 5650 of *Lecture Notes in Computer Science*, pages 60–74, Seattle, États-Unis, July 2009. Springer. The original publication is available at www.springerlink.com.
3. S. Craw, N. Wiratunga, and R. C. Rowe. Learning adaptation knowledge to improve case-based reasoning. *Artificial Intelligence*, 170(16-17):1175–1192, 2006.
4. M. d'Aquin, F. Badra, S. Lafrogne, J. Lieber, A. Napoli, and L. Szathmary. Case base mining for adaptation knowledge acquisition. In *International Joint Conference on Artificial Intelligence, IJCAI'07*, pages 750–756, 2007.
5. M. D'Aquin, S. Brachais, J. Lieber, and A. Napoli. Decision Support and Knowledge Management in Oncology using Hierarchical Classification. In Katherina Kaiser, Silvia Miksch, and Samson W. Tu, editors, *Proceedings of the Symposium on Computerized Guidelines and Protocols - CGP-2004*, volume 101 of *Studies in Health Technology and Informatics*, pages 16–30, Prague, Czech Republic, 2004. Silvia Miksch and Samson W. Tu, IOS Press.
6. M. d'Aquin, J. Lieber, and A. Napoli. Adaptation Knowledge Acquisition: a Case Study for Case-Based Decision Support in Oncology. *Computational Intelligence (an International Journal)*, 22(3/4):161–176, 2006.
7. U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, pages 37–54, 1996.
8. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, 1999.
9. K. Hanney and M. T. Keane. Learning Adaptation Rules From a Case-Base. In I. Smith and B. Faltings, editors, *Advances in Case-Based Reasoning – Third European Workshop, EWCBR'96*, LNAI 1168, pages 179–192. Springer, 1996.
10. C. K. Riesbeck and R. C. Schank. *Inside Case-Based Reasoning*. Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey, 1989.
11. L. Szathmary and A. Napoli. CORON: A Framework for Levelwise Itemset Mining Algorithms. *Supplementary Proc. of The Third International Conference on Formal Concept Analysis (ICFCA '05), Lens, France*, pages 110–113, 2005.
12. M. J. Zaki and C.-J. Hsiao. CHARM: An efficient algorithm for closed itemset mining. In *SIAM International Conference on Data Mining SDM'02*, pages 33–43, 2002.

Fast Computation of Proper Premises

Uwe Ryssel[1], Felix Distel[2], and Daniel Borchmann[3]

[1] Institute of Applied Computer Science, Technische Universität Dresden, Dresden, Germany, uwe.ryssel@tu-dresden.de
[2] Institute of Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany, felix@tcs.inf.tu-dresden.de
[3] Institute of Algebra, Technische Universität Dresden, Dresden, Germany, borch@tcs.inf.tu-dresden.de

Abstract. This work is motivated by an application related to refactoring of model variants. In this application an implicational base needs to be computed, and runtime is more crucial than minimal cardinality. Since the usual stem base algorithms have proven to be too costly in terms of runtime, we have developed a new algorithm for the fast computation of proper premises. It is based on a known link between proper premises and minimal hypergraph transversals. Two further improvements are made, which reduce the number of proper premises that are obtained multiple times and redundancies within the set of proper premises. We provide heuristic evidence that an approach based on proper premises will also be beneficial for other applications.

1 Introduction

Today, graph-like structures are used in many model languages to specify algorithms or problems in a more readable way. Examples are data-flow-oriented simulation models, such as MATLAB/Simulink, state diagrams, and diagrams of electrical networks. Generally, such models consist of blocks or elements and connections among them. Using techniques described in Section 5.2, a formal context can be obtained from such models. By computing an implicational base of this context, dependencies among model artifacts can be uncovered. These can help to represent a large number of model variants in a structured way.

For many years, computing the stem base has been the default method for extracting a small but complete set of implications from a formal context. There exist mainly two algorithms to achieve this [10,15], and both of them compute not only the implications from the stem base, but also concept intents. This is problematic as a context may have exponentially many concept intents. Recent theoretical results suggest that existing approaches for computing the stem base may not lead to algorithms with better worst-case complexity [6,1].

Bearing this in mind, we focus on *proper premises*. Just like pseudo-intents, that are used to obtain the stem base, proper premises yield a sound and complete set of implications. Because this set of implications does not have minimal cardinality, proper premises have been outside the focus of the FCA community

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

for many years. However, there are substantial arguments to reconsider using them. Existing methods for computing proper premises avoid computing concept intents. Thus, in contexts with many concept intents they may have a clear advantage in runtime over the stem base algorithms. This is particularly true for our application where the number of concept intents is often close to the theoretical maximum. Here, attributes often occur together with their negated counterparts, and the concept lattice can contain several millions of elements. In Section 5.1 we provide arguments that we can expect the number of concept intents to be larger than the number of proper premises in most contexts, assuming a uniform random distribution.

Often, in applications, runtime is the limiting factor, not the size of the basis. But even where minimal cardinality is a requirement, computing proper premises is worth considering, since there are methods to transform a base into the stem base in polynomial time [16].

In this paper we present an algorithm for the fast computation of proper premises. It is based on three ideas. The first idea is to use a simple connection between proper premises and minimal hypergraph transversals. The problem of enumerating minimal hypergraph transversals is well-researched. Exploiting the link to proper premises allows us to use existing algorithms that are known to behave well in practice. A first, naïve algorithm iterates over all attributes and uses a black-box hypergraph algorithm to compute proper premises of each attribute.

A drawback when iterating over all attributes is that the same proper premise may be computed several times for different attributes. So we introduce a candidate filter in the second step: For each attribute m, the attribute set is filtered and proper premises are searched only among the candidate attributes. We show that this filtering method significantly reduces the number of multiple-computed proper premises while maintaining completeness. In a third step we exploit the fact that there are obvious redundancies within the proper premises. These can be removed by searching for proper premises only among the meet-irreducible attributes.

We argue that our algorithms are trivial to parallelize, leading to further speedups. Due to their incremental nature, parallelized versions of the stem base algorithms are not known to date. We conclude by providing experimental results. These show highly significant improvements for the contexts obtained from the model refactoring application. For a sample context, where Next-Closure required several hours to compute the stem base, runtime has dropped to fractions of a second. For contexts from other applications the improvements are not as impressive but still large.

2 Preliminaries

We provide a short summary of the most common definitions in formal concept analysis. A *formal context* is a triple $\mathbb{K} = (G, M, I)$ where G is a set of objects, M a set of attributes, and $I \subseteq G \times M$ is a relation that expresses whether an

object $g \in G$ has an attribute $m \in M$. If $A \subseteq G$ is a set of objects then A' denotes the set of all attributes that are shared among all objects in A, i.e., $A' = \{ m \in M \mid \forall g \in G \colon gIm \}$. Likewise, for some set $B \subseteq M$ we define $B' = \{ g \in G \mid \forall m \in B \colon gIm \}$. Pairs of the form (A, B) where $A' = B$ and $B' = A$ are called formal concepts. Formal concepts of the form $(\{ m \}', \{ m \}'')$ for some attribute $m \in M$ are called *attribute concept* and are denoted by μm. We define the partial order $\leq$ on the set of all formal concepts of a context to be the subset order on the first component. The first component of a formal concept is called the *concept extent* while the second component is called the *concept intent*.

Formal concept analysis provides methods to mine implicational knowledge from formal contexts. An *implication* is a pair (B_1, B_2) where $B_1, B_2 \subseteq M$, usually denoted by $B_1 \to B_2$. We say that *the implication* $B_1 \to B_2$ *holds* in a context $\mathbb{K}$ if $B_1' \subseteq B_2'$. An implication $B_1 \to B_2$ *follows from a set of implications* $\mathcal{L}$ if for every context $\mathbb{K}$ in which all implications from $\mathcal{L}$ hold, $B_1 \to B_2$ also holds. We say that $\mathcal{L}$ is *sound* for $\mathbb{K}$ if all implications from $\mathcal{L}$ hold in $\mathbb{K}$, and we say that $\mathcal{L}$ is *complete* for $\mathbb{K}$ if all implications that hold in $\mathbb{K}$ follow from $\mathcal{L}$. There exists a sound and complete set of implications for each context which has minimal cardinality [12]. This is called the stem base. The exact definition of the stem base is outside the scope of this work.

A sound and complete set of implications can also be obtained using *proper premises*. For a given set of attributes $B \subseteq M$, define $B^\bullet$ to be the set of those attributes in $M \setminus B$ that follow from B but not from a strict subset of B, i.e.,

$$B^\bullet = B'' \setminus \Big(B \cup \bigcup_{S \subsetneq B} S'' \Big).$$

B is called a *proper premise* if $B^\bullet$ is not empty. It is called a *proper premise for* $m \in M$ if $m \in B^\bullet$. It can be shown that $\mathcal{L} = \{ B \to B^\bullet \mid B \text{ proper premise} \}$ is sound and complete [11]. Several alternative ways to define this sound and complete set of implications can be found in [2].

We write $g \not\swarrow m$ if g' is maximal with respect to the subset order among all object intents which do not contain m.

3 Proper Premises as Minimal Hypergraph Transversals

We present a connection between proper premises and minimal hypergraph transversals, which forms the foundation for our enumeration algorithms. It has been exploited before in database theory to the purpose of mining functional dependencies from a database relation [14]. Implicitly, it has also been known for a long time within the FCA community. However, the term *hypergraph* has not been used in this context (cf. Prop. 23 from [11]).

Let V be a finite set of vertices. Then a *hypergraph on* V is simply a pair (V, H) where H is a subset of the power set 2^V. Intuitively, each set $E \in H$ represents an edge of the hypergraph, which, in contrast to classical graph theory,

may be incident to more or less than two vertices. A set $S \subseteq V$ is called a *hypergraph transversal* of H if it intersects every edge $E \in H$, i.e.,

$$\forall E \in H \colon S \cap E \neq \emptyset.$$

S is called a *minimal hypergraph transversal of* H if it is minimal with respect to the subset order among all hypergraph transversals of H. The *transversal hypergraph* of H is the set of all minimal hypergraph transversals of H. It is denoted by $Tr(H)$. The problem of deciding for two hypergraphs G and H whether H is the transversal hypergraph of G is called TransHyp. The problem of enumerating all minimal hypergraph transversals of a hypergraph G is called TransEnum. Both problems are relevant to a large number of fields and therefore have been well-researched. TransHyp is known to be contained in coNP. Since it has been shown that TransHyp can be decided in quasipolynomial time [9], it is not believed to be coNP-complete. Furthermore, it has been shown that it can be decided using only limited non-determinism [8]. For the enumeration problem it is not known to date whether an output-polynomial algorithm exists. However, efficient algorithms have been developed for several classes of hypergraphs [8,4].

The following proposition can be found in [11] among others.

Proposition 1. *$P \subseteq M$ is a premise of $m \in M$ iff*

$$(M \setminus g') \cap P \neq \emptyset$$

holds for all $g \in G$ with $g \not\downarrow m$. P is a proper premise for m iff P is minimal (with respect to $\subseteq$) with this property.

We immediately obtain the following corollary.

Corollary 1. *P is a premise of m iff P is a hypergraph transversal of (M, H) where*

$$H := \{M \setminus g' \mid g \in G, g \not\downarrow m\}.$$

The set of all proper premises of m is exactly the transversal hypergraph

$$Tr(\{M \setminus g' \mid g \in G, g \not\downarrow m\}).$$

In particular this proves that enumerating the proper premises of a given attribute m is polynomially equivalent to TransEnum. This can be exploited in a naïve algorithm for computing all proper premises of a formal context (Algorithm 1). Being aware of the link to hypergraph transversals, we can benefit from existing efficient algorithms for TransEnum in order to enumerate proper premises similar to what has been proposed in [14]. Of course, it is also possible to use other enumeration problems to which TransEnum can be reduced. Examples are the enumeration of prime implicants of Horn functions [2] and the enumeration of set covers.

4 Improvements to the Algorithm

4.1 Avoiding Duplicates using Candidate Sets

We can further optimize Algorithm 1 by reducing the search space. In the naïve algorithm proper premises are typically computed multiple times since they can be proper premises of more than one attribute. Our goal is to avoid this wherever possible.

The first idea is shown in Algorithm 2. There we introduce a *candidate set* C of particular attributes, depending on the current attribute m. We claim now that we only have to search for minimal hypergraph transversals P of $\{ M \setminus g' \mid g \not\downarrow m \}$ with $P \subseteq C$. We provide some intuition for this idea.

Algorithm 1 Naïve Algorithm for Enumerating All Proper Premises

Input: $\mathbb{K} = (G, M, I)$
$\mathcal{P} = \emptyset$
for all $m \in M$ **do**
  $\mathcal{P} = \mathcal{P} \cup Tr(\{M \setminus g' \mid g \in G, g \not\downarrow m\})$
end for
return $\mathcal{P}$

Algorithm 2 A Better Algorithm for Enumerating All Proper Premises

Input: $\mathbb{K} = (G, M, I)$
$\mathcal{P} = \{ \{ m \} \mid m \in M, \{ m \}$ is a proper premise of $\mathbb{K} \}$
for all $m \in M$ **do**
  $C = \{ u \in M \setminus \{ m \} \mid \nexists v \in M : \mu u \wedge \mu m \leq \mu v < \mu m \}$
  $\mathcal{P} = \mathcal{P} \cup \{ P \subseteq C \mid P$ minimal hypergraph transversal of $\{ M \setminus g' \mid g \not\downarrow m \} \}$
end for
return $\mathcal{P}$

Let us fix a formal context $\mathbb{K} = (G, M, I)$, choose $m \in M$ and let $P \subseteq M$ be a proper premise for m. Then we know that $m \in P''$, which is equivalent to

$$\bigwedge_{p \in P} \mu p \leq \mu m.$$

If we now find another attribute $n \in M \setminus \{ m \}$ with

$$\bigwedge_{p \in P} \mu p \leq \mu n < \mu m$$

it suffices to find the set P as a proper premise for n, because from $\mu n < \mu m$ we can already infer $m \in P''$. Conversely, if we search for all proper premises for m,

we only have to search for those who are not proper premises for attributes n with $\mu n < \mu m$. Now suppose that there exists an element $u \in P$ and an attribute $v \in M$ such that

$$\mu m \wedge \mu u \leq \mu v < \mu m. \tag{1}$$

Then we know

$$(\bigwedge_{p \in P} \mu p) \wedge \mu m = \bigwedge_{p \in P} \mu p \leq \mu v < \mu m,$$

i.e., P is already a proper premise for v. In this case, we do not have to search for P, since it will be found in another iteration. On the other hand, if P is a proper premise for m but not for any other attribute $n \in M$ with $\mu n < \mu m$, the argument given above shows that an element $u \in P$ and an attribute $v \in M$ satisfying (1) cannot exist.

Lemma 1. *Algorithm 2 enumerates for a given formal context $\mathbb{K} = (G, M, I)$ all proper premises of $\mathbb{K}$.*

Proof. Let P be a proper premise of $\mathbb{K}$ for the attribute m. P is a proper premise and therefore $m \in P''$ holds, which is equivalent to $\mu m \geq (P', P'')$. Let $c \in M$ be such that $\mu m \geq \mu c \geq (P', P'')$ and μc is minimal with this property. We claim that either $P = \{ c \}$ or P is found in the iteration for c of Algorithm 2.

Suppose $c \in P$. Then $m \in \{ c \}''$ follows from $\mu m \geq \mu c$. As a proper premise, P is minimal with the property $m \in P''$. It follows $P = \{ c \}$ and P is found by Algorithm 2 during the initialization.

Now suppose $c \notin P$. Consider

$$C := \{ u \in M \setminus \{ c \} \mid \nexists v \in M \colon \mu u \wedge \mu c \leq \mu v < \mu c \}.$$

We shall show $P \subseteq C$. To see this, consider some $p \in P$. Then $p \neq c$ holds by assumption. Suppose that $p \notin C$, i.e., there is some $v \in M$ such that $\mu p \wedge \mu c \leq \mu v < \mu c$. Because of $p \in P$, $\mu p \geq (P', P'')$ and together with $\mu c \geq (P', P'')$ we have

$$(P', P'') \leq \mu p \wedge \mu c \leq \mu v < \mu c$$

in contradiction to the minimality of μc. This shows $p \in C$ and all together $P \subseteq C$.

To complete the proof it remains to show that P is a minimal hypergraph transversal of $\{ M \setminus \{ g \}' \mid g \not\downarrow c \}$, i.e., that P is also a proper premise for c, not only for m. Consider $n \in P$. Assume $c \in (P \setminus \{ n \})''$. Since $\{c\}$ implies m, then $P \setminus \{ n \}$ would be a premise for m in contradiction to the minimality of P. Thus $c \notin (P \setminus \{ n \})''$ holds for all $n \in P$ and therefore P is a proper premise for c.

4.2 Irreducible Attributes

We go one step further and also remove attributes m from our candidate set C whose attribute concept μm is the meet of other attribute concepts $\mu x_1, \ldots, \mu x_n$, where $x_1, \ldots, x_n \in C$, i.e., $\mu m = \bigwedge_{i=1}^{n} \mu x_i$. This results in Algorithm 3 that no

longer computes all proper premises, but a subset that still yields a complete implicational base. We show that we only have to search for proper premises P with $P \subseteq N$ where N is the set of irreducible attributes of $\mathbb{K}$.

To ease the presentation, let us assume for the rest of this paper that the formal context $\mathbb{K}$ is attribute-clarified.

Algorithm 3 Computing Enough Proper Premises

Input: $\mathbb{K} = (G, M, I)$
$\mathcal{P} = \{ \{ m \} \mid m \in M, \{ m \}$ is a proper premise of $\mathbb{K} \}$
$N = M \setminus \{ x \in M \mid \mu x = \bigwedge_{i=1}^{n} \mu x_i$ for an $n \in \mathbb{N}$ and $x_i \in M$ for $1 \leq i \leq n \}$
for all $m \in M$ **do**
  $C = \{ u \in N \setminus \{ m \} \mid \nexists v \in M : \mu u \wedge \mu m \leq \mu v < \mu m \}$
  $\mathcal{P} = \mathcal{P} \cup \{ P \subseteq C \mid P$ minimal hypergraph transversal of $\{ M \setminus g' \mid g \not\supset m \} \}$
end for
return $\mathcal{P}$

Proposition 2. *Let m be an attribute and let P be a proper premise for m. Let $x \in P$, $n \in \mathbb{N}$, and for $1 \leq i \leq n$ let $x_i \in M$ be attributes satisfying*

- $m \notin \{ x_1, \ldots, x_n \}$,
- $\mu x = \bigwedge_{i=1}^{n} \mu x_i$,
- $x_i \notin \emptyset''$ *for all* $1 \leq i \leq n$ *and*
- $\mu x < \mu x_i$ *for all* $1 \leq i \leq n$.

Then $\{ x \}$ is a proper premise for all x_i and there exists a nonempty set $Y \subseteq \{ x_1, \ldots, x_n \}$ such that $(P \setminus \{ x \}) \cup Y$ is a proper premise for m.

Proof. It is clear that $\{ x \}$ is a proper premise for all x_i, since $x_i \in \{ x \}''$ and $x_i \notin \emptyset''$. Define

$$Q_Y := (P \setminus \{ x \}) \cup Y$$

for $Y \subseteq \{ x_1, \ldots, x_n \}$. We choose $Y \subseteq \{ x_1, \ldots, x_n \}$ such that Y is minimal with respect to $m \in Q_Y''$. Such a set exists, since $m \in ((P \setminus \{ x \}) \cup \{ x_1, \ldots, x_n \})''$ because of $\{ x_1, \ldots, x_n \} \to \{ x \}$. Furthermore, $Y \neq \emptyset$, since $m \notin (P \setminus \{ x \})''$.

We now claim that Q_Y is a proper premise for m. Clearly $m \notin Q_Y$, since $m \notin Y$. For all $y \in Y$ it holds that $m \notin (Q_Y \setminus \{ y \})''$ or otherwise minimality of Y would be violated. It therefore remains to show that $m \notin (Q_Y \setminus \{ y \})''$ for all $y \in Q_Y \setminus Y = P \setminus \{ x \}$.

$$\begin{aligned}(Q_Y \setminus \{ y \})'' &= ((P \setminus \{ x, y \}) \cup Y)'' \\ &\subseteq ((P \setminus \{ y \}) \cup Y)'' \\ &= (P \setminus \{ y \})''\end{aligned}$$

since $\{ x \} \to Y$ and $x \in P \setminus \{ y \}$. Since $m \notin (P \setminus \{ y \})''$, we get $m \notin (Q_Y \setminus \{ y \})''$ as required. In sum, Q_Y is a proper premise for m.

Lemma 2. *Let N be the set of all meet-irreducible attributes of a context $\mathbb{K}$. Define*

$$\mathcal{P} = \{\, X \subseteq M \mid |X| \leq 1, X \text{ proper premise} \,\} \cup \{\, X \subseteq N \mid X \text{ proper premise} \,\}$$

Then the set $\mathcal{L} = \{\, P \to P^\bullet \mid P \in \mathcal{P} \,\}$ is sound and complete for $\mathbb{K}$.

Proof. Let m be an attribute and let P be a proper premise for m. If $P \notin \mathcal{P}$ then it follows that $P \not\subseteq N$. Thus we can find $y_1 \in P \setminus N$ and elements $x_1, \ldots, x_n \in M$ with $n \geq 1$ such that

- $m \notin \{\, x_1, \ldots, x_n \,\}$,
- $\mu y_1 = \bigwedge_{i=1}^{n} \mu x_i$,
- $x_i \notin \emptyset''$ for all $1 \leq i \leq n$ and
- $\mu x < \mu x_i$ for all $1 \leq i \leq n$.

By Proposition 2 we can find a proper premise P_1 such that $P \to \{\, m \,\}$ follows from $\{\, y_1 \,\} \to \{\, x_1, \ldots, x_n \,\}$ and $P_1 \to \{\, m \,\}$. Clearly $\{\, y_1 \,\} \in \mathcal{P}$, since all singleton proper premises are contained in $\mathcal{P}$. If $P_1 \notin \mathcal{P}$ then we can apply Proposition 2 again and obtain a new proper premise P_2, etc. To see that this process terminates consider the strict partial order $\prec$ defined as

$$P \prec Q \text{ iff } \forall q \in Q \colon \exists p \in P \colon \mu p < \mu q.$$

It is easy to see that with each application of Proposition 2 we obtain a new proper premise that is strictly larger than the previous with respect to $\prec$. Hence, the process must terminate. This yields a set $\mathcal{P}' = \{\, \{\, y_1 \,\}, \ldots, \{\, y_k \,\}, P_k \,\} \subseteq \mathcal{P}$ such that $P \to \{\, m \,\}$ follows from $\{\, Q \to Q^\bullet \mid Q \in \mathcal{P}' \,\}$. Thus $\mathcal{L}$ is a sound and complete set of implications.

Together with Lemma 1 this yields correctness of Algorithm 3.

Corollary 2. *The set of proper premises computed by Algorithm 3 yields a sound and complete set of implications for the given formal context.*

5 Evaluation

5.1 Computing Proper Premises Instead of Intents

In both the stem base algorithms and our algorithms, runtime can be exponential in the size of the input. In the classical case the reason is that the number of intents can be exponential in the size of the stem base [13]. In the case of our algorithms there are two reasons: the computation of proper premises is TransEnum-complete, and there can be exponentially many proper premises. The first issue is less relevant in practice because algorithms for TransEnum, while still exponential in the worst case, behave well for most instances.

To see that there can be exponentially many proper premises in the size of the stem base, let us look at the context $\mathbb{K}_n$ from Table 1 for some $n \geq 2$, consisting

of two contranominal scales of dimension $n \times n$ and one attribute a with empty extent. It can be verified that the proper premises of the attribute a are exactly the sets of the form $\{m_i \mid i \in I\} \cup \{m'_i \mid i \notin I\}$ for some $I \subseteq \{1, \ldots, n\}$, while the only pseudo-intents are the singleton sets and $\{m_1, \ldots, m_n, m'_1, \ldots, m'_n\}$. Hence there are 2^n proper premises for a, while there are only $2n + 2$ pseudo-intents.

Table 1. Context $\mathbb{K}_n$ with Exponentially Many Proper Premises

	$m_1 \ \ldots \ m_n$	$m'_1 \ \ldots \ m'_n$	a
g_1 $\vdots$ g_n	$I_{\neq}$	$I_{\neq}$	

Next-Closure behaves poorly on contexts with many intents while our algorithms behave poorly on contexts with many proper premises. In order to provide evidence that our algorithm should behave better in practice we use formulae for the expectation of the number of intents and proper premises in a formal context that is chosen uniformly at random among all $n \times m$-contexts for fixed natural numbers n and m.[4] Derivations of these formulae can be found in [7].

The expected value for the number of intents in an $n \times m$-context is

$$\mathbb{E}_{\text{intent}} = \sum_{q=0}^{m} \binom{m}{q} \sum_{r=0}^{n} \binom{n}{r} 2^{-rq} (1 - 2^{-r})^{m-q} (1 - 2^{-q})^{n-r},$$

while the expected value for the number of proper premises for a fixed attribute a in an $n \times m$-context is

$$\mathbb{E}_{\text{pp}} = 2^{-n} \sum_{r=0}^{n} \binom{n}{r} \sum_{q=0}^{m-1} \binom{m}{q} q! \, 2^{-q^2} \sum_{\substack{(p_1,\ldots,p_q) \in \mathbb{N}^q \\ 1 \leq p_1 < \cdots < p_q \leq r}} \prod_{i=0}^{q} \left(1 - 2^{-q}(1+i)\right)^{p_{i+1} - p_i - 1}.$$

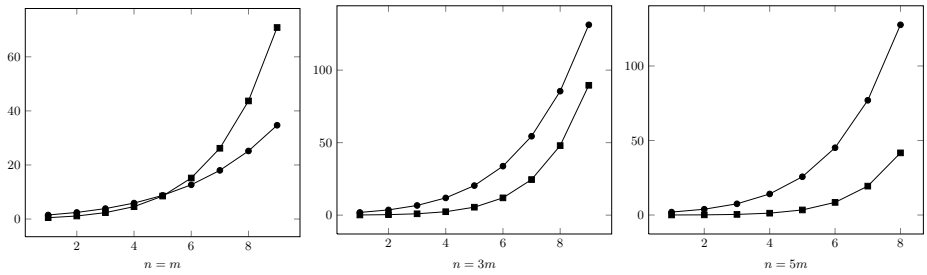
Figure 1 shows the values of $m \cdot \mathbb{E}_{\text{pp}}$ (squares) and $\mathbb{E}_{\text{intent}}$ (bullets) for quadratic contexts and for contexts with $n = 3m$ and $n = 5m$. While there are more proper premises for quadratic contexts, less proper premises need to be computed for contexts with a large number of objects.

5.2 Applications in Model Refactoring

In the Section 5.3 we shall see that the proper premise approach performs extremely well on contexts obtained from model refactoring. We briefly describe how these contexts are obtained. Working with data-flow-oriented simulation

[4] We ignore renaming of attributes and objects.

Fig. 1. Expected Number of Intents and Proper Premises for Certain Families of Formal Contexts



models will result fast in many variants of similar models, which are difficult to manage. One solution for that problem is to refactor these variants to configurable models, containing all parts and information about the valid combinations. Using a generator, the single variants, which are merged, can be created later on demand.

Using matching algorithms, corresponding artifacts (i.e., blocks, connections, and parameter settings) among the variants can be found. This results in a minimal set of artifacts, from that each variant can be built. To restrict the combinations, and with it the number of variants that can be generated, to the given model variants, we have to specify the dependencies among the artifacts. These can be defined by a set of implications, which can be calculated by the formal concept analysis. The whole process is described in [17,18].

Using the matching result, a formal context can be built: The model variants form the object set and the artifacts form the attribute set. The relation between variants and artifacts is defined by incidence. If a block or connection exists in a variant, they are related. The many-valued relation among variants and parameters is resolved by using parameter settings (i.e., pairs of parameters and their values) as attributes. The resulting implications should also contain negated literals, so we also need the negated counterparts of the artifact attributes. The semantic of the negated attributes is in this case a non-existing block, a non-existing connection, and a parameter that is not set to a certain value. In our application, only negated counterparts of irreducible attributes are needed.

This results in dense contexts (usually with a fill ratio greater than 0.3) with a typical size of 20 objects and 60 attributes, from which a complete set of implications has to be calculated.

5.3 Experimental Comparison to Other Algorithms

We experimentally compare our proposed algorithm to other well known ones. For this, we name the algorithms we want to compare, briefly discuss some implementation details, and then present the achieved results.

Algorithms We compare the following implementations: *SB* which is an implementation of Next-Closure, *HT* which computes all proper premises as hypergraph transversals as in Algorithm 1, and *PP*, an implementation of Algorithm 3.

Implementation An easy optimization we have made in HT and PP concerns parallelization. In all the listings we have presented so far, the iterations over the set of attributes in our formal context are independent of each other. It is natural to evaluate those iterations in parallel to improve the overall performance of our algorithms.

In our experiments we did not use special-purpose algorithms to compute the hypergraph transversals in HT and PP. Instead we have used an ad-hoc implementation that is based on backtracking and some simple heuristics [3]. Compared to advanced algorithms for TransEnum, this leaves room for further optimizations.

Data Sets Our test data sets can be divided into two categories, random and structured. For the tests on structured data we have chosen two data sets from the UCI Machine Learning Repository. The first data set is the testing data set of SPECT [5], which describes Single Proton Emission Computed Tomography (SPECT) images. This data set is given as a dyadic formal context with 187 objects, 23 attributes, and an approximate density of 0.38. The second one is a data set on Congressional Voting Records of the U.S. House of Representatives from 1984 [19]. It contains 435 objects, 16 attributes and is given as many valued context. It has been nominally scaled, resulting in a context with 50 attributes and an approximate density of 0.34.[5] The third structured data set originates from the application described in Section 5.2 and [18]. It has 26 objects, 79 attributes and an approximate density of 0.35.

The other part of testing data sets consist of randomly generated contexts. For this we fix the number n_G of objects, n_M of attributes and the density n_I of crosses. Then we generate for those three numbers one context of the given size $n_G \times n_M$ with an approximate density of n_I.

Experimental Results We have implemented all three algorithms as part of `conexp-clj`, a general-purpose FCA tool developed by one of the authors. The implementations itself are not highly optimized but rather prototypical, so the absolute running times of the test cases should not be taken as best possible. However, for comparing the three algorithms SB, HT, and PP, those implementations are sufficient and give a good impression on their performance. The experimental results (runtime and size of implication base) are given in Table 2 and Table 3.

As one can see from the results, HT most often runs faster then SB, but both are outperformed by PP. This can be seen most drastically with the Data-Flow-Model data sets, where PP only runs a fraction of a second whereas both

[5] Note that this is another scaling as the one in [15], so the times obtained there cannot be compared directly to ours.

Table 2. Behaviour on Structured Data

Context	SB		HT		PP	
Data-Flow-Model	6.5 hrs	86	37 hrs	1480	0.1 sec	86
SPECT	175 sec	1778	16 sec	6830	5.4 sec	6830
Voting	13 hrs	18,572	6 hrs	140,032	17 min	140,032

Table 3. Behaviour on Random Data

Context	SB		HT		PP	
$20 \times 40 \times 0.9$	75 sec	78	0.8 sec	988	1.4 sec	527
$20 \times 40 \times 0.8$	820 sec	879	4.3 sec	11263	2.4 sec	9223
$20 \times 40 \times 0.3$	8.9 sec	556	4.6 sec	3780	2.5 sec	3698
$20 \times 40 \times 0.2$	5.2 sec	386	2.2 sec	1817	0.9 sec	1478
$40 \times 20 \times 0.9$	17 sec	62	0.04 sec	105	0.04 sec	105
$40 \times 20 \times 0.8$	104 sec	920	1.5 sec	2017	0.45 sec	2017
$40 \times 20 \times 0.3$	1.6 sec	388	1 sec	1258	0.7 sec	1258
$40 \times 20 \times 0.2$	0.4 sec	173	0.4 sec	503	0.4 sec	503
$25 \times 25 \times 0.9$	17 sec	72	0.1 sec	154	0.04 sec	122
$25 \times 25 \times 0.8$	143 sec	565	1 sec	2533	0.4 sec	2533
$25 \times 25 \times 0.3$	1.2 sec	252	0.8 sec	1231	0.6 sec	1231
$25 \times 25 \times 0.2$	0.9 sec	226	0.34 sec	550	0.3 sec	533

SB and HT run for hours. The same occurs with the Voting data set. The same observation, although not that drastically, can also be seen with the randomly generated data sets.

The number of implications returned varies significantly not only between SB and HT/PP, but also between different runs of PP. Most often, HT and PP will return the same result, i.e., if the input context is attribute reduced. However, if it is not, the number of implications returned by PP may be significantly smaller than the overall number of proper premises, as one can see with the Data-Flow-Model data set, where the number of returned implications is the smallest possible.

However, most of the time the number of implications computed by HT and PP is much larger then the size of the stem base. The observed factors mostly range between 5 and 20. This might be a problem in practice, in particular if this factor is much higher. Therefore, one has to consider a tradeoff between the time one wants to spend on computing a sound and complete set of implications and on the size of this set of implications. The actual requirements of the particular application decide on the usefulness of the particular algorithm.

References

1. Mikhail Babin and Sergei Kuznetsov. Recognizing pseudo-intents is coNP-complete. In Marzena Kryszkiewicz and Sergei Obiedkov, editors, *Proc. of the*

7th Int. Conf. on Concept Lattices and Their Applications, volume 672. CEUR Workshop Proceedings, 2010.
2. K. Bertet and B. Monjardet. The multiple facets of the canonical direct unit implicational basis. *Theoretical Computer Science*, 411(22–24):2155–2166, 2010.
3. Daniel Borchmann. conexp-clj. `http://www.math.tu-dresden.de/~borch/conexp-clj/`.
4. E. Boros, K. Elbassioni, V. Gurvich, and L. Khachiyan. An efficient incremental algorithm for generating all maximal independent sets in hypergraphs of bounded dimension. *Parallel Processing Letters*, 10:253–266, 2000.
5. Krzysztof J. Cios, Lukasz A. Kurgan, and Lucy S. Goodenday. UCI Machine Learning Repository: SPECT Heart Data Set, 2001.
6. Felix Distel. Hardness of enumerating pseudo-intents in the lectic order. In Barış Sertkaya and Léonard Kwuida, editors, *Proc. of the 8th Int. Conf. on Formal Concept Analysis*, volume 5986 of *Lecture Notes in Artificial Intelligence*, pages 124–137. Springer, 2010.
7. Felix Distel and Daniel Borchmann. Expected numbers of proper premises and concept intents. Preprint, Institut für Algebra, TU Dresden, 2011.
8. Thomas Eiter, Georg Gottlob, and Kazuhisa Makino. New results on monotone dualization and generating hypergraph transversals. *SIAM J. Comput.*, 32:514–537, February 2003.
9. Michael L. Fredman and Leonid Khachiyan. On the complexity of dualization of monotone disjunctive normal forms. *Journal of Algorithms*, 21(3):618 – 628, 1996.
10. Bernhard Ganter. Two basic algorithms in concept analysis. Preprint 831, Fachbereich Mathematik, TU Darmstadt, Darmstadt, Germany, 1984.
11. Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, New York, 1997.
12. 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.
13. Sergei O. Kuznetsov. On the intractability of computing the Duquenne-Guigues base. *Journal of Universal Computer Science*, 10(8):927–933, 2004.
14. Heikki Mannila and Kari-Jouko Räihä. Algorithms for inferring functional dependencies from relations. *Data & Knowledge Engineering*, 12(1):83 – 99, 1994.
15. Sergei Obiedkov and Vincent Duquenne. Attribute-incremental construction of the canonical implication basis. *Annals of Mathematics and Artificial Intelligence*, 49(1-4):77–99, 2007.
16. Sebastian Rudolph. Some notes on pseudo-closed sets. In Sergei O. Kuznetsov and Stefan Schmidt, editors, *Proc. of the 5th Int. Conf. on Formal Concept Analysis*, volume 4390 of *Lecture Notes in Computer Science*, pages 151–165. Springer, 2007.
17. Uwe Ryssel, Joern Ploennigs, and Klaus Kabitzsch. Automatic variation-point identification in function-block-based models. In *Proc. of the 9th Int. Conf. on Generative Programming and Component Engineering*, pages 23–32. ACM, 2010.
18. Uwe Ryssel, Joern Ploennigs, and Klaus Kabitzsch. Extraction of feature models from formal contexts. In *Proc. of the 15th Int. Software Product Line Conference*, pages 4:1–4:8. ACM, 2011.
19. Jeff Schlimmer. UCI Machine Learning Repository: 1984 United Stated congressional voting records, 1987.

Block relations in fuzzy setting*

Jan Konecny, Michal Krupka

Department of Computer Science
Palacký University in Olomouc
17. listopadu 12, CZ-77146 Olomouc
Czech Republic
jan.konecny@upol.cz, michal.krupka@upol.cz

Abstract. One of the main problems in FCA (especially in fuzzy setting) is to reduce a concept lattice to appropriate size to make it graspable and understandable. We generalize known results on the correspondence of block relations of formal contexts and complete tolerances on concept lattices to fuzzy setting. Using an illustrative example we show that the results can be used for reduction of fuzzy concept lattices.

Keywords: Fuzzy concept lattice, Tolerance, Block relation, Factorization

1 Introduction

The problem of the size of concept lattices is recognized as one of the most important problems of Formal Concept Analysis (FCA). The main aim of this paper is to describe approximation of formal concepts by fuzzy tolerances in fuzzy setting.

In [3, 2], a method of factorization of fuzzy concept lattices is presented. A similarity threshold a is supplied and the method outputs a factor lattice instead of the whole concept lattice which might be large. The elements of the factor lattice are maximal blocks of concepts from the whole concept lattice which are pairwise similar to degree at least a. We generalize the results to a more general family of similarities (complete fuzzy tolerances) than those induced by a single threshold. We also show that the factor lattice can be computed from a special kind of superrelations of the incidence relation, called fuzzy block relations; and the relationship between complete fuzzy tolerances and fuzzy block relations. That way we generalize a Wille's results on (crisp) tolerances and block relations in [17] (see also [7]).

In [13], Meschke proposed a method of approximation of (crisp) concepts by tolerances and block relations. We use some of his ideas (namely the selection of important objects and attributes) in an illustrative example at the end of this paper.

The organization of the paper is as follows. In Section 2 we recall the fundamental notions and results on the basic fuzzy structures used in the paper. Section 3 is devoted to study of fuzzy tolerance relations: we define complete tolerance relations on completely lattice ordered fuzzy sets and show fundamental properties of the corresponding factor sets. In Section 4 we describe fuzzy block relations and prove the main result

* Supported by grant no. P103/10/1056 of the Czech Science Foundation.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

of the paper: the correspondence between fuzzy block relations and factor structures of concept lattices. Section 5 contains an illustrative example.

Due to space limitations, we abbreviate, resp. omit, some proofs.

2 Preliminaries

We recall basic notions and facts of residuated lattices, fuzzy sets and fuzzy relations, fuzzy Galois connections and fuzzy concept lattices.

2.1 Residuated lattices and fuzzy sets

We use complete residuated lattices as basic structures of truth degrees. A complete residuated lattice [3, 9, 16] 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., $a \otimes b \leq c$ iff $a \leq b \rightarrow c$.

0 and 1 denote the least and greatest elements. The partial order of $\mathbf{L}$ is denoted by $\leq$. Throughout the paper, $\mathbf{L}$ denotes an arbitrary complete residuated lattice.

Elements a of L are called truth degrees. $\otimes$ and $\rightarrow$ (truth functions of) "fuzzy conjunction" and "fuzzy implication".

Common examples of complete residuated lattices include those defined on a unit interval, (i.e., $L = [0, 1]$) or on a finite chain in the unit interval, e.g. $L = \{0, \frac{1}{n}, \ldots, \frac{n-1}{n}, 1\}$, $\wedge$ and $\vee$ being minimum and maximum, $\otimes$ being a left-continuous t-norm with the corresponding $\rightarrow$.

The three most important pairs of adjoint operations on the unit interval are

Łukasiewicz:
$$a \otimes b = \max(a + b - 1, 0)$$
$$a \rightarrow b = \min(1 - a + b, 1)$$

Gödel:
$$a \otimes b = \min(a, b)$$
$$a \rightarrow b = \begin{cases} 1 & a \leq b \\ b & \text{otherwise} \end{cases}$$

Goguen (product):
$$a \otimes b = a \cdot b$$
$$a \rightarrow b = \begin{cases} 1 & a \leq b \\ \frac{b}{a} & \text{otherwise} \end{cases}$$

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 the support of a complete residuated lattice. The set of all $\mathbf{L}$-sets in a universe X is denoted L^X.

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 $\{{}^{A(x)}/x \,|\, x \in X\}$. If for all $y \in X$ distinct from $x_1, x_2, \ldots, x_n$ we have $A(y) = 0$, we also write $\{{}^{A(x_1)}/x_1, {}^{A(x_2)}/x_1, \ldots, {}^{A(x_n)}/x_n\}$.

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). An **L**-set $A \in L^X$ is called crisp if $A(x) \in \{0,1\}$ for each $x \in X$. Crisp **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 **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 $A \in L^X$, $a \otimes A \in L^X$ and $a \to A \in L^X$ are defined by

$$(a \otimes A)(x) = a \otimes A(x) \text{ and } (a \to A)(x) = a \to A(x).$$

For an **L**-set $A \in L^X$ and $a \in L$, the *a-cut of A* is a crisp subset ${}^aA \subseteq X$ such that $x \in {}^aA$ iff $a \leq A(x)$. This definition applies also to binary **L**-relations, whose a-cuts are interpreted as classical (crisp) binary relations.

For universe X we define **L**-relation *graded subsethood* $L^X \times L^X \to L$ by:

$$S(A,B) = \bigwedge_{x\in X} A(x) \to B(x) \tag{1}$$

Graded subsethood generalizes the classical subsethood relation $\subseteq$; indeed, in the crisp case (i.e. $L = \{0,1\}$) (1) becomes $S(A,B) = 1$ iff $\forall x \in X : x \in A$ implies $y \in B$. Note that unlike $\subseteq$, S is a binary **L**-relation on L^X. Described verbally, $S(A,B)$ represents a degree to which A is a subset of B. In particular, we write $A \subseteq B$ iff $S(A,B) = 1$. As a consequence, we have $A \subseteq B$ iff $A(x) \leq B(x)$ for each $x \in X$.

Further we set

$$A \approx^X B = S(A,B) \wedge S(B,A). \tag{2}$$

The value $A \approx^X B$ is interpreted as the degree to which the sets A and B are similar.

A binary **L**-relation R on a set X is called *reflexive* if $R(x,x) = 1$ for any $x \in X$, *symmetric* if $R(x,y) = R(y,x)$ for any $x,y \in X$, and *transitive* if $R(x,y) \otimes R(y,z) \leq R(x,z)$ for any $x,y,z \in X$. R is called an **L**-*tolerance*, if it is reflexive and symmetric, **L**-*equivalence* if it is reflexive, symmetric and transitive. If R is an **L**-equivalence such that for any $x,y \in X$ from $R(x,y) = 1$ it follows $x = y$, then R is called an **L**-*equality* on X. **L**-equalities are often denoted by $\approx$. The similarity $\approx^X$ of **L**-sets (2) is an **L**-equality on L^X.

Let $\sim$ be an **L**-equivalence on X, $A \in L^X$ an **L**-set. We say that A is *compatible with* $\sim$ (or *extensional with respect to* $\sim$) if the following condition holds for any $x,x' \in X$:

$$A(x) \otimes (x \sim x') \leq A(x').$$

For each $A \in L^X$ there exists a minimal (with respect to inclusion of **L**-sets) **L**-set $C_\approx(A) \in L^X$, compatible with $\sim$, and such that $A \subseteq C_\sim(A)$. It holds

$$C_\sim(A)(x) = \bigvee_{x'\in X} A(x') \otimes (x' \sim x). \tag{3}$$

We say that a binary **L**-relation R on X is *compatible with* $\sim$ if for each $x,x',y,y' \in X$,

$$R(x,y) \otimes (x \sim x') \otimes (y \sim y') \leq R(x',y').$$

For a mapping $f: X \to Y$, where Y is endowed with an $\mathbf{L}$-equality $\approx$, and $\mathbf{L}$-set $A \in L^X$ we define the *image of A with respect to f* as an $\mathbf{L}$-set $f(A) \in L^Y$ satisfying

$$f(A)(y) = \bigvee_{x \in X} A(x) \otimes (f(x) \approx y).$$

$f(A)$ is compatible with $\approx$ and $f(A) = C_{\approx}(B)$, where B is is an $\mathbf{L}$-set in Y given by $B(y) = \bigvee_{x \in f^{-1}(\{y\})} A(x)$.

In the following we use well-known properties of residuated lattices and fuzzy structures which can be found e.g. in [3, 9].

2.2 L-ordered sets

In this section, we recall basic definitions and results of the theory of $\mathbf{L}$-ordered sets. Basic references are [1, 3] and the references therein.

An $\mathbf{L}$-*order* on a set U with an $\mathbf{L}$-equality relation $\approx$ is a binary $\mathbf{L}$-relation $\preceq$ on U which is compatible with $\approx$, reflexive, transitive and satisfies $(u \preceq v) \wedge (v \preceq u) \leq u \approx v$ for any $u, v \in U$ (*antisymmetry*). The tuple $\mathbf{U} = \langle \langle U, \approx \rangle, \preceq \rangle$ is called an $\mathbf{L}$-*ordered set*. An immediate consequence of the definition is that for any $u, v \in U$ it holds

$$u \approx v = (u \preceq v) \wedge (v \preceq u). \tag{4}$$

If $\mathbf{U} = \langle \langle U, \approx \rangle, \preceq \rangle$ is an $\mathbf{L}$-ordered set, then the tuple $\langle U, {}^1\preceq \rangle$, where ${}^1\preceq$ is the 1-cut of $\preceq$, is a (partially) ordered set. We sometimes write $\leq$ instead of ${}^1\preceq$ and use the symbols $\wedge$, $\bigwedge$ resp. $\vee$, $\bigvee$ for denoting infima resp. suprema in $\langle U, {}^1\preceq \rangle$.

For two $\mathbf{L}$-ordered sets $\mathbf{U} = \langle \langle U, \approx_U \rangle, \preceq_U \rangle$ and $\mathbf{V} = \langle \langle V, \approx_V \rangle, \preceq_V \rangle$, a mapping $f: U \to V$ is called an *isomorphism of* $\mathbf{U}$ *and* $\mathbf{V}$, if it is a bijection and $(u_1 \preceq_U u_2) = (f(u_1) \preceq_V f(u_2))$ for any $u_1, u_2 \in V$. $\mathbf{U}$ and $\mathbf{V}$ are then called *isomorphic*.

For an $\mathbf{L}$-ordered set $\mathbf{U}$ and an $\mathbf{L}$-set $V \in L^U$ we define $\mathbf{L}$-sets $\mathscr{U}V$, $\mathscr{L}V \in L^U$ by

$$\mathscr{L}V(u) = \bigwedge_{v \in U} V(v) \to (u \preceq v), \qquad \mathscr{U}V(u) = \bigwedge_{v \in U} V(v) \to (v \preceq u).$$

Using basic rules of fuzzy logic, $\mathscr{L}V$ (resp. $\mathscr{U}V$) can be interpreted as the set of elements which are smaller (resp. greater) than or equal to elements of V. $\mathscr{L}V$ (resp. $\mathscr{U}V$) is called *the lower cone* (resp. *the upper cone*) of U.

For any $\mathbf{L}$-set $V \in L^U$ there exists at most one element $u \in U$ such that $\mathscr{L}V(u) \wedge \mathscr{U}(\mathscr{L}V)(u) = 1$ (resp. $\mathscr{U}V \wedge \mathscr{L}(\mathscr{U}V)(u) = 1$) [1, 3]. If there is such an element, we call it *the infimum of* V (resp. *the supremum of* V) and denote $\inf V$ (resp. $\sup V$); otherwise we say, that the infimum (resp. supremum) does not exist.

If $u, v \in U$, $v \leq u$, then the $\mathbf{L}$-set $[v, u] = \mathscr{U}\{v\} \cap \mathscr{L}\{u\}$ is called an *interval* in $\mathbf{U}$. It holds $\inf[v, u] = v$, $\sup[v, u] = u$.

An $\mathbf{L}$-ordered set $\mathbf{U}$ is called *completely lattice* $\mathbf{L}$-*ordered*, if for each $V \in L^U$, both $\inf V$ and $\sup V$ exist.

Let $\mathbf{U}$ be an $\mathbf{L}$-ordered set. For $a \in L$ and $u \in U$ we set $a \to u = \inf\{a/u\}$ and $a \otimes u = \sup\{a/u\}$. $a \to u$ is called *the a-shift of u*, $a \otimes u$ is called *the a-multiple of u*. $\mathbf{U}$ is called *shift complete* (resp. *multiple complete*) if $a \to u$ (resp. $a \otimes u$) exists for any $a \in L$, $u \in U$.

The notions of shift and multiple coincide with the notions of *cotensor* and *tensor* [10, 15, 18] and shift-complete resp. multiple-complete **L**-ordered sets are cotensored resp. tensored Ω-categories from [15, 18]. In this paper, we use basic properties of shifts and multiples, which can be found in these papers and also in [12]. We summarize some of them below.

For any $u, v \in U$ and $a \in L$ it holds $v = a \to u$ iff for each $w \in U$,

$$(w \preceq v) = a \to (w \preceq u). \tag{5}$$

Similarly, $v = a \otimes u$ iff for each $w \in U$,

$$(v \preceq w) = a \to (u \preceq w). \tag{6}$$

Shifts and multiples satisfy the following adjointness condition: If $a \to v$ and $a \otimes u$ both exist then $(u \preceq (a \to v)) = ((a \otimes u) \preceq v)$.

The following theorem has been proved in [18] (Propositions 3.12, 3.13) and shows how shifts and multiples can be efficiently used for proving that an **L**-ordered set is completely lattice **L**-ordered.

Theorem 1. **V** *is a completely lattice* **L***-ordered set iff* **V** *is shift complete and multiple complete and* $\langle V, {}^1\preceq \rangle$ *is a complete lattice.*

2.3 Isotone L-Galois connections

We introduce the notion of isotone **L**-Galois connections. For details, see [8].

An *isotone* **L***-Galois connection* between **L**-ordered sets **U** and **V** is a pair $\langle f, g \rangle$, where $f : U \to V$, $g : V \to U$ are mappings such that for each $u \in U$, $v \in V$ it holds

$$(f(u) \preceq v) = (u \preceq g(v)). \tag{7}$$

A pair $\langle u, v \rangle$, where $u \in U$ and $v \in V$, is called a *fixpoint of* $\langle f, g \rangle$ if $f(u) = v$ and $g(v) = u$. More generally, *the degree to which* $\langle u, v \rangle$ *is a fixpoint of* $\langle f, g \rangle$ is defined by

$$\mathrm{Fix}_{\langle f,g \rangle}(\langle u, v \rangle) = (f(u) \approx v) \wedge (g(v) \approx u). \tag{8}$$

$\mathrm{Fix}_{\langle f,g \rangle}$ is an **L**-set in $U \times V$ and is called *the* **L***-set of fixpoints of* $\langle f, g \rangle$.

Theorem 2 (basic properties of isotone L-Galois connections). *Let* $\langle f, g \rangle$ *be an isotone* **L***-Galois connection between* **L***-ordered sets* **U** *and* **V**. *Then*

(a) $u \le g(f(u))$ *for each* $u \in U$, $f(g(v)) \le v$ *for each* $v \in V$.

(b) f *and* g *are increasing:* $(u_1 \preceq u_2) \le (f(u_1) \preceq f(u_2))$ *and* $(v_1 \preceq v_2) \le (g(v_1) \preceq g(v_2))$.

(c) $f(g(f(u))) = f(u)$, $g(f(g(v))) = g(v)$.

(d) If $U' \in L^U$ *then* $f(\inf U') \le \inf f(U')$ *and* $f(\sup U') \ge \sup f(U')$. *If* $V' \in L^V$ *then* $g(\inf V') \le \inf g(V')$ *and* $g(\sup V') \ge \sup g(V')$.

(e) If **U** *and* **V** *are completely lattice* **L***-ordered sets and* $K \in L^{U \times V}$, *then*

$$S(K, \mathrm{Fix}_{\langle f,g \rangle}) \le \mathrm{Fix}_{\langle f,g \rangle}(\langle \inf pr_U(K), f(g(\inf pr_V(K))) \rangle), \tag{9}$$

$$S(K, \mathrm{Fix}_{\langle f,g \rangle}) \le \mathrm{Fix}_{\langle f,g \rangle}(\langle g(f(\sup pr_U(K))), \sup pr_V(K) \rangle), \tag{10}$$

where $pr_U : U \times V \to U$ *and* $pr_V : U \times V \to V$ *are Cartesian projections.*

Proof. Omitted. Some of the properties are proved in [8].

For an **L**-ordered set **U**, an isotone **L**-Galois connection $\langle f, g \rangle$ between **U** and **U** is called *an inflationary Galois connection on* **U** if f is deflationary and g inflationary:

$$f(u) \leq u \quad \text{and} \quad g(u) \geq u \tag{11}$$

for each $u \in U$. Notice that if one of the conditions (11) holds true, then the second one follows from (7).

2.4 Composition Operators and Concept Lattices Associated to I

We use three relational composition operators, $\circ$, $\triangleleft$, and $\triangleright$. The composition operators are defined by

$$(A \circ B)(x,y) = \bigvee_{z \in Z} A(x,z) \otimes B(z,y), \tag{12}$$
$$(A \triangleleft B)(x,y) = \bigwedge_{z \in Z} A(x,z) \to B(z,y), \tag{13}$$
$$(A \triangleright B)(x,y) = \bigwedge_{z \in Z} B(z,y) \to A(x,z). \tag{14}$$

Note that these operators were extensively studied by Bandler and Kohout, see e.g. [11] to which we refer for an overview of knowledge processing applications. One may easily see that $\triangleright$ can be defined in terms of $\triangleleft$ and vice versa. They have natural verbal descriptions. For instance, $(A \circ B)(x,y)$ is the truth degree of the proposition "there is a factor z such that z applies to the object x and the attribute y is a manifestation of z"; $(A \triangleleft B)(x,y)$ is the truth degree of "for every factor z, if z applies to the object x then the attribute y is a manifestation of z". Note also that for $L = \{0,1\}$, $A \circ B$ coincides with the well-known composition of binary relations.

A *formal* **L**-*context* (or just context) is a triple $\langle X, Y, I \rangle$ where X and Y are sets whose elements are called objects and attributes, respectively and I is an L-relation between X and Y. Consider the following pairs of operators between L^X and L^Y induced by an **L**-relation $I \in L^{X \times Y}$:

$$A^{\uparrow}(y) = \bigwedge_{x \in X} A(x) \to I(x,y), \qquad B^{\downarrow}(x) = \bigwedge_{y \in Y} B(y) \to I(x,y), \tag{15}$$
$$A^{\cap}(y) = \bigvee_{x \in X} A(x) \otimes I(x,y), \qquad B^{\cup}(x) = \bigwedge_{y \in Y} I(x,y) \to B(y), \tag{16}$$
$$A^{\wedge}(y) = \bigwedge_{x \in X} I(x,y) \to A(x), \qquad B^{\vee}(x) = \bigvee_{y \in Y} B(y) \otimes I(x,y), \tag{17}$$

for $A \in L^X$, $B \in L^Y$. We call (15) antitone concept-forming operators and (16), (17) isotone concept-forming operators (these are less common if FCA). Furthemore, denote the set of fixpoints of $\langle {}^{\uparrow}, {}^{\downarrow} \rangle$ by $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$. We have

$$\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I) = \{\langle A, B \rangle \,|\, A^{\uparrow} = B, B^{\downarrow} = A\},$$

$\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$ is a completely lattice **L**-ordered set with the **L**-equality $\approx$ and **L**-order $\preceq$ defined by

$$\langle A_1, B_1 \rangle \approx \langle A_2, B_2 \rangle = (A_1 \approx^X A_2) \qquad (= B_1 \approx^Y B_2), \tag{18}$$
$$\langle A_1, B_1 \rangle \preceq \langle A_2, B_2 \rangle = S(A_1, A_2) \qquad (= S(B_2, B_1)), \tag{19}$$

and is called the **L**-*concept lattice associated to I*. Its elements are called *formal* **L**-*concepts*. **L**-concept lattices are the fundamental structures of formal concept analysis in fuzzy setting [7, 3]. For a formal **L**-concept $\langle A, B\rangle$, A and B are called the *extent* and the *intent* and they represent the collection of objects and attributes to which the formal concept applies. The sets of all extents and intents of the respective concept lattices are denoted by $\mathrm{Ext}(X^{\uparrow}, Y^{\downarrow}, I)$ and $\mathrm{Int}(X^{\uparrow}, Y^{\downarrow}, I)$, respectively. It may be shown that

$$\mathrm{Ext}(X^{\uparrow}, Y^{\downarrow}, I) = \{A \in L^X \mid A = A^{\uparrow\downarrow}\},$$
$$\mathrm{Int}(X^{\uparrow}, Y^{\downarrow}, I) = \{B \in L^Y \mid B = B^{\downarrow\uparrow}\}.$$

The above-defined operators and their sets of fixpoints have been extensively studied, see e.g. [1, 3, 8, 14].

Example 1. Let **L** be the 6-element Łukasiewicz chain (i.e., $L = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$) and $\langle X, Y, I\rangle$ be a formal **L**-context, where $X = \{\mathrm{BV}, \mathrm{LH}, \mathrm{MD}, \mathrm{TSS}\}$, $Y = \{\mathrm{c1}, \mathrm{c2}, \mathrm{c3}, \mathrm{c4}, \mathrm{c5}\}$, and I is depicted in Fig. 1. Elements of X are four selected movies by director David Lynch, elements of Y are five film critics and values of I are ratings the critics assigned to the movies, taken from `www.metacritic.com` and rescaled to the six-element scale. The context is our central example in this paper and we describe it deeper and work with it in Section 5. For now, we use it just to give an example of an **L**-concept lattice.

Note that our data are suitable for interpreting by means of fuzzy logic. A movie rating (usually given by number of "stars" or by a percentage) can be interpreted as an answer to the question: "Do you like this movie?", given in degrees. In most cases, it is not possible to answer the above question with simple "Yes" or "No", and, in the same time, there are no doubts about the answer (especially, if given by a film critic).

	c1	c2	c3	c4	c5
BV	0.4	0.4	0.2	0.4	0.6
LH	0.8	0.8	0.8	1	1
MD	0.8	0.4	1	0.8	1
TSS	0.4	0.4	0.8	0.6	0.4

Fig. 1. Formal context of movies ("Blue Velvet" (BV), "Lost Highway" (LH), "Mulholland Drive" (MD), and "The Straight Story" (TSS)), reviewers (David Sterritt (c1), Desson Thomson (c2), Jonathan Rosenbaum (c3), Owen Gleiberman (c4) and Roger Ebert (c5)) and their ratings on the 6-element Łukasievicz chain $L = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$. Data taken from `www.metacritic.com` on March 20, 2011.

The **L**-concept lattice $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$ is depicted in Fig. 2. When displaying **L**-concept lattices, we use labeled Hasse diagrams to include all the information on extents and intents. For any $x \in X$, $y \in Y$ and formal **L**-concept $\langle A, B\rangle$ we have $A(x) \geq a$ and $B(y) \geq b$ if and only if there is a formal concept $\langle A_1, B_1\rangle \leq \langle A, B\rangle$, labeled by ${}^{a}/x$ and a formal concept $\langle A_2, B_2\rangle \geq \langle A, B\rangle$, labeled by ${}^{b}/y$. We use labels x resp. y instead of ${}^{1}/x$ resp. ${}^{1}/y$ and omit redundant labels (i.e., if a concept has both the labels ${}^{a}/x$ and ${}^{b}/x$ then we keep only that with the greater degree; dually for attributes). The whole structure of **L**-ordered set on $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$ can be determined from the labeled diagram using the results from [1] (see also [3]).

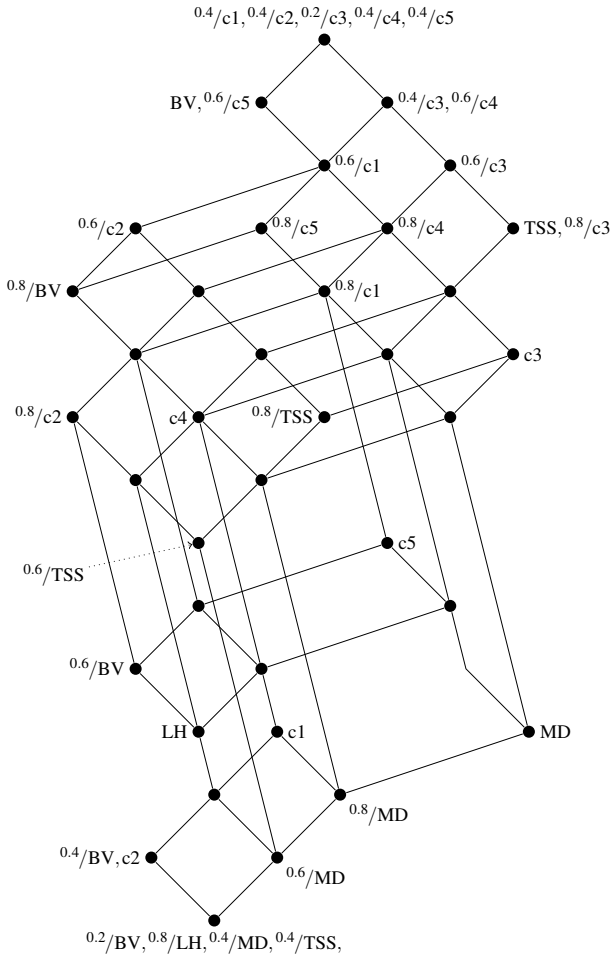


Fig. 2. **L**-concept lattice of movies, critics and ratings (from Fig. 1).

3 Complete L-tolerances

In this section we develop a generalization of the known results on factorization of complete lattices by complete tolerances [6, 17, 7] to fuzzy setting. We give a definition of a complete **L**-tolerance on a completely lattice **L**-ordered set, introduce a structure of an **L**-ordered set on the corresponding factor set and show that together with this structure, the factor set is a completely lattice **L**-ordered set (Theorem 7). In addition, we show that complete **L**-tolerances on a completely lattice **L**-ordered set are in one-to-one correspondence with inflationary **L**-Galois connections (Theorem 8).

For an **L**-tolerance $\sim$ on a set X, an **L**-set $B \in L^X$ is called a *block of the* **L**-*tolerance* $\sim$ if for each $x_1, x_2 \in X$ it holds $B(x_1) \otimes B(x_2) \leq (x_1 \sim x_2)$. A block B is called *maximal*

if for each block B' from $B \subseteq B'$ it follows $B = B'$. The set of all maximal blocks of $\sim$ always exists by Zorn's lemma, is called *the factor set of X by $\sim$* and denoted by $X/\sim$.

Further we set for each $x \in X$, $[\![x]\!]_\sim(y) = x \sim y$, obtaining an $\mathbf{L}$-set $[\![x]\!]_\sim$ called *the class of $\sim$ determined by x*.

An $\mathbf{L}$-tolerance $\sim$ on a completely lattice $\mathbf{L}$-ordered set $\mathbf{U} = \langle\langle U, \approx\rangle, \preceq\rangle$ is called *complete* if it is compatible with $\approx$ and for each $R \subseteq \sim$ it holds

$$\sup(pr_1(R)) \sim \sup(pr_2(R)) = \inf(pr_1(R)) \sim \inf(pr_2(R)) = 1, \tag{20}$$

where pr_1 (resp. pr_2) is the projection $U \times U \to U$ to the first (resp. second) component.

Note that in the classical case the condition (20) becomes the well-known condition of completeness of $\sim$ [17, 7]: Any subset $R \subseteq \sim$ is a set of pairs $\{\langle u_j, v_j\rangle \mid j \in J\}$ such that for each $j \in J$ it holds $u_j \sim v_j$ and $\sup(pr_1(R)) = \bigvee_{j\in J} u_j$, $\sup(pr_2(R)) = \bigvee_{j\in J} v_j$ and similarly for infima.

For the rest of this section we suppose $\sim$ is a complete $\mathbf{L}$-tolerance on a completely lattice $\mathbf{L}$-ordered set $\mathbf{U} = \langle\langle U, \approx\rangle, \preceq\rangle$.

Lemma 1. *If $u \sim v = 1$ then $(a \otimes u) \sim (a \otimes v) = 1$ and $(a \to u) \sim (a \to v) = 1$.*

Proof. Follows directly from the definition.

Lemma 2. *If $\sim$ is a complete $\mathbf{L}$-tolerance on a complete lattice $\mathbf{L}$-ordered set $\mathbf{U} = \langle\langle U, \approx\rangle, \preceq\rangle$ then so is $a \to \sim$, for each $a \in L$.*

Proof. Since $\sim$ is compatible with $\approx$ then for $u, u', v, v' \in U$ we have $(u \approx u') \otimes (a \to (u' \sim v')) \otimes (v' \approx v) \leq a \to ((u \approx u') \otimes (u' \sim v') \otimes (v' \approx v)) \leq a \to (u \sim v)$, proving $a \to \sim$ is compatible with $\approx$.

Let $R \subseteq a \to \sim$. We have for each $u, v \in U$, $(uRv) \leq a \to (u \sim v)$, which is equivalent to $a \otimes (uRv) \leq (u \sim v)$. Thus, $a \otimes R \subseteq \sim$ and we can use (20). We have $\sup(pr_1(a \otimes R)) = \sup(a \otimes pr_1(R)) = a \otimes \sup(pr_1(R))$, $\inf(pr_1(a \otimes R)) = \inf(a \otimes pr_1(R)) = a \to \inf(pr_1(R))$ and similarly for pr_2. Now the result follows from Lemma 1.

Theorem 3. *If $\sim$ is a complete $\mathbf{L}$-tolerance on $\mathbf{U}$, then for each $a \in L$ the a-cut ${}^a\!\sim$ is a complete tolerance on the complete lattice $\langle U, {}^1\!\preceq\rangle$.*

Proof. For $a = 1$ the assertion follows easily from the definition of complete $\mathbf{L}$-tolerances. The rest follows from Lemma 2.

We set for any $u \in U$,

$$u^\sim = \sup[\![u]\!]_\sim, \quad u_\sim = \inf[\![u]\!]_\sim. \tag{21}$$

Theorem 4. *The pair $\langle {}_\sim, {}^\sim\rangle$ is an inflationary isotone $\mathbf{L}$-Galois connection on $\mathbf{U}$.*

Proof. We show that for $u, v \in U$ it holds $(v_\sim \preceq u) \leq (v \preceq u^\sim)$, the proof of the opposite inequality is similar. Denote $(v_\sim \preceq u) = a$. Since $\mathscr{U}\{{}^a/v_\sim\}(u) = 1$ (by the definition of upper cone), then we have $a \otimes v_\sim \leq u$. Now, $(a \otimes v) \sim (a \otimes v_\sim) = 1$ (Lemma 1) and $u \sim (u \vee (a \otimes v)) = 1$. Thus, $u \vee (a \otimes v) \leq u^\sim$ which means that $(a \otimes v) \leq u^\sim$ or $v \preceq u^\sim \geq a$. The fact that $\langle {}_\sim, {}^\sim\rangle$ is inflationary is trivial.

Lemma 3. *Let $u,v \in U$ be such that $v \le u$, $u \sim v = 1$. If $w_1 \le u$, then $(v \preceq w_1) \le (u \sim w_1)$ and if $v \le w_2$, then $(w_2 \preceq u) \le (w_2 \sim v)$.*

Proof. By completeness of $\sim$, $(u \vee w_1) \sim (v \vee w_1) = 1$. Thus,

$$(v \preceq w_1) = (v \vee w_1) \approx w_1 = (u \sim (v \vee w_1)) \otimes ((v \vee w_1) \approx w_1) \le (u \sim w_1).$$

Similarly for w_2.

Theorem 5. *For each $u \in U$ the class $[\![u]\!]_\sim$ is equal to the interval $[u_\sim, u^\sim]$ in* **U**.

Proof. The inclusion $[\![u]\!]_\sim \subseteq [u_\sim, u^\sim]$ follows from basic properties of cones (for example, $[\![u]\!]_\sim \subseteq \mathscr{L}\mathscr{U}[\![u]\!]_\sim = \mathscr{L}\{u^\sim\}$).

We prove the opposite inclusion. For $w \in U$ denote $u_\sim \preceq w = a$ and $w \preceq u^\sim = b$. We have $[u_\sim, u^\sim](w) = a \wedge b$. By Lemma 3, $(u_\sim \preceq (w \wedge u)) \le ((w \wedge u) \sim u)$ and $((w \vee u) \preceq u^\sim) \le ((w \vee u) \sim u)$. But $u_\sim \preceq (w \wedge u) = a$ and $(w \vee u) \preceq u^\sim = b$ which means that $[u_\sim, u^\sim](w) = a \wedge b \le ((w \wedge u) \sim u) \wedge ((w \vee u) \sim u)$. Using Theorem 3 we easily obtain that the right-hand side is less than or equal to $w \sim u$.

Theorem 6. *Maximal blocks of $\sim$ are exactly* **L***-intervals $[v,u]$ where $\langle u, v\rangle$ are fixpoints of $\langle {}_\sim, {}^\sim\rangle$.*

Proof. Let $w_1, w_2 \in U$. We have

$$\begin{aligned}
[v,u](w_1) \otimes [v,u](w_2) &= ((v \preceq w_1) \wedge (w_1 \preceq u)) \otimes ((v \preceq w_2) \wedge (w_2 \preceq u)) \\
&\le ((v \preceq w_1) \otimes (w_2 \preceq u)) \wedge ((w_1 \preceq u) \otimes (v \preceq w_2)) \\
&= ((v \preceq w_1) \otimes (w_{2\sim} \preceq v)) \wedge ((w_1 \preceq u) \otimes (u \preceq w_2^\sim)) \\
&\le (w_{2\sim} \preceq w_1) \wedge (w_1 \preceq w_2^\sim) = [\![w_2]\!]_\sim(w_1) = (w_1 \sim w_2),
\end{aligned}$$

showing $[v,u]$ is a block.

Conversely, if $W \in L^U$ is a block then so is the interval $[\inf W, \sup W]$ which contains W as a subset. Among all intervals $[v,u]$, the maximal blocks are those satisfying $v^\sim = u$ and $u_\sim = v$.

We define two binary **L**-relations $\approx$ and $\preceq$ on $U/\!\sim$ by

$$[v_1,u_1] \approx [v_2,u_2] = v_1 \approx v_2 \quad (= u_1 \approx u_2), \tag{22}$$

$$[v_1,u_1] \preceq [v_2,u_2] = v_1 \preceq v_2 \quad (= u_1 \preceq u_2) \tag{23}$$

(the equalities in the parentheses follow directly from (4), Theorem 4, and Theorem 2 (b))

Theorem 7. *The tuple $\langle\langle U/\!\sim, \approx\rangle, \preceq\rangle$ is a completely lattice* **L***-ordered set.*

Proof. Sketch: we use Theorem 1, (5), (6).

Let $\langle f, g\rangle$ be an inflationary Galois connection on **U**. Set for each $u, v \in U$

$$u \sim_{\langle f,g\rangle} v = (f(u) \preceq v) \wedge (v \preceq g(u)). \tag{24}$$

Theorem 8. *$\sim_{\langle f,g\rangle}$ is an* **L***-complete tolerance, satisfying $u_{\sim_{\langle f,g\rangle}} = f(u)$, $u^{\sim\langle f,g\rangle} = g(u)$, for each $u \in U$. The assignment $\langle f,g\rangle \mapsto \sim_{\langle f,g\rangle}$ is a bijection between the set of all inflationary isotone* **L***-Galois connections* **U** *and the set of all* **L***-complete tolerances on* **U**.

Proof. Sketch: We use basic properties of isotone **L**-Galois connections (Theorem 2) and the definition of an **L**-complete tolerance on **U**.

4 Block L-relations

This section introduces block **L**-relations on **L**-formal contexts, their properties and the relationship to complete **L**-tolerances on **L**-concept lattices. The main results are contained in Theorems 11 and 12, where we show that block **L**-relations of any formal **L**-context are in one-to-one correspondence with complete **L**-tolerances on the associated **L**-concept lattice and that the **L**-concept lattice of each of the block relations is isomorphic to the original **L**-concept lattice, factorized by the corresponding complete **L**-tolerance.

We define block relations as follows: *Block* **L***-relation* of $I \in L^{X\times Y}$ is an **L**-relation $J \supseteq I$ such that $\mathrm{Ext}(X^{\uparrow},Y^{\downarrow},J) \subseteq \mathrm{Ext}(X^{\uparrow},Y^{\downarrow},I)$ and $\mathrm{Int}(X^{\uparrow},Y^{\downarrow},J) \subseteq \mathrm{Int}(X^{\uparrow},Y^{\downarrow},I)$.

In crisp setting, [17] defines block relation as a relation $J \supseteq I$ where each row is an intent of I and each column is an extent of I. Lemma 4 says that block **L**-relation is a proper generalization of crisp block relation. The reason we define the notion that way is to allow an analogous definition in the isotone case.

Lemma 4. *$J \in L^{X\times Y}$, $J \supseteq I$ is a block* **L***-relation of I iff $\{x\}^{\uparrow_I} \in \mathrm{Int}(X^{\uparrow},Y^{\downarrow},I)$ for each $x \in X$ and $\{y\}^{\downarrow_I} \in \mathrm{Ext}(X^{\uparrow},Y^{\downarrow},I)$ for each $y \in Y$.*

The following theorem provide characterization of block **L**-relations.

Theorem 9. *Let $I \in L^{X\times Y}$ be an* **L***-relation. The following statements are equivalent:*

(a) J is a block relation of I.

(b) $J = I \triangleright S_{\mathrm{i}}$ with $S_{\mathrm{i}} \in L^{Y\times Y}$ and for the induced mapping $^{\wedge S_{\mathrm{i}}}$ we have $B^{\wedge S_{\mathrm{i}}} \in \mathrm{Int}(X^{\uparrow},Y^{\downarrow},I)$ and $B \subseteq B^{\wedge S_{\mathrm{i}}}$ for each $B \in \mathrm{Int}(X^{\uparrow},Y^{\downarrow},I)$.

(c) $J = S_{\mathrm{e}} \triangleleft I$ with $S_{\mathrm{e}} \in L^{X\times X}$ and for the induced mapping $^{\cup S_{\mathrm{e}}}$ we have $A^{\cup S_{\mathrm{e}}} \in \mathrm{Ext}(X^{\uparrow},Y^{\downarrow},I)$ and $A \subseteq A^{\cup S_{\mathrm{e}}}$ for each $A \in \mathrm{Ext}(X^{\uparrow},Y^{\downarrow},I)$.

Proof. (sketch, equivalence of (a) and (b)): We have $\mathrm{Ext}(X^{\uparrow},Y^{\downarrow},J) \subseteq \mathrm{Ext}(X^{\uparrow},Y^{\downarrow},I)$ iff there exists a matrix S_{i} such that $J = I \triangleright S_{\mathrm{i}}$ by [4, Theorem 7]. We have $A^{\uparrow_J} = A^{\uparrow_{I\triangleright S_{\mathrm{i}}}} = (A^{\uparrow_I})^{\wedge S_{\mathrm{i}}}$ for $A \in L^X$. Since $A^{\uparrow_I}$ is any intent B in $\mathrm{Int}(X^{\uparrow},Y^{\downarrow},I)$ and $A^{\uparrow_J} = B^{\wedge S_{\mathrm{i}}} \in \mathrm{Int}(X^{\uparrow},Y^{\downarrow},I)$ we obtain $B^{\wedge S_{\mathrm{i}}} \in \mathrm{Int}(X^{\uparrow},Y^{\downarrow},I)$ for each $B \in \mathrm{Int}(X^{\uparrow},Y^{\downarrow},I)$. By (16) and (14) we have that $J(x,y)$ is equal to $\{x\}^{\uparrow_I \wedge S_{\mathrm{i}}}(y)$ and with the property $B \subseteq B^{\wedge S_{\mathrm{i}}}$ we get $I \subseteq J$ proving that J is a block **L**-relation.

Theorem 10. *Let $I \in L^{X\times Y}$ be an* **L***-relation between X and Y.*

(a) The set of all block **L***-relations J of I is an* **L***-closure system (i.e. it is closed under $\bigwedge$ and $\to$).*

(b) The set of all S_{e} (from Theorem 9) is an **L***-interior system (i.e. it is closed under $\bigvee$ and $\otimes$).*

(c) The set of all S_{i} (from Theorem 9) is an **L***-interior system.*

Proof. (a) Let J_i be block $\mathbf{L}$-relations of I. Let $J = \bigwedge_i J_i$ and let $B \in \mathrm{Int}(X^{\uparrow}, Y^{\downarrow}, J)$, hence $B = A^{\uparrow J}$ for some $A \in L^X$. By definition of $^{\uparrow J}$ and properties of residuated lattices we have

$$A^{\uparrow J}(y) = \bigwedge_{x \in X} A(x) \to J(x,y) = \bigwedge_{x \in X} A(x) \to \bigwedge_i J_i(x,y) = \bigwedge_i \bigwedge_{x \in X} A(x) \to J_i(x,y) = \bigwedge_i A^{\uparrow J_i}.$$

Thus we have $A^{\uparrow J} = \bigcap_i A^{\uparrow J_i} \in \mathrm{Int}(X^{\uparrow}, Y^{\downarrow}, I)$. Similarly, $B^{\downarrow J} = \bigcap_i B^{\downarrow J_i} \in \mathrm{Ext}(X^{\uparrow}, Y^{\downarrow}, I)$, $A^{\uparrow a \to J_i} = a \to A^{\uparrow J_i} \in \mathrm{Int}(X^{\uparrow}, Y^{\downarrow}, I)$, and $B^{\downarrow a \to J_i} = \bigcap_i a \to B^{\downarrow J_i} \in \mathrm{Ext}(X^{\uparrow}, Y^{\downarrow}, I)$.

Finally, from $I \subseteq J_i$ we have $I \subseteq \bigwedge_i J_i$ and $I \subseteq a \to J_i$. Whence $\bigwedge_i J_i$ and $I \subseteq a \to J_i$ are block $\mathbf{L}$-relations proving that set of all block $\mathbf{L}$-relations J of I is an $\mathbf{L}$-closure system.

(b) and (c) can be proved similarly.

The induced operators $^{\cap}, ^{\cup}$ of the matrices S_e and S_i have following properties:

Lemma 5. *Let $I \in L^{X \times Y}$ be an $\mathbf{L}$-relation between X and Y and let J be its block $\mathbf{L}$-relation:*

(a) $A^{\cup S_\mathrm{e}} = A^{\uparrow I \downarrow J}$ for any $A \in \mathrm{Ext}(X^{\uparrow}, Y^{\downarrow}, I)$, $B^{\wedge S_\mathrm{i}} = B^{\downarrow I \uparrow J}$ for any $B \in \mathrm{Int}(X^{\uparrow}, Y^{\downarrow}, I)$.

(b) $\langle A^{\cup S_\mathrm{e}}, A^{\cup S_\mathrm{e} \cap S_\mathrm{e} \uparrow I} \rangle \in \mathscr{B}(X^{\uparrow}, Y^{\downarrow}, J)$ and $\langle B^{\wedge S_\mathrm{i} \vee S_\mathrm{i} \downarrow I}, B^{\wedge S_\mathrm{i}} \rangle \in \mathscr{B}(X^{\uparrow}, Y^{\downarrow}, J)$ for each $\langle A, B \rangle \in \mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$.

(c) $A^{\cap S_\mathrm{e} \uparrow I} = A^{\uparrow I \wedge S_\mathrm{i}} = A^{\uparrow J}$ for any $A \in L^X$ and $B^{\vee S_\mathrm{i} \downarrow I} = B^{\downarrow I \cup S_\mathrm{e}} = B^{\downarrow J}$ for any $B \in L^Y$.

Let $I \in L^{X \times Y}$ be an $\mathbf{L}$-relation between X and Y and let J be its block relation. Denote by $^{\theta_J}$ a mapping $^{\theta_J} : \mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I) \to \mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$ defined by

$$\langle A, B \rangle^{\theta_J} = \langle A^{\cup S_\mathrm{e}}, A^{\cup S_\mathrm{e} \uparrow I} \rangle \tag{25}$$

and denote by ${}_{\theta_J}$ a mapping ${}_{\theta_J} : \mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I) \to \mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$ defined by

$$\langle A, B \rangle_{\theta_J} = \langle B^{\wedge S_\mathrm{i} \downarrow I}, B^{\wedge S_\mathrm{i}} \rangle \tag{26}$$

Notice, that different block $\mathbf{L}$-relations J of an $\mathbf{L}$-relation I induce different mappings $^{\theta_J}$, ${}_{\theta_J}$. In what follows we omit subscript in the notation $^{\theta_J}$, ${}_{\theta_J}$ and write just $^{\theta}$, ${}_{\theta}$. Obviously, for each $\langle A, B \rangle \in \mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$ we have $\langle A, B \rangle_{\theta} \leq \langle A, B \rangle \leq \langle A, B \rangle^{\theta}$.

Now, we explain the relationship between block $\mathbf{L}$-relations and compatible $\mathbf{L}$-tolerances.

Theorem 11. *(a) Let $J \supseteq I$ be a block $\mathbf{L}$-relation. Then the mappings $^{\theta}, {}_{\theta}$ form an inflationary $\mathbf{L}$-Galois connection on $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$.*

(b) For any inflationary $\mathbf{L}$-Galois connection $\langle f, g \rangle$ on $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$ the $\mathbf{L}$-relation $J \in L^{X \times Y}$ given by

$$\begin{aligned} J(x,y) &= f(\langle \{x\}^{\uparrow I \downarrow I}, \{x\}^{\uparrow I} \rangle) \preceq \langle \{y\}^{\downarrow I}, \{y\}^{\downarrow I \uparrow I} \rangle \\ &\quad (= \langle \{x\}^{\uparrow I \downarrow I}, \{x\}^{\uparrow I} \rangle \preceq g(\langle \{y\}^{\downarrow I}, \{y\}^{\downarrow I \uparrow I} \rangle)) \end{aligned} \tag{27}$$

is a block $\mathbf{L}$-relation of $\langle X, Y, I \rangle$ such that its mappings $^{\theta}$ and ${}_{\theta}$ are equal to the mappings f and g, respectively.

Proof. (a) Since J is a block $\mathbf{L}$-relation then $A^{\uparrow_J}$ is an intent of $\langle X,Y,I\rangle$ and $f(\langle A,B\rangle) \in \mathscr{B}(X^{\uparrow},Y^{\downarrow},I)$. Similarly for g.

The pair $\langle {}^{\uparrow_J}, {}^{\downarrow_J}\rangle$ is an antitone $\mathbf{L}$-Galois connection between L^X and L^Y. Thus, for each $A \in L^X$ and $B \in L^Y$ we have $S(A,B^{\downarrow_J}) = S(B,A^{\uparrow_J})$. Now,

$$f(\langle A_1,B_1\rangle) \preceq \langle A_2,B_2\rangle = S(B_2,A_1^{\uparrow_J}) = S(A_1,B_2^{\downarrow_J}) = \langle A_1,B_1\rangle \preceq g(\langle A_2,B_2\rangle)$$

proving $\langle f,g\rangle$ is an isotone $\mathbf{L}$-Galois connection. Since $I \subseteq J$ then $A^{\uparrow_I} \subseteq A^{\uparrow_J}$ and f is deflationary.

(b) First we need to show that each object intent of $\langle X,Y,J\rangle$ is an intent of $\langle X,Y,I\rangle$ and each attribute extent of $\langle X,Y,J\rangle$ is an extent of $\langle X,Y,I\rangle$. We shall prove the part for extents. For each $x \in X$ we have

$$\begin{aligned}\{x\}^{\uparrow_J}(y) = J(x,y) = S(f_X(\{x\}^{\uparrow_I\downarrow_I}),\{y\}^{\downarrow_I}) &= \bigwedge_{x'\in X} f_X(\{x\}^{\uparrow_I\downarrow_I})(x') \to \{y\}^{\downarrow_I}(x') \\ &= \bigwedge_{x'\in X} f_X(\{x\}^{\uparrow_I\downarrow_I})(x') \to I(x',y) = f_X(\{x\}^{\uparrow_I\downarrow_I})^{\uparrow_I}(y)\end{aligned}$$

and $\{x\}^{\uparrow_J}$ is an intent of $\langle X,Y,I\rangle$. That proves that $\mathrm{Ext}(X^{\uparrow},Y^{\downarrow},J) \subseteq \mathrm{Ext}(X^{\uparrow},Y^{\downarrow},I)$; similarly can be shown that $\mathrm{Int}(X^{\uparrow},Y^{\downarrow},J) \subseteq \mathrm{Int}(X^{\uparrow},Y^{\downarrow},J)$. From properties of inflationary $\mathbf{L}$-Galois connections and antitone Galois connections, we have $f_X(\{x\}^{\uparrow_I\downarrow_I})^{\uparrow_I}(y) \geq \{x\}^{\uparrow_I\downarrow_I\uparrow_I}(y) = \{x\}^{\uparrow_I}(y)$; hence $J \supseteq I$. The equality of ${}^{\theta}, {}_{\theta}$ and f,g is immediate.

Theorem 11 and Theorem 8 generalize [7, Theorem 15]. The following theorem represents the main result of this paper.

Theorem 12. *(a) Complete* $\mathbf{L}$*-tolerances are in one-to-one correspondence with block* $\mathbf{L}$*-relations.*

(b) If complete $\mathbf{L}$*-tolerance* $\sim$ *and block* $\mathbf{L}$*-relation* J *are in that correspondence, then* $\mathscr{B}(X^{\uparrow},Y^{\downarrow},J)$ *is isomorphic to the lattice of blocks of* $\mathbf{L}$*-tolerance* $\sim$*.*

Proof. The one-to-one correspondence follows from Theorem 11 and the fact that different block relation produce different mappings ${}^{\theta}, {}_{\theta}$. The isomorphism of $\mathscr{B}(X^{\uparrow},Y^{\downarrow},J)$ and the lattice of $\sim$ then follows from definition of ${}^{\theta}, {}_{\theta}$ and their equality to mappings f,g of $\sim$.

5 Illustrative Example

In this section, we use results developed in the previous sections to factorize the $\mathbf{L}$-concept lattice $\mathscr{B}(X^{\uparrow},Y^{\downarrow},I)$ from Example 1.

Our aim is to reduce the size of the $\mathbf{L}$-concept lattice $\mathscr{B}(X^{\uparrow},Y^{\downarrow},I)$ by factorization by complete $\mathbf{L}$-tolerances. First we take the approach from [2]. This approach is based on a choice of a threshold $a \in L$ and using the a-cut ${}^{a}\approx$ of the $\mathbf{L}$-equality $\approx$ on $\mathscr{B}(X^{\uparrow},Y^{\downarrow},I)$ for factorization. The a-cut ${}^{a}\approx$ is a complete tolerance on the complete lattice $\langle\mathscr{B}(X^{\uparrow},Y^{\downarrow},I),{}^{1}\preceq\rangle$ and, as noted in [5], the factor lattice is isomorphic to the crisp part of the concept lattice $\mathscr{B}(X^{\uparrow},Y^{\downarrow},a \to I)$. As it can be easily seen, these results are a special case of the results of this paper. Namely, one can take the complete

L-tolerance $a \to \approx$ on $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$ (the fact that it is indeed a complete **L**-tolerance easily follows from Lemma 2) and construct the associated block **L**-relation $J \supseteq I$, which is equal to $a \to I$. The factorized **L**-concept lattice $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)/^{a}\approx$ is isomorphic to $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, a \to I)$.

As an example, we present in Fig. 3 results we obtained for our formal context of movies, critics and ratings with values of the threshold a equal to 0.4 and 0.3.

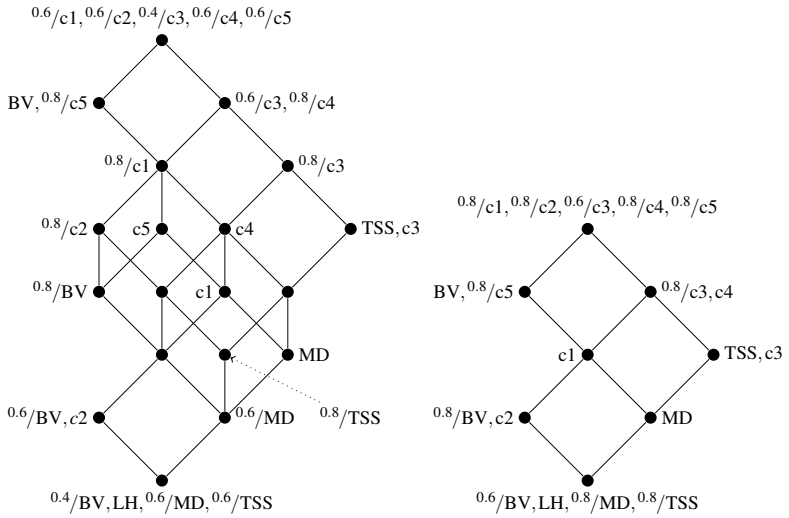


Fig. 3. **L**-concept lattice of movies, critics and ratings (from Fig. 1), factorized with respect to the threshold $a = 0.8$ (left) and $a = 0.6$ (right).

We also tried a more sophisticated approach, based on [13]. The author's method is based on using a complete tolerance on a complete lattice induced by an interior and closure operator. As an example, the author uses a complete tolerance on a (crisp) concept lattice, obtained by selecting subsets $X' \subseteq X$ and $Y' \subseteq Y$ of important objects and important attributes, respectively.

Although a proper fuzzification of the results from [13] remains to be developed, it is possible to try some experiments. We provide a short outline of our method here and leave the details to a forthcoming paper.

For a formal **L**-context $\langle X, Y, I \rangle$ we select two **L**-sets $X' \in L^X$ and $Y' \in L^Y$ and interpret them as **L**-sets of "important objects" and "important attributes", respectively. Thus, for an object $x \in X$, the value $X'(x)$ is the degree to which x is important, and similarly for attributes. Further we define two **L**-relations $I_{X'}, I_{Y'} \in L^{X \times Y}$ by

$$I_{X'}(x,y) = X'(x) \to I(x,y), \quad I_{Y'}(x,y) = Y'(y) \to I(x,y).$$

As it can be easily seen from (15), $\mathrm{Int}(X^{\uparrow}, Y^{\downarrow}, I_{X'}) \subseteq \mathrm{Int}(X^{\uparrow}, Y^{\downarrow}, I)$ and $\mathrm{Ext}(X^{\uparrow}, Y^{\downarrow}, I_{Y'}) \subseteq \mathrm{Ext}(X^{\uparrow}, Y^{\downarrow}, I)$. Thus, the **L**-relations $I_{X'}$ and $I_{Y'}$ select some intents and extents of formal

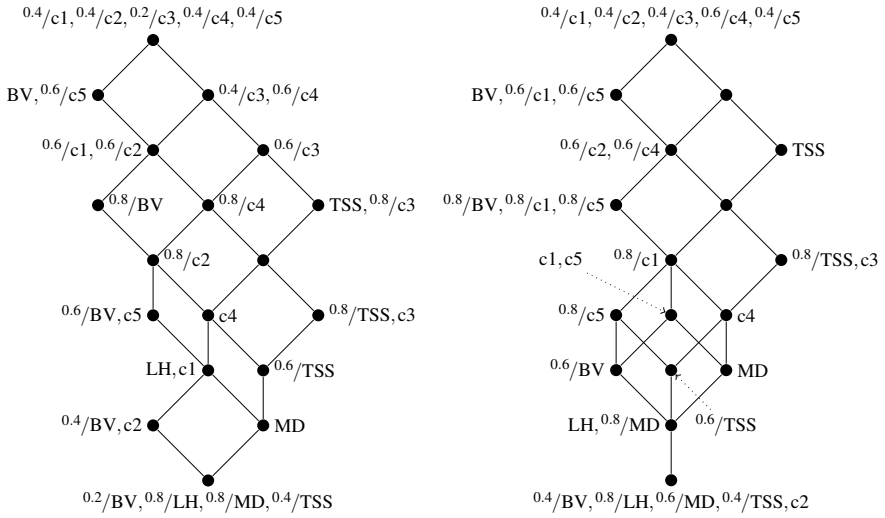


Fig. 4. **L**-concept lattice of movies, critics and ratings (form Fig. 1), factorized with respect to an **L**-set of important objects $X' = \{^{0.8}/\mathrm{BV}, \mathrm{LH}, \mathrm{MD}, \mathrm{TSS}\}$ and **L**-set of important attributes $Y' = \{\mathrm{c1}, ^{0.8}/\mathrm{c2}, \mathrm{c3}, \mathrm{c4}, \mathrm{c5}\}$ (left) and with respect to an **L**-set of important objects $X' = \{\mathrm{BV}, \mathrm{LH}, \mathrm{MD}, \mathrm{TSS}\}$ and **L**-set of important attributes $Y' = \{\mathrm{c1}, ^{0.6}/\mathrm{c2}, \mathrm{c3}, \mathrm{c4}, \mathrm{c5}\}$ (right).

concepts from $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$. These intents and extents are interpreted as "important". Now, let $J_{X',Y'} \in L^{X \times Y}$ be the minimal (with respect to **L**-set inclusion) block **L**-relation of I such that both intent and extent of each formal **L**-concept from $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, J_{X',Y'})$ are important ($J_{X',Y'}$ always exists because of Theorem 10(a); details are omitted). Using the results of this paper (Theorem 12) we obtain that $J_{X',Y'}$ induces a complete **L**-tolerance on the **L**-concept lattice $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)$ and the corresponding factor completely lattice **L**-ordered set is isomorphic to the **L**-concept lattice $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, J_{X',Y'})$.

Note that this approach contains the approach from [2] as a special case. The factor completely lattice **L**-ordered set $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, I)/^{a}\approx$ is isomorphic to $\mathscr{B}(X^{\uparrow}, Y^{\downarrow}, J_{X',Y'})$ for $X' = \{^{a}/x \mid x \in X\}$ and $Y' = \{^{a}/y \mid y \in Y\}$.

We apply the above considerations to our example. Suppose we consider the film BV less important than the other films (perhaps because we have not seen BV) and the critic c2 less important than the other critics (because we do not like his opinion on MD). More precisely, set $X' = \{^{a}/\mathrm{BV}, \mathrm{LH}, \mathrm{MD}, \mathrm{TSS}\}$ and $Y' = \{\mathrm{c1}, ^{b}/\mathrm{c2}, \mathrm{c3}, \mathrm{c4}, \mathrm{c5}\}$, where $a, b \in L$. In Fig. 4 we can see the resulting concept lattices in two cases: first $a = b = 0.8$ and second $a = 1$ and $b = 0.6$.

6 Conclusions

We presented a generalization of the theory of complete tolerances on complete lattices and the relationship of block relations on formal contexts and complete tolerances on the corresponding concept lattices [6, 17, 7] to fuzzy setting. Our results can be used

for reducing the size of fuzzy concept lattices by means of factorization and offer a greater degree of variability than the known approach from [2]. In the future we will focus namely on the investigation of block relations in the isotone case and developing the theory of approximations in fuzzy concept lattices which would give a proper theoretical background to our experiments from Sec. 5.

References

1. Belohlavek, R.: Concept lattices and order in fuzzy logic. Ann. Pure Appl. Log. 128(1-3), 277–298 (2004)
2. Belohlavek, R.: Similarity relations in concept lattices. J. Log. Comput. 10(6), 823–845 (2000)
3. Belohlavek, R.: Fuzzy Relational Systems: Foundations and Principles. Kluwer Academic Publishers, Norwell, USA (2002)
4. Belohlavek, R., Konecny, J.: Operators and spaces associated to matrices with grades and their decompositions, to appear, submitted to Fundamenta Informaticae
5. Belohlavek, R., Outrata, J., Vychodil, V.: Direct factorization by similarity of fuzzy concept lattices by factorization of input data. In: Yahia, S.B., Nguifo, E.M., Belohlavek, R. (eds.) CLA. Lecture Notes in Computer Science, vol. 4923, pp. 68–79. Springer (2006)
6. Czédli, G.: Factor lattices by tolerances. Acta Sci. Math. 44, 35–42 (1982)
7. Ganter, B., Wille, R.: Formal Concept Analysis – Mathematical Foundations. Springer (1999)
8. Georgescu, G., Popescu, A.: Non-dual fuzzy connections. Arch. Math. Log. 43(8), 1009–1039 (2004)
9. Hájek, P.: Metamathematics of Fuzzy Logic (Trends in Logic). Springer (November 2001)
10. Kelly, G.M.: Basic Concepts of Enriched Category Theory. Cambridge University Press (1982)
11. Kohout, L. J.; Bandler, W.: Relational-product architectures for information processing. Information Sciences 37(1-3), 25–37 (1985)
12. Krupka, M.: An alternative version of the main theorem of fuzzy concept lattices. In: Trappl, R. (ed.) Cybernetics and Systems 2010. pp. 9–14. Austrian Society for Cybernetic Studies, Vienna (2010), extended version submitted
13. Meschke, C.: Approximations in concept lattices. In: Kwuida, L., Sertkaya, B. (eds.) Formal Concept Analysis, Lecture Notes in Computer Science, vol. 5986, pp. 104–123. Springer Berlin / Heidelberg (2010)
14. Pollandt, S.: Fuzzy Begriffe: Formale Begriffsanalyse von unscharfen Daten. Springer–Verlag, Berlin–Heidelberg (1997)
15. Stubbe, I.: Categorical structures enriched in a quantaloid: tensored and cotensored categories. Theory Appl. Categ. 16, No. 14, 283–306 (electronic) (2006), http://www.ams.org/mathscinet-getitem?mr=2223039
16. Ward, M., Dilworth, R.P.: Residuated lattices. Transactions of the American Mathematical Society 45, 335–354 (1939)
17. Wille, R.: Complete tolerance relations of concept lattices. In: Eigenthaler, G., et al. (eds.) Contributions to General Algebra, vol. 3, pp. 397–415. Hölder-Pichler-Tempsky, Wien (1985)
18. Zhao, H., Zhang, D.: Many valued lattices and their representations. Fuzzy Sets and Systems 159(1), 81–94 (Jan 2008)

A closure algorithm using a recursive decomposition of the set of Moore co-families

Pierre Colomb[1], Alexis Irlande[2], Olivier Raynaud[1], Yoan Renaud[3]

[1] Clermont Université, Université Blaise Pascal, Campus des Cézeaux
63173 Clermont-Ferrand, France
pierre@colomb.me, raynaud@isima.fr
[2] Universidad Nacional de Colombia
Bogota, Colombia
irlande@lirmm.fr
[3] INSA de Lyon, Bâtiment Blaise Pascal
Campus de la Doua, 69621 Villeurbanne, France
yoan.renaud@insa-lyon.fr

Abstract. Given a set $U_n = \{1, ..., n\}$, a collection $\mathcal{M}$ of subsets of U_n that is closed under intersection and contains U_n is known as a Moore family. The set of Moore families for a fixed n is in bijection with the set of Moore co-families (union-closed families containing the empty set) denoted itself $\mathbb{M}_n$. This paper follows the work initiated in [8] and [9] about the recursive decomposition of the lattice of Moore co-families. We first show that each Moore co-family can be represented by a decomposition tree and we use this principle to design an original algorithm to close under union any given family. Then we discuss about the time complexity and give some experimental results.

1 Introduction

The concept of collection of sets on a ground set U_n closed under intersection appears with different names depending of the scientific fields. The name 'Moore families' was first used by Birkhoff in [4] referring to E.H. Moore's researches. But, very frequently, such a collection is called 'closure spaces' (or 'closure systems') or 'convexity spaces'. This concept is applied in numerous fields in pure or applied mathematics and computer science. For mathematical researches in algebra and topology we can cite [7, 19, 20]. For researches in order and lattice theory we have to cite [4, 10] for their works on closure operators. Formally a closure operator is an extensive, isotone and idempotent function on 2^{U_n}, and a closure system is then the sets of its fixed points. In particular it is shown that any closure system is a complete lattice. From 1937 Birkhoff in [3] gave a compact representation of 'quasi ordinal spaces' (in other words of collections closed by intersection and union also called distributive lattices). More recently the collection appears again as the main concept in computer science with researches in relational databases ([11]), in data analysis and formal concept analysis ([13, 1, 15]). More precisely, Ganter and Wille define a mathematical framework for classification and Barbut and Monjardet used the Galois lattice for equivalent tasks.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

In the same time, in 1985 ([12]) such collections are called 'knowledge spaces' by Doignon et Falmagne. An important fact is that the collection of Moore families on U_n is itself a Moore family or a closure system. Indeed, the system composed of Moore families contains one maximum element (2^{U_n}, all subsets of U_n) and the intersection of two Moore families is a Moore family itself. To get an overall view of the properties of this closure system, see the survey of Caspard and Monjardet [6].

Previously in the introduction, we have defined a Moore family on U_n as a collection of sets containing U_n and closed by intersection. But for reasons of legibility of the results and simplification of expressions, this article deals with the set of families closed by union and containing the empty set called Moore co-families and denoted by $\mathbb{M}_n$. Basically, the set of Moore families is in bijection with this set. For a given Moore family, one only has to complement every set to obtain a Moore co-family. For example, the Moore family $\{\{1\},\{1,2\},\{1,3\},\{1,2,3\}\}$ on U_3 corresponds to the Moore co-family $\{\emptyset,\{2\},\{3\},\{2,3\}\}$ and vice versa.

In [8], Colomb & al counted the exact number of Moore co-families for $n = 7$ and stated a recursive decomposition theorem of the lattice of Moore co-families in [9]. In the same article authors state that the set $\mathbb{M}_n$ is endowed with the quotient partition associated with the operator h (each class of the partition contains all the families which have the same image by h) and prove that each class has a distributive lattice structure. This operator h is the main concept underlying the recursive description of $\mathbb{M}_n$. More recently in [2] authors give a complete characterization of the set $h(\mathcal{M}) \setminus \mathcal{M}$.

In the present article we pursue investigations involving Moore co-families by using the recursive decomposition theorem. In the third section we describe succinctly the theorem and we show that each Moore co-family can be represented by a decomposition tree. In the fourth section we describe an original algorithm to generate a Moore co-family from the set of its join-irreducible elements. Then we discuss about the time complexity of the process. Some experimental results are given in section five.

In the rest of the paper, we denote elements by numbers $(1, 2, 3, \dots)$. Sets are denoted by capital letters $(A, B, C, \dots)$. Families of sets are denoted by calligraphic letters $(\mathcal{A}, \mathcal{B}, \mathcal{C}, \dots)$. Finally, we denote the sets of families of sets by black board letters $(\mathbb{A}, \mathbb{B}, \mathbb{C}, \dots)$.

2 Definitions and notations

For any integer $n \geq 1$, let U_n denote the set $\{1, \dots, n\}$. Let $\mathcal{M}$ be a family, we note $(\mathcal{M}, \subseteq)$ the corresponding partial order. Two sets M, M' in $\mathcal{M}$ such that neither $M \subseteq M'$ and $M' \subseteq M$ are said *incomparable* in $(\mathcal{M}, \subseteq)$. A family where every pair of sets is incomparable is called an antichain. We say that M is covered by M' in $(\mathcal{M}, \subseteq)$, denoted $M \prec M'$, if $M \subset M'$ and there is not $M" \in \mathcal{M}$ such that $M \subset M"$ and $M" \subset M'$.

Given a family $\mathcal{M}$, a subfamily $\mathcal{I}$ of $\mathcal{M}$ is an *ideal* of $\mathcal{M}$ if it satisfies the following implication for any pair M and M' in $\mathcal{M}$,

$$M \subseteq M' \text{ and } M' \in \mathcal{I} \Rightarrow M \in \mathcal{I}.$$

In other words, an ideal of $\mathcal{M}$ is some antichain of $\mathcal{M}$ and everything below it. We shall use $\mathbb{I}_{\mathcal{M}}$ to denote the sets of ideals on $\mathcal{M}$. Given a set X in a family $\mathcal{M}$, there exists a unique ideal $\mathcal{I}$ of $\mathcal{M}$ with X as a maximum set. Let $\mathcal{I}_{\mathcal{M}}(X)$ denote this ideal. A set $J \in \mathcal{M}$ is called a join-irreducible if J covers only one set. The set of all join-irreducible sets of $\mathcal{M}$ is denoted $\mathcal{J}_{\mathcal{M}}$. Let $\mathcal{M}$ be a Moore co-family and x an element, we denote $\mathcal{M}_x$ (resp. $\mathcal{M}_{\bar{x}}$) the restriction of $\mathcal{M}$ to its sets containing x (resp. to its sets not containing x). The empty set is added to $\mathcal{M}_x$. By extension, we denote by $\mathcal{J}_x$ (resp. $\mathcal{J}_{\bar{x}}$) the set of join-irreducible elements of $\mathcal{M}$ which contain the element x (resp. which do not contain the element x).

3 Recursive decomposition of the Moore co-families lattice

In previous work, Colomb & al. gave a recursive definition of the Moore co-families on U_n, for any $n \geq 1$ (the only Moore co-family on $U_0 = \emptyset$, is $\{\emptyset\}$).

Definition 1. *For any integer n such that $n \geq 1$, we define*

$$\begin{array}{lll}
g_{1,n} : \mathbb{M}_{n-1} & \longrightarrow & \mathbb{M}_n \\
\mathcal{M} & \longmapsto & \{X \in 2^{U_n} \mid \exists M \in \mathcal{M} \setminus \{\emptyset\} \textit{such that } X = M \cup \{n\}\} \cup \{\emptyset\}, \\
g_{2,n} : \mathbb{M}_{n-1} & \longrightarrow & \mathbb{M}_n \\
\mathcal{M} & \longmapsto & \{X \in 2^{U_n} \mid \exists M \in \mathcal{M} \textit{ such that } X = M \cup \{n\}\} \cup \{\emptyset\}, \\
h_n : \ \mathbb{M}_{n-1} & \longrightarrow & \mathbb{M}_{n-1} \\
\mathcal{M} & \longmapsto & \{X \in 2^{U_{n-1}} \mid \forall M \in g_{1,n}(\mathcal{M}) \setminus \{\emptyset\},\ X \cup M \in g_{1,n}(\mathcal{M})\}.
\end{array}$$

We may use h, g_1 and g_2 instead of $h_n, g_{1,n}$ and $g_{2,n}$ when no confusion is possible. Function g_2 consists in adding element n to every set of a family $\mathcal{M}$ in $\mathbb{M}_{n-1}$ (thus including the singleton $\{n\}$) plus the empty set. Function g_1 behaves similarly but removes the singleton $\{n\}$. With these functions defined, we can state Theorem 1, giving a description of $\mathbb{M}_n$ with respect to $\mathbb{M}_{n-1}$.

Theorem 1 (Colomb & al. [9]).
For any integer n such that $n \geq 1$,

$$\begin{array}{l}
\mathbb{M}_n = \bigcup_{\mathcal{M} \in \mathbb{M}_{n-1}} \{g_1(\mathcal{M}) \cup \mathcal{M}' \mid \mathcal{M}' \in \mathcal{I}_{\mathbb{M}_{n-1}}(h(\mathcal{M}))\} \cup \\
\qquad \bigcup_{\mathcal{M} \in \mathbb{M}_{n-1}} \{g_2(\mathcal{M}) \cup \mathcal{M}'' \mid \mathcal{M}'' \in \mathcal{I}_{\mathbb{M}_{n-1}}(\mathcal{M})\}
\end{array}$$

In other words there exists a natural bi-partition of $\mathbb{M}_n$. Families not containing the singleton $\{n\}$ under the form $g_1(\mathcal{M}) \cup \mathcal{M}'$ with $\mathcal{M}'$ a sub-Moore co-family of $h(\mathcal{M})$ and families containing $\{n\}$ under the form $g_2(\mathcal{M}) \cup \mathcal{M}''$ with $\mathcal{M}''$ a sub-Moore co-family of $\mathcal{M}$.

3.1 Decomposition tree of a Moore co-family

Another interpretation is that for any element x in U_n and any Moore co-family $\mathcal{M}$ in $\mathbb{M}_n$, $\mathcal{M}$ can be written $\mathcal{M}_{\bar{x}} \cup g_{i,x}(\mathcal{M}')$ with $\mathcal{M}'$ such that $g_{i,x}(\mathcal{M}')$ corresponds to $\mathcal{M}_x$. By applying recursively this decomposition principle to $\mathcal{M}$ we obtain a decomposition binary tree of $\mathcal{M}$ (each leaf is an empty set which cannot be further decomposed). Basically each leaf corresponds to a set of the initial family $\mathcal{M}$: to obtain the set corresponding to a leaf, one has only to apply iteratively the different functions $g_{1,x}$ or $g_{2,x}$ that can be found in the path from the chosen leaf to the root of the tree. An example of decomposition of Moore co-family is given in Figure 1.

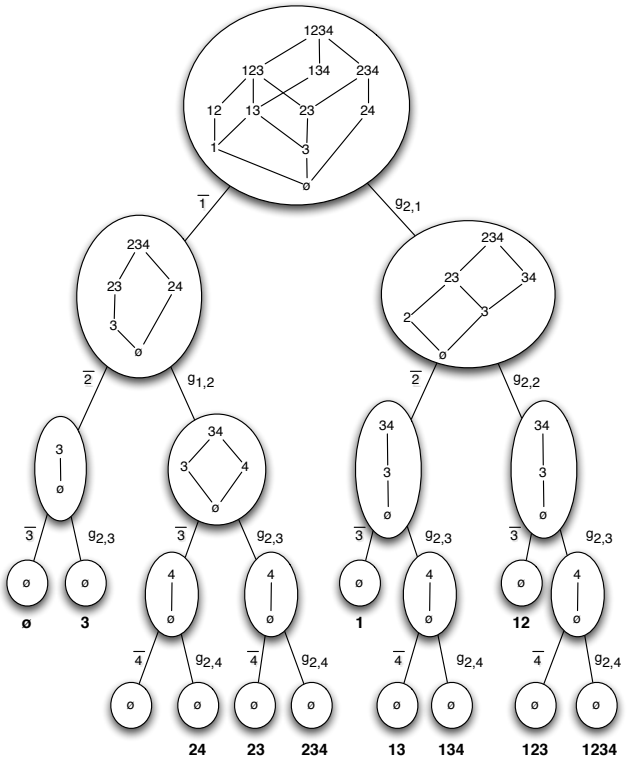


Fig. 1. Decomposition of a Moore co-family

4 A new algorithm to compute a union-closed family

In this section, we present an algorithm to generate a Moore co-family $\mathcal{M}$ from its set of join-irreducible elements denoted $\mathcal{J}_{\mathcal{M}}$. This algorithm is based on the theorem 1.

4.1 Closure under union algorithm

Problem of generation of closed-sets from a representative family has been well-studied these last years. Most of existing algorithms are based on a decomposition strategy [14, 5, 17, 16]. As previously stated, generation of closed-sets and co-closed sets can be treated in the same way (i.e. one only have to complement every sets of input and output families to obtain both closed and co-closed families). The question we ask here is, given a family of sets $\mathcal{J}$, how to compute its associated Moore co-family $\mathcal{M}$, i.e. the smallest Moore co-family that contains all sets of $\mathcal{J}$. We choose to denote $\mathcal{J}$ the given set because the smallest representative family for a Moore co-family is the family of its join-irreducible sets.

We propose an algorithm based on the decomposition of a Moore co-family that only uses families of join-irreducible sets. In other words, for each step of the recursive decomposition, given a local set $\mathcal{J}_{\mathcal{M}}$, we compute families of join-irreducible sets $\mathcal{J}_{\mathcal{M}_{\bar{x}}}$ and $\mathcal{J}_{\mathcal{M}'}$ corresponding to $\mathcal{M}_{\bar{x}}$ and $\mathcal{M}'$ such that $\mathcal{M} = \mathcal{M}_{\bar{x}} \cup g_i(\mathcal{M}')$. Leaves of the decomposition tree obtained after the whole recursive decomposition correspond to the closure of the initial set $\mathcal{M}$. In the following, we describe how to compute $\mathcal{J}_{\mathcal{M}_{\bar{x}}}$ and $\mathcal{J}_{\mathcal{M}'}$.

Proposition 1. *Let $\mathcal{M}$, $\mathcal{M}'$ in $\mathbb{M}_n$, x in U_n and i in $[1,2]$ with $\mathcal{M} = \mathcal{M}_{\bar{x}} \cup g_{i,x}(\mathcal{M}')$. Then*

- $\mathcal{J}_{\mathcal{M}_{\bar{x}}} = \mathcal{J}_{\bar{x}}$.
- $\mathcal{J}_{\mathcal{M}'} \subseteq \{J \backslash \{x\} \mid J \in \mathcal{J}_x\} \cup \{J_1 \cup J_2 \mid J_1 \in \{J \backslash \{x\} \mid J \in \mathcal{J}_x\},\ J_2 \in \mathcal{J}_{\bar{x}}\}$

Proof.

- $\mathcal{J}_{\mathcal{M}_{\bar{x}}} = \mathcal{J}_{\bar{x}}$
  Any join-irreducible element of $\mathcal{M}$ not containing x is also join-irreducible of $\mathcal{M}_{\bar{x}}$. Indeed, every predecessors of each set in $\mathcal{M}$ not containing x, doesn't contain x itself. Similarly, there is no new join-irreducible element in $\mathcal{J}_{\mathcal{M}_{\bar{x}}}$.
- $\mathcal{J}_{\mathcal{M}'} \subseteq \{J \backslash \{x\} \mid J \in \mathcal{J}_x\} \cup \{J_1 \cup J_2 \mid J_1 \in \{J \backslash \{x\} \mid J \in \mathcal{J}_x\},\ J_2 \in \mathcal{J}_{\bar{x}}\}$
  Let us show that $g_i(\mathcal{J}_{\mathcal{M}'}) \subseteq \mathcal{J}_x \cup \{J_1 \cup J_2 \mid J_1 \in \mathcal{J}_x,\ J_2 \in \mathcal{J}_{\bar{x}}\}$.
  By contradiction, let J be a set in $g_{i,x}(\mathcal{J}_{\mathcal{M}'})$ such that $J \notin \mathcal{J}_x \cup \{J_1 \cup J_2 \mid J_1 \in \mathcal{J}_x,\ J_2 \in \mathcal{J}_{\bar{x}}\}$. So, we have $x \in J$, $J \notin \mathcal{J}_x$ and $J \notin \mathcal{J}_{\bar{x}}$. Hence, $J \notin \mathcal{J}_{\mathcal{M}}$.
  Let $S = \{s_1, s_2, ..., s_n\}$ (resp. $S' = \{s'_1, s'_2, ..., s'_m\}$) be the family of maximal join-irreducible sets of $\mathcal{M}_{\bar{x}}$ (resp. of $\mathcal{M}_x$) such that $\forall i \in [1,n]$, $s_i \subset J$ (resp. $\forall j \in [1,m]$, $s'_j \subseteq J$).

Since $J \notin \mathcal{J}_x \cup \{J_1 \cup J_2 \mid J_1 \in \{J \backslash \{x\} \mid J \in \mathcal{J}_x\},\ J_2 \in \mathcal{J}_{\bar{x}}\}$ we have $\nexists i \in [1,n]$ and $\nexists j \in [1,m]$ such that $s_i \cup s'_j = J$ or $s'_j = J$. So, $\exists i,j \in [1,n]$ and $\exists k,l \in [1,m]$ such that $s_i \cup s'_k$ is incomparable with $s_j \cup s'_l$ and such that $\forall u \in [1,n]$ and $\forall v \in [1,m]$, either $s_u \cup s'_v \subseteq s_i \cup s'_k$, or $s_u \cup s'_v \subseteq s_j \cup s'_l$.
But, with $J = s_i \cup s'_k \cup s_j \cup s'_l$, we can say that J covers $s_i \cup s'_k$ and $s_j \cup s'_l$. Hence, J is not a join-irreducible set of $\mathcal{M}_x$, $J \setminus \{x\}$ is not a join-irreducible set of $\mathcal{M}'$ and we conclude that J does not belongs to $g_{i,x}(\mathcal{J}_{\mathcal{M}'})$. Contradiction.

□

Straightforward from proposition 1 we can design an algorithm to close any given input family (see algorithm 1).

For the first step of algorithm Co-closure, P is initialized to U_n that represents all possible choices of an element for the next decomposition. S is initialized to the empty set and allows to store element that would form a co-closed set. $Verif$ is initialized to $true$. An example of an execution is given in figure 2.

4.2 Discussion about the time complexity of algorithm Co-closure.

Basically, the decomposition tree obtained at the end of the whole process of Algorithm 1 is a binary tree. Internal nodes have zero, one or two children. Only the last case induces the algorithm to compute a direct product which create a new path in the tree corresponding to a new closed set. For this reason we can say that Algorithm 1 computes as many direct products as the number of different closed sets of the final result. By this way, the most important part of the whole time complexity, which corresponds to the computation of direct products, can be dispatched on each internal node with two children. Locally, the algorithm computes one direct product between two families whose the size is variable. Let S be an hypothetic maximal size of a family involved in a product. Then, treatment of an internal node is done with runtime of $\mathcal{O}(n.S^2)$. But one have to know that the size of the family resulting of the product is never superior to the number of leaves of the sub tree rooted to this internal node. That means that the local complexity is polynomially linked with the result.

We give in the last section the experimental results. They are far to be favorable to our approach and one explication could be the following: we said that the size of the direct product computed for each internal node is linked to the number of families which have to be generated further, but the process to compute this product is not linear with the size of the local result of this product. This point give us a huge space to improve our global process. That means that Algorithm 1 makes the link between computing a closure and computing a direct product under constraints to be defined.

Algorithm 1: Co-closure($\mathcal{J}$, P, S, $Verif$)

Input: $\mathcal{J}$ a family of sets; P, S sets of elements; $Verif$ a Boolean.
begin
  if $\mathcal{J} \neq \{\emptyset\}$ *and* $\mathcal{J} \neq \{\}$ **then**
    Choose an element $x \in P$; $\mathcal{J}_{\mathcal{M}_{\bar{x}}} \leftarrow \mathcal{J}_{\bar{x}}$;
    Co-closure($\mathcal{J}_{\mathcal{M}_{\bar{x}}}$, $\bigcup \mathcal{J}_{\mathcal{M}_{\bar{x}}}$, S, $Verif$);
    $\mathcal{T} \leftarrow \{J \backslash \{x\} \mid J \in \mathcal{J}_x\}$;
    $\mathcal{J}_{\mathcal{M}'} \leftarrow \mathcal{T} \cup \{T \cup J \mid T \in \mathcal{T},\ J \in \mathcal{J}_{\mathcal{M}_{\bar{x}}}\}$;
    if $\{x\} \notin \mathcal{J}$ **then**
      $Verif \leftarrow false$;
    else
      $Verif \leftarrow true$;
    Co-closure($\mathcal{J}_{\mathcal{M}'}$, $\bigcup \mathcal{J}_{\mathcal{M}'}$, $S \cup \{x\}$, $Verif$);
  else
    if $Verif = true$ *and* $\mathcal{J} \neq \{\}$ **then**
      $return(S)$;
    $S \leftarrow \{\}$;
end

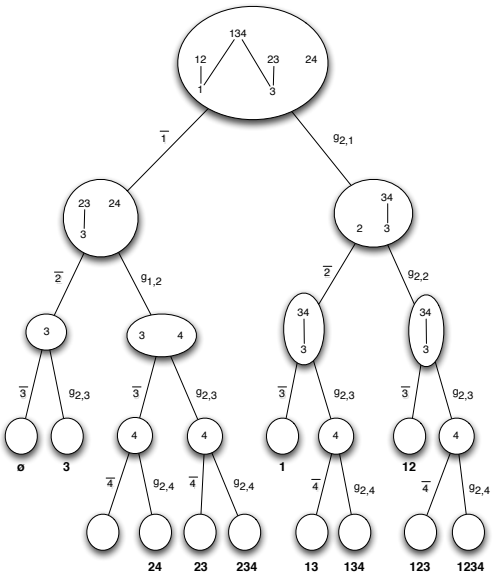


Fig. 2. Decomposition tree corresponding to the execution of the algorithm on the input family $\{\{1\}, \{1,2\}, \{3\}, \{2,3\}, \{1,3,4\}, \{2,4\}\}$

5 Experimental results

5.1 Experimental design

In order to assess performances, the approach described previously has been implemented in C. The executable file was generated with compiler GCC 4.6. The experiments were performed on a Intel Core2 Q9300 processor with 2.2 GHz of speed and 8 Go of memory. Several well-known benchmark real-world data-sets were chosen from [21]. The following table summarizes the data-sets caracteristics.

Data-sets	# Attributes	# Sets	# Attributes per set	# Closed set
Mushroom	119	8124	23	238709
Chess	75	3196	37	930851336
Connect	129	67557	43	1415737994

To compare performances of Co-closure and previous approaches with respect to the size of family, we implemented a version of Norris's algorithm (see [18]) as well as a version of Ganter's algorithm called Next-closure (see [14]). Note that we have used the natural one to one mapping between Moore and co-Moore families to compute closed sets using Co-Closure algorithm.

5.2 Results

We have compared execution time of these different algorithms on parts of benchmark data-sets. In that way, we have executed algorithms on the first m sets of each benchmark data-sets by varying m. On figures 3, 4, and 5 we give numbers of co-closed sets we obtained by execution of the three algorithms for each size of the considered data-sets partition. In the last three rows we give the execution time for each algorithm.

#sets	# closed	Ganter	Norris	Co-Closure
100	10867	20ms	10ms	160ms
200	73033	80ms	80ms	1s56
500	1051399	1s54	2s98	7s3
1000	38558373	2m04	2m48	31m52
1500	183655113	13m24	1h04	?
2000	292107880	26m36	?	?
2500	386235477	46m25	?	?
3196	930851336	2h33	?	?

Fig. 3. Chess

#sets	# closed	Ganter	Norris	Co-Closure
10	75	20ms	20ms	20ms
20	270	20ms	20ms	30ms
50	1208	30ms	20ms	40ms
100	3459	30ms	30ms	110ms
200	7086	30ms	3ms	380ms
500	17781	60ms	5ms	2s
1000	32513	210ms	100ms	6s
1500	48414	460ms	170ms	15s
2000	58982	750ms	250ms	30s
2500	72008	1s23	360ms	45s
3000	80901	1s75	480ms	79s
3500	94350	2s32	650ms	112s
4000	104104	2s99	810ms	137s
4500	122950	3s	1s21	194s
5000	136401	4s	1s47	237s
5500	150948	5s	1s78	304s
6000	156573	6s	1s92	348s
6500	195677	7s	2s80	517s
7000	214950	8s	3s30	593s
7500	230882	10s	3s73	718s
8000	237874	11s	3s94	815s
8124	238709	12s	3s96	818s

Fig. 4. Mushroom

#sets	# closed	Ganter	Norris	Co-Closure
100	13406	410ms	430ms	550ms
200	63360	470ms	440ms	2s92
500	445676	1s	1s48	2m09
1000	1069569	2s	8s	14m26
2000	4732622	22s	56s	?
5000	22543073	3m51	16m50	?
10000	69916189	23m18	3h58	?
20000	242644240	2h50	?	?
67557	1415737994	4j	?	?

Fig. 5. Connect

6 Conclusion

In [8] authors have computed the exact size of $|\mathbb{M}_7|$. From this first challenge has derived a recursive decomposition theorem which has been formalized in [9]. In the present article we first show that each Moore co-family can be represented by a decomposition tree and we use this principle to design an original algorithm to close under union any given family. The process has been implemented and the results on well-known benchmarks are given in the last section. Experimentations state the correctness of the process but show that time complexity and space complexity are not favorable. However, we think the process is interesting for two main reasons. Firstly, the decomposition tree obtained by the process is based on the recursive definition of the set of Moore co-families and corresponds to a new approach. And secondly the computing of each closure is shown to be linked to the computing of a direct product under constraints to be defined. It will be interesting to use some better practices concerning the direct product to improve the process. In that case the time complexity for each generated set will correspond to the time complexity needed for the algorithmic treatment of an internal node with two children. So, any new performance to compute the local direct product should lead to a more efficient algorithm.

References

1. Barbut, M., Monjardet, B.: Ordre et classification. Hachette (1970)
2. Beaudou, L., Colomb, P., Raynaud, O. In: submitted to ISAAC. (2011)
3. Birkhoff, G.: Rings of sets. Duke Mathematical Journal **3** (1937) 443–454
4. Birkhoff, G.: Lattice Theory. Third edn. American Mathematical Society (1967)
5. Bordat, J.P.: Calcul pratique du treillis de galois d'une correspondance. Math. Sci. Hum. **96** (1986) 31–47
6. Caspard, N., Monjardet, B.: The lattices of closure systems, closure operators, and implicational systems on a finite set : a survey. Discrete Applied Mathematics **127** (2003) 241–269
7. Cohn, P.: Universal Algebra. Harper and Row, New York (1965)
8. Colomb, P., Irlande, A., Raynaud, O.: Counting of Moore families on n=7. In: ICFCA, LNAI 5986, Agadir, Marocco. (2010)
9. Colomb, P., Irlande, A., Raynaud, O., Renaud, Y.: About the recursive décomposition of the lattice of moore co-families. In: ICFCA, Nicosia, Cyprius. (2011)
10. Davey, B.A., Priestley, H.A.: Introduction to lattices and orders. Second edn. Cambridge University Press (2002)
11. Demetrovics, J., Libkin, L., Muchnik, I.: Functional dependencies in relational databases: A lattice point of view. Discrete Applied Mathematics **40(2)** (1992) 155–185
12. Doignon, J.P., Falmagne, J.C.: Knowledge Spaces. Springer, Berlin (1999)
13. Duquenne, V.: Latticial structure in data analysis. Theoretical Computer Science **217** (1999) 407–436
14. Ganter, B.: Two basic algorithms in concept analysis., Preprint 831, Technische Hochschule Darmstadt (1984)
15. Ganter, B., Wille, R.: Formal concept analysis, mathematical foundation, Berlin-Heidelberg-NewYork et al.:Springer (1999)

16. Gely, A.: A generic algorithm for generating closed sets of a binary relation. In: ICFCA'05. (2005)
17. Lindig, C.: Fast concept analysis. In: Working with Conceptual Structures - Contributions to ICCS 2000, Shaker Verlag (August 2000) 152–161
18. 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
19. G.Sierksma: Convexity on union of sets. Compositio Mathematica **volume 42** (1981) 391–400
20. Ven, L.V.D.: Theory of convex structures. North-Hollande, Amsterdam (1993)
21. : Uci machine learning repository.

Iterative Software Design of Computer Games through FCA ⋆

David Llansó, Pedro Pablo Gómez-Martín,
Marco Antonio Gómez-Martín and Pedro A. González-Calero

Dep. Ingeniería del Software e Inteligencia Artificial
Universidad Complutense de Madrid, Spain
email: {llanso,pedrop,marcoa,pedro}@fdi.ucm.es

Abstract. If iteration is the rule in modern software development practices, this is more the case in game development. While the secret recipe for fun in games remains hidden, game development will remain a highly iterative trial-and-error design process.
In this paper we present a semi-automatic process that, through FCA, can assist in the software design of modern videogames. Through FCA we can identify candidate distributions of responsibilities among components, and let the users edit such distributions. We support iteration by facilitating the application of past edits when going through a new iteration of identifying candidate components to accommodate for new version of the game requirements.

1 Introduction

Nowadays, videogames are incredible complex software systems developed after many months (even years) of work. At the same time, they are, for many people, considered a specific way of art, where designers, drawers, 3D modeller and musicians work side by side to create a rich interactive experience. The artistic aspect of videogames turns them into a big challenge in a software engineering point of view, because their requirements are always changing.

One of the most affected modules of this uncertainty is the one responsible of the management of the *game entities* (players, enemies, items, interactive objects, etc.). Traditionally, this piece of code was implemented using a class hierarchy programmed using object-oriented languages such as C++. Modern videogames, however, use a *component-based software architecture* [9, 19, 4, 15]. It is important to mention that in the context of game development this *component-base architecture* term has a different meaning than the one used in Software Engineering. When talking about *components* here we should think of something similar to mixins or traits [7]: they are just small classes that implement specific and ideally independent capabilities in such a way that a game entity is just a collection of this components (see Section 2 for details). Though this may lead to confusion in the rest of the paper we will keep the nomenclature used in game development and we will use the term "*component*" meaning this small classes.

⋆ Supported by the Spanish Ministry of Science and Education (TIN2009-13692-C03-03)

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

Even though the use of components promote flexibility, reusability and extensibility, it does not come without costs (see Section 3 for details). One of the main issues is the lack support in the mainstreams languages (mainly C++) that forces programmers to create a complex source code infrastructure that supports components and the ability of composing entities with them. This also leads to the lack of compiler support in the detection of different problems related to the lost of datatype information that a traditional class hierarchy would have exposed immediately to the programmer.

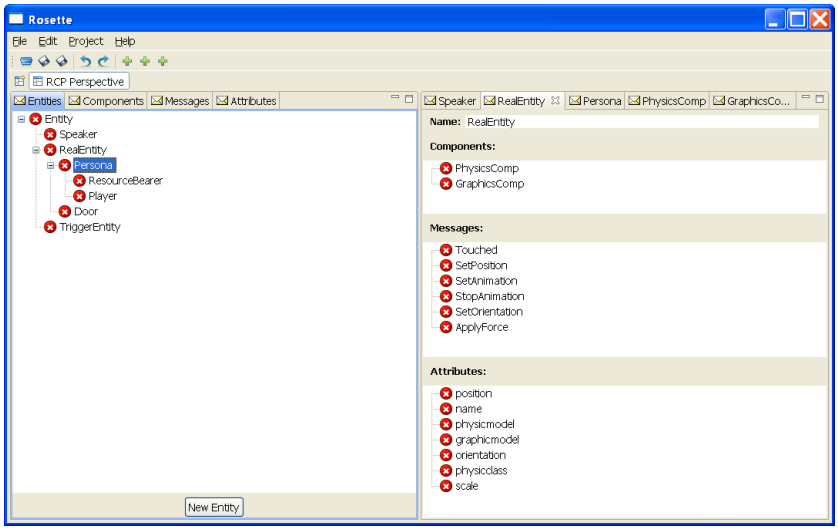


Fig. 1. *Rosette*, the game entity editor

In this paper, we present a new method for easing the design of a component-based architecture. Using a visual tool called *Rosette* [14] (see Figure 1) the user first graphically defines the entity hierarchy (data and actions), a well-known task for game programmers. Using Formal Concept Analysis (FCA) [8], *Rosette* automatically suggests a *component* distribution that will fit the requirements of the provided classical hierarchy (Section 4).

The candidate distribution is shown to the user through a visual interface, where she has the opportunity to modify it (Section 5) before *Rosette* generates a big percentage of the entity manager game code.

Our tool uses OWL (Web Ontology Language[1]) as the underlying representation for the hierarchy and resulting components. It let us track all the applied changes and reason about possible inconsistencies between entities and components that would be hidden to the compiler. This part of the system has been described elsewhere [14].

As said before, videogames specification will surely change in the long run, and the original hierarchy (and component distribution) will need several revisions. As a

[1] http://www.w3.org/TR/owl-features/

result of that *agile methodologies*, such as *scrum* [16] or *extreme programming* [1, 2], are taking a big importance in the videogame developments. Agile methodologies give repeated opportunities to assess the direction of a videogame throughout the entire development lifecycle, which is split in iterations or *sprints*. The principal priority is having functioning pieces of software at the end of every sprint to reevaluate the project goals. To this end, *Rosette* allows an *iterative* hierarchy definition during these sprints, remembering all the changes applied to the previously proposed component distributions, and redoing them into the last one (Section 6). This process is possible using the lattices [3] created using FCA and represented with OWL.

2 Component-Based Architecture

Videogames are very complex systems developed by a lot of people who write a big number of code lines during many months. Due to the changing nature of videogames, where requirements are always changing, there are some parts of the game that should be carefully implemented. Specifically, the module responsible of the management of the *game entities* is the most affected module by those continuous changes so it must be flexible enough to be adapted to unexpected changes in the specification and also offer a good way to reuse code for different entities.

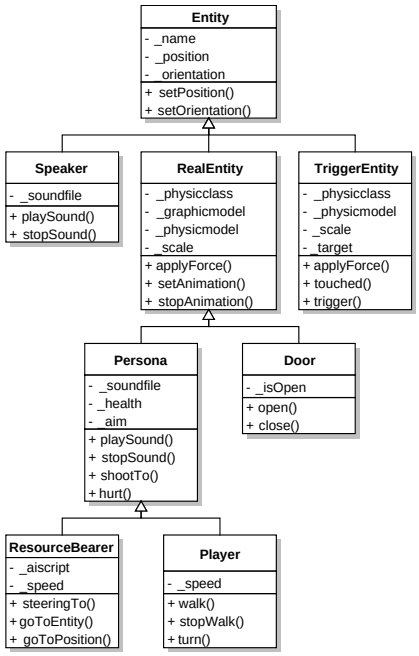


Fig. 2. An entity hierarchy

Traditionally the code layer responsible of the management of the game entities took the form of an inheritance hierarchy of C++ classes. These classes represent the hierarchy of entities and procedures that, in some sense, may be viewed as the actions that these entities were able to perform. Figure 2 shows one of those hierarchies. Each entity class possesses some attributes (data) and methods that code the actions the entity is able to perform. Furthermore, the inheritance propagates features from parents to children; for example, all the attributes and methods of the *Persona* class are also available for *Player*s.

However, the straightforward approach of organizing entities in class hierarchies soon probed too rigid and hard to maintain and evolve. Although well accepted, single inheritance is not expressive enough to handle complex entity distributions. Programmers are usually forced to duplicate code or implement methods too high in the hierarchy [7]. In the last few years, however, the component-based software architecture is the design of choice for managing entities in modern video games [9, 19, 4, 15], embracing this way dynamic object composition instead of static class hierarchies.

Components provide an intermediate level of abstraction between methods and classes by gathering common behaviour and, in this way, they can be seen as mixins or traits [7]. All of them, components, mixins and traits, implement a small set of functionality (usually only one feature) and they invoke other pieces of software that belong to other sibling structures.

The main difference, however, is that in the context of components and games, the entiry hierarchy is flattened and the final result is just an entity class without subclasses that acts as a component container. Every functionality, skill or ability that the entity has, is implemented by a component.

Furthermore, components are more self-contained than mixins or traits since they can have attributes which can specify some states whilst mixins and traits are just a set of methods (we are using the trait concept formalized in [7]). A direct consequence of this is that the entity class can be the same for every entity type, because both their attributes and their functionality are managed by the components. It needs to be just a component container that provides component communication. So, instead of having entities of concrete classes that provide *glue methods* to connect components together like in mixing or traits, the component communication is done like in the *Command* design pattern, where method invocation is transformed into an object that is passed around the components. The piece of information used to execute functionality is called *message* and components decides which messages they will accept to execute the corresponding functionality.

As an example, figure 3 shows the *hand-made* component-based version of the hierarchy of Figure 2. The *Player* entity in the legacy class hierarchy shown in Figure 2 becomes a generic entity containing an instance of the *PlayerControllerComp*, *FightComp*, *PhysicsComp*, *SpeakerComp* and *GraphicsComp* components. Nonetheless, this relationship between entities and components is now done outside the code, usually in plain text files.

Such architecture promotes flexibility, reusability and extensibility but makes the code more difficult to understand, since now the behaviour of a given entity is built at run-time by linking components. The use of a distributed component-based architecture

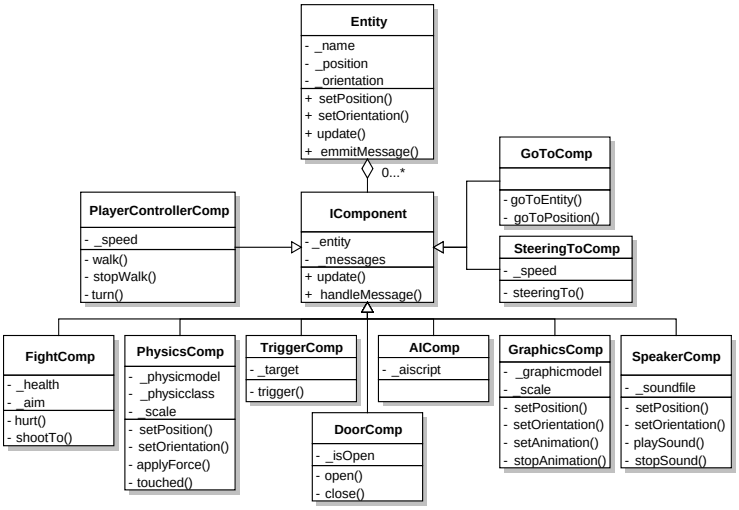


Fig. 3. A component-based architecture

may be confusing for programmers due to the loss of the class hierarchy where the entity distribution is seen at a glance. At the same time, compilers will lose important datatype information, decreasing the amount of compile time errors that will be able to detect.

3 Fighting Against Component Issues

When moving from inheritance hierarchies to component-based systems, the inheritance links are turned into aggregation links where the polymorphism is substituted by a delegation mechanism (delegating in components). This has been identified in [6] as *role aggregation* and it has its own drawbacks that we summarize in:

1. **Loss of information about the domain:** Despite all the issues class hierarchies manifest, they can be seen as a conceptual entity ontology and therefore they are a non-formal definition of the game model. In a component-based system a lot of semantic information becomes hidden behind components and scattered between them. Although in a component-based architecture is so easy to create a new entity, by enumerating in a plain text file which components it will have, entities are not related in between as in hierarchies. Even worse is that abstract classes (such as *RealEntity* in Figure 2) are inevitably lost because only entities that are meant to be used to create objects are modelled. The final result of this lack of structure is that the adoption of such design suffered the resistance of some programmers [5, 19].
2. **Entity inconsistencies:** With a data-driven architecture where creating new entity types is so easy, things may go really wrong, when declaring inconsistent entities that cannot work at execution time. When creating entities in hierarchies, it is obvious that some methods that execute some functionality may depends on other methods in the same entity or in ancestor classes (i.e. the *steeringTo* method of

the *ResourceBearer* in Figure 2 depends on the *applyForce* method of the *RealEntity*). If one invoked method does not belong to the entity, these inconsistency is checked easier, since the compiler provides some feedback. However, when working with components this inconsistency is not checked at compilation-time since method invocation is carried out through message passing. If the *ResourceBearer* has a *SteeringToComp* component (which may execute the *steeringTo* functionality) but does not have the *PhysicsComp* component where the *applyForce* resides, no error is fired at compilation-time. Even no error is produced at run-time, simply the functionality is never executed. On the other hand, when attribute values are also data-driven some error may arise at execution-time when the given value does not correspond with the expected type.

In order to alleviate these drawbacks, we have developed *Rosette* [14], a visual authoring tool that helps experts in the task of designing a game domain with a component-based architecture. Instead of decompose from the very begining all the entities in components, the tool promotes the use of conceptual hierarchies that put entities, components, messages and attributes in order. These conceptual hierarchies recover the possibility of seeing the entity distribution and hierarchy at a glance (solving this way the first of the identified issues) but it is also used to create a knowledge-rich representation of the game domain using OWL that brings a lot of semantic knowledge to light. With the purpose of giving feedback to the end user, *Rosette* use the *Pellet Integrity Constraint Validator (Pellet ICV)*[2] to detect constraint violations in the OWL formal domain and this way detect some domain inconsistencies, such as the ones mentioned previously.

The inner details of the process has been described in [14]. In short it ends with the OWL representation of the game domain and the user being confident about its lack of inconsistences. The next step, the one detailed here, is to find out the best set of components that should be implemented to build the entity hierarchy using a component based architecture. This work alleviate the transition from hierarchies to component-based architectures taking advantage of the modelled domain where experts only describes the entities of the game in a conceptual hierarchy. This way we are facilitating the development of component-based architectures for novice users in this area, more used to hierarchies, but also for expert programmers and designers that may use the system to accelerate the designing process. As we will see in the next section, the analisys of the domain and inference of the set of components is done by using FCA.

4 Generating components through Formal Concept Analysis

When using *Rosette*, the first step is to graphically specify the entity hierarchy of the game. Users must provide both the entities and their features (attributes and methods). For example, they will indicate that a *Player* can *walk* and *turn* whilst a *ResourceBearer* must be able to *goToPosition* or store an *_aiScript*.

Once done, we transform the hierarchy into a *formal context* in order to apply FCA. Entity types (*Player*, *ResourceBearer*) become *formal objects* and features (*walk*, *goToPosition*, *_aiScript*) become *formal attributes*. Therefore, our formal context $\langle G, M, I \rangle$

[2] http://clarkparsia.com/pellet/icv/

	setPosition	setAnimation	touched	_physicclass	_aiscript	...
Entity	■					...
Trigger	■		■	■		...
Persona	■	■		■		...
ResourceBearer	■	■		■	■	...

Table 1. Partial formal context of the game domain

is built in such a way that G contains every entity type and M will have every functionality and attribute. Finally, I is the binary (incidence) relation $I \subseteq G \text{x} M$, expressing which attributes of M describe each object of G or which objects are described using an attribute. This is filled in by going through the entity hierarchy annotating the relations between entity types and their features and considering inheritance (subclasses will include all the formal attributes of their parent classes). Table 1 shows a partial view of the formal context extracted from the hierarchy shown in Figure 2.

The application of FCA over such a formal context is done by using the Galicia project [18], a project usually used by end-users through their visual interface but that can be used by applications like *Rosette* using its API. The application of FCA gives us a set of formal concepts and their relationships, $\beta(G, M, I)$. Every formal concept represents all the entity types that form its extent and will need every functionality and attributes in its intent. Starting from the lattice (Figure 4) and with the goal of extrapolating the formal concepts to a programming abstraction, a naïve approach is to generate a hierarchy of classes with multiple inheritance. Unfortunately, the result is a class hierarchy that makes an extensive use of multiple inheritance, which is often considered as undesirable due to its complexity.

Therefore, though this approach of converting formal concepts into classes has been successfully used by others [12, 10], it is not valid in our context. After all, our final goal is to *remove* inheritance and to identify *common entity features* in order to create an independent class (a component) for each one. Once done, we will create just a single and generic *Entity* class that will keep a list of components, as promotes the component-based architecture. The addition of new features is done by adding new components to the entity, and these components are independent among themselves. The consequence of this independence is that, now, sharing the implementation of the same feature in different entities is not hard-wired in the code, but dynamically chosen in execution, avoiding the common problems in the long run of the rigid class hierarchies.

Using FCA, we reach this goal focusing not on the objects (reduced extents of the formal concepts), but *in the attributes* (reduced intents). The idea is based on the fact that when a formal concept has a non-empty *reduced intent* this means that the concept contributes with some attributes and/or functionality that did not have appeared so far (when traversing the lattice top-bottom). The immediate result is that the reduced extent

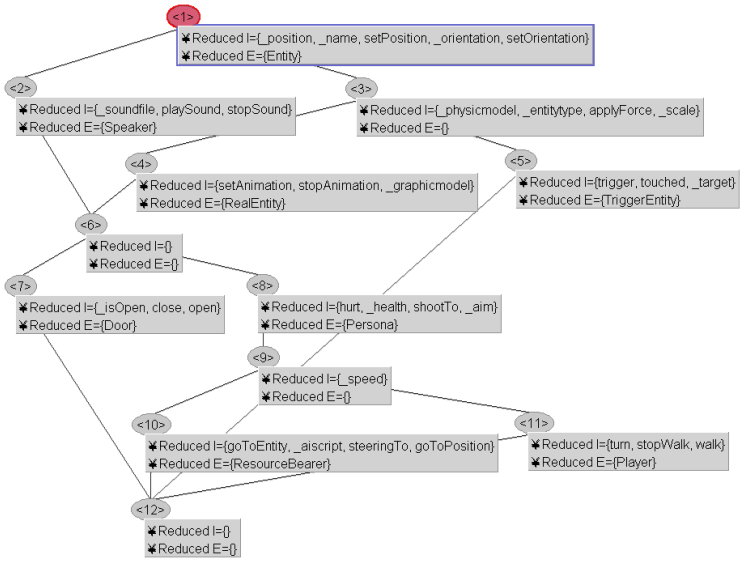


Fig. 4. Concept lattice

of objects differs from the objects in the superconcepts in those properties and it should be consider to build a component with them. At the same time, we will know that all the instances of the entities in the reduced extent of the formal concept will include the new created component.

For example, when analysing the formal concept labelled *11* in Figure 4, our technique will extract a new component containing the features *turn*, *stopWalk* and *walk*, and will annotate that all entities (generic instances) of the concept *Player* will need to include one of those components (and any other components extracted from the formal concepts over it).

The general process performed by *Rosette* with the hierarchy H created by the user is:

G = entity_types(H)
M = attributes_and_messages(H)
I = buildRelation(H)
L = $\beta(G, M, I)$
P = empty component list
for each formal concept C in L
  if C == $\top$ **then continue**
  B_r = reducedIntent(C)
  if B_r is empty **then continue**
  add(P, component(B_r))
end for

All the lines are self-explicative except that with the *add*. The *component* function receives the reduced intent of the formal concept and builds the component representation that has its attributes and functionalities.

In some cases, the top concept ($\top$) has a non-empty intent, so it would also generate a component with all its features (*name*, *position* and *orientation* in our example of Figure 4). That component would be added in *all entities* so, instead of keeping ourselves in a pure component-based architecture with an empty generic *Entity* class, we can move all those top features to it. Figure 5 shows the components extracted from *Rosette* using the lattice from Figure 4. The components have been automatically named concatenating each attribute name of the component or, when no one is available, by concatenating all the message names that the component is able to carry out. For example, let us say that the original name of the *FightComp* component was *C_health_aim*.

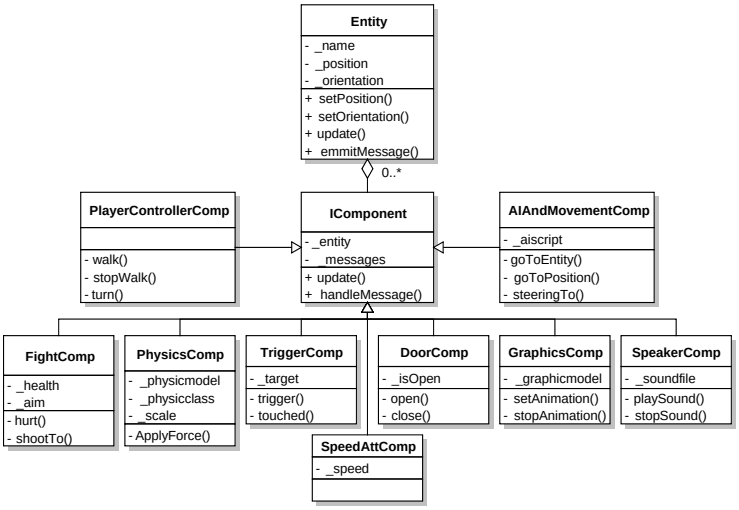


Fig. 5. The candidate components proposed by Rosette

Summarizing all the process, when analysing a concept lattice, every formal concept that provides a new feature (having no empty reduced intent) does not represent a new entity type but a new component. The only exception is the formal concept in the top of the lattice that represents the generic *entity* class, which has data and functionality shared by all the entity types. Both the generic entity and every new component have the ability of carrying out actions in the reduced intent of the formal concept and they are populated with corresponding attributes.

This way, we have easily obtained the candidate generic entity class and components, but we still have to describe the entity types. Starting from every concept which their reduced extents contain an entity type, *Rosette* uses the superconcept relation and goes up until reaching the concept in the top of the lattice. For example, the *Persona* entity type (Figure 4) would have components represented by formal concepts number

8, *4*, *3* and *2* (the number *6* has an empty *reduced intent* so it does not represent a component) whilst the *ResourceBearer* entity type would have the same components but also the number *10* and *9*. Obviously, components of every entity type are stored in the generic entity container represented by the formal concept number *1*.

Keep in mind that the final component distribution does not include information about what components are needed for each entity. This knowledge is not thrown away: *Rosette* stores all the information in the original lattice using OWL, which provides a knowledge-rich representation that will let it provide some extra functionalities described in the next sections.

5 Expert Tuning

The automatic process detailed above ends up with a collection of proposed components with a generated name, and the *Entity* base class that may have some common functionality. This result is presented to developers, who will be able to modify it using their prior experience. Some of the changes will affect to the underlying formal lattice (that is *never* shown to the users) in such a way that the relationship between it and the initial formal context extracted from the class hierarchy will be broken. At this stage of the process this does not represent an issue, because we will not use FCA anymore over it. On the other hand, changes could be so dramatic that the lattice could even become an invalid one. Fortunately, *Rosette* uses OWL as the underlying representation, that can be used to represent richer structures than mere partially ordered sets. In any case, for simplicity, in the rest of the paper we will keep talking about lattices although internally our tool will not be using them directly.

Users will be able to perform the next four operators over the proposed component distribution:

1. **Rename**: proposed components are automatically named according to their attribute names. The first operator users may perform is to rename them in order to clarify its purpose.
2. **Split**: in some cases, two functionalities not related to each other may end up in the same component due to the entity type definitions (FCA will group two functionalities when both of them appears together in every entity type created in the formal hierarchy). In that case, *Rosette* gives developers the chance of splitting them in two different components. The expert will then decide which features remain in the original component and which ones are moved to the new one (which is manually named). Formally speaking, this operator would modify the underlying concept lattice creating two concepts $(A1, B1)$ and $(A2, B2)$ that will have the same subconcepts and superconcepts than the original formal concept (A, B) where $A \equiv A1 \equiv A2$ and $B \equiv B1 \cup B2$. The original concept is removed. Although this is not correct mathematically speaking, since with this operation we do not have concepts anymore, we still use the term in this and in the other operators for simplicity.
3. **Move features**: this is the opposite operator. Sometimes some features lie in different components but the expert considers that they must belong to the same component. In this context, features of one component (some elements of the reduced

intent) can be transferred to a different component. In the lattice, this means that some attributes are moved from a node to another one. When this movement goes up-down (for example from node *9* to node *10*), *Rosette* will detect the possible inconsistency (entities extracted from node *11* would end with missed features) and warns the user to *clone* the feature also in the component generated from node *11*. If the developer moves all the features of a component the result is an useless and empty component that is therefore removed from the system.

4. **Add features**: some times features must be copied from one component to another one when FCA detects relationships that will not be valid in the long run. In our example, the dependency between node *3* and *4* indicates that all entities with a graphic model (*4*, *GraphicsComp*) will have physics (*3*, *PhysicsComp*), something valid in the initial hierarchy but that is likely to change afterwards. With the initial distribution, all graphical entities will have an *_scale* thanks to the physic component, but experts could envision that this should be a native feature of the *GraphicsComp* too. This operator let them to add those "missing" features to any component to avoid dependencies with other ones.

The expert interaction is totally necessary, first of all because she has to name the components but also because the system ignores some semantic knowledge and information based in the developer experience. However, the bigger the example is, with more entity types, the more alike is the proposed and the final set of components, just because the system has more knowledge to distribute responsibilities.

While using operators, coherence is granted because of the knowledge-rich OWL representation that contains semantic information about entities, components, and features (attributes and actions). This knowledge is useful while users tune the component distribution, but also to check errors in the domain and in future steps of the game development (as creating AIs that reason over the domain).

Once users validate the final distribution, *Rosette* generates a big amount of source code for all the components, that programmers will be fill up with the concrete behaviours.

5.1 Example

Figure 5 showed the resultant candidate of components proposed by *Rosette* for the hierarchy of Figure 1, that can now be manipulated by the expert to tune some aspects. The first performed changes are component rename (*rename* operator) that is, in fact, applied in the figure.

A hand-made component distribution of the original hierarchy would have ended with that one shown in Figure 3, that is quite similar to the distribution provided by *Rosette*. When using a richer hierarchy, both distributions are even more similar.

With the purpose of demonstrating how the expert would use the available operators to transform the proposed set of components, we apply some modifications to the automatically proposed distribution in order to turn it into the other one.

First of all, we can consider the *SpeedAttComp* that has the *_speed* attribute but no functionalities. In designing terms this is acceptable, but rarely has sense from the implementation point of view. *Speed* is used separately by *PlayerControllerComp* and

AIAndMovementComp to adjust the movement, so we will apply the *move features* operator moving (and cloning) the *_speed* feature to both components, and removing *SpeedAttComp* completely. This operator is coherent with the lattice (Figure 4): we are moving the intent of the node labelled *9* to both subconcepts (*10* and *11*).

After that, another application of the *move features* operator results in the movement of the *touched* message interpretation from the *TriggerComp* to the *PhysicsComp*. This is done for technical reasons in order to maintain all physic information in the same component.

Then, the *split* operator, which split components, is applied over the *AIAndMovementComp* component twice. Due to the lack of entity types in the example, some features resides in the same component though in the real implementation are divided. In the first application of the *split* operator, the *goToEntity* and the *goToPosition* message interpretations are moved to a new component, which is named *GoToComp*. The second application results in the new *SteeringToComp* component with the *steeringTo* message interpretation and the *_speed* attribute. The original component is renamed as *AIComp* by the *rename* operator and keeps the *_aiscript* attribute.

Finally, although the *Entity* class has received some generic features (from the top concept, $\top$), they are especially important in other components. Instead of just use those features from the entity, programmers would prefer to maintain them also in those other components. For this reason, we have to apply the *add features* operator over the *GraphicsComp*, *PhysicsComp* and *SpeakerComp* components in order to add the *setPosition* and the *setOrientation* functionalities to them.

6 Iterative Software Development with FCA

In the previous section we have presented a semi-automatic technique for moving from class hierarchies to components. The target purpose is helping programmers facing up to this kind of distributed system, which is widely used in computer game developments. Through the use of FCA, this technique splits entity behaviours in candidate components but also provides experts with mechanisms for modifying these component candidates. These mechanisms are the operators defined in Section 5, which execution in the domain alter somehow the underlying formal lattice generated during the FCA process.

Attentive readers will have realized that the previous technique is valid for the first step of the development but not for further development steps. Due to computer game requirements change throughout the game development, the entity distribution is always changing. When the experts face up to this situation, they may decide to change the entity hierarchy in order to use *Rosette* for generating a new set of components. The application of FCA results in a new lattice that probably does not change a lot from the previous one. However, the experts usually would have performed some modifications in the proposed component distribution using our operators. As the process is now repeated, these changes would be lost every time the expert request a new candidate set of components.

Our intention in this section is to extend the previous technique in order to allow an iterative software design. In this new approach, the modifications applied over one

lattice can be extrapolated to other lattices in future iterations. Keep in mind that the domain operators (Section 5) are applied over components that has been created from a formal concept. So, these operators could be applied on similar formal concepts, of another domain, in case that both domains share the part of the lattice affected by the operators.

From a high-level point of view, in order to preserve changes applied over the previous component suggestions, the system compares the new formal lattice, obtained through FCA, with the previous one. The methodology identifies the part of the lattice that does not significantly change between the two FCA applications. This way the tuning operators executed in concepts of this part of the lattice could be reapplied in the new lattice.

The identification of the target part of the lattice is a semi-automatic process, where formal concepts are related in pairs. *Rosette* automatically identifies the *constant* part of the lattice, which for our purpose is the set of pairs of formal concepts that have the same reduced intent. We do not care about the extent in our approach since the component suggestion lays its foundations in the reduced intent. The components extrated from the formal concepts that have not been matched up are presented to the expert. Then she can provide matches between old components and new ones to the considered *constant* part of the lattice.

It is worth mentioning that some of the operators could not be executed in the new domains due to component distribution may vary a lot after various domain iterations but it is just because these operators become obsoleted.

6.1 Example

In Section 5.1 FCA is applied to a hierarchy and the automatic part of the proposed methodology leads us to the set of components in Figure 5. The resultant domain was modified by the expert, by using the tuning operators, and the component-based system developed ends up with the components in Figure 3.

Now, let us recover the example and suppose that the game design has new requirements. The game designers propose the addition of two new entity types: the *BreakableDoor*, which is a door that can be broken using weapons, and a *Teleporter*, which moves entities that enter in them to a far target place. Designers also require the modification of the *ResourceBearer* entity, which must have a *currentEnemy* attribute for the artificial intelligence. The *Rosette* expert captures these domain changes by modifying the entity hierarchy and uses the component suggestion module to distribute responsibilities. The application of FCA to the current domain results in the lattice in Figure 6, where formal concepts are tagged with letters from *a* to *n*.

Comparing the new lattice with the lattice of the previous FCA application (Figure 4), *Rosette* determines that the pairs of formal concepts *<1,a>*, *<2,b>*, *<4,d>*, *<7,f>*, *<9,k>* and *<11,m>* remain from the previous to the current iteration. When *Rosette* finishes this automatic match, the formal concepts that were not put into pairs and with no empty reduced intent are presented in the screen. In this moment, the expert put the formal concepts *<3,c>*, *<5,e>*, *<8,j>* and *<10,l>* into pairs, based on their experience and in the fact that these concepts are very similar (only some attributes

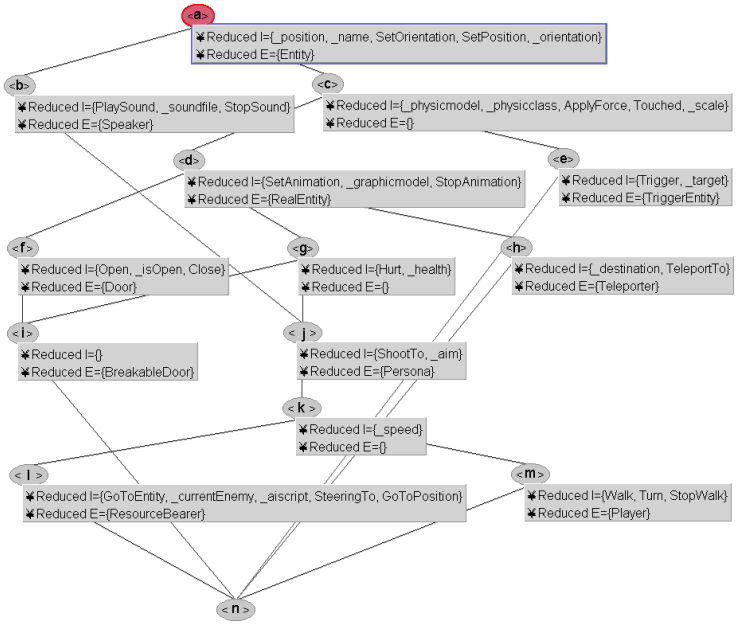


Fig. 6. New concept lattice

changes). Just the *g* and *h* formal concepts have no pairs and will become new components.

So, in these steps, the part of the lattice that does not significantly change has been identified and *Rosette* can extrapolate the modifications applied in the previous lattice to the new one. After applying the operators to the new domain, the new set of candidate components are finally given to the expert. Figure 7 shows these components, where we can compare the result with the components in Figure 5. The general structure is maintained but some actions and attributes has been moved between components. Furthermore two new components have arisen. The stressed features denote new elements (or moved ones) whilst the crossed out features mean that they do not belong to this component anymore (*FightComp*. At this point the expert could continue with the iteration by applying new operators to this set of components (i.e change the auto-generated names of the new components).

7 Related Work and Conclusions

Regarding related work, we can mention other applications of FCA to software engineering. The work described in [12] focuses on the use of FCA during the early phases of software development. They propose a method for finding or deriving class candidates from a given use case description. Also closely related is the work described in [10], where they propose a general framework for applying FCA to obtain a class hierarchy in different points of the software life-cycle: design from scratch using a set

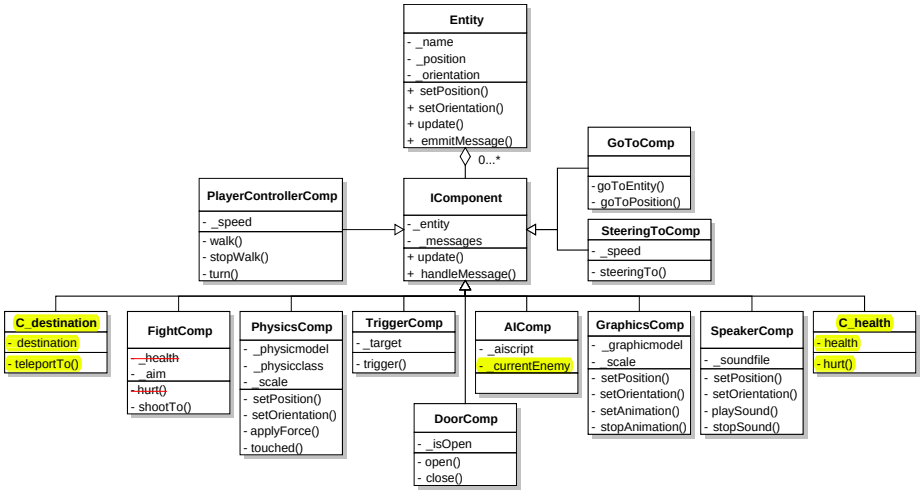


Fig. 7. The new candidate components proposed by Rosette

of class specifications, refactoring from the observation of the actual use of the classes in applications, and hierarchy evolution by incrementally adding new classes. The main difference with the approach presented here is that they try to build a class hierarchy while we intend to distribute functionality among sibling components, which solve the problem with multiple inheritance in FCA lattices.

The process of identifying components with FCA is not very different of identifying traits [13] and aspects [17]. In [13] Lienhard et al. present a process that identifies traits from inheritance hierarchies that is bases in the same principles than our system but is not exactly the same due to components are more autonomous pieces of software than traits. Components save their own state whilst traits are just a set of methods. However, which makes the difference between both proposals is the iterability.

A possible scenario for applying the techniques described in the paper is to re-engineer a game from class hierarchy to components. In the last years, we have been working on Javy 2 [11], a educational game that was initially developed using an entity hierarchy (a portion was shown in Figure 1), and afterwards manually converted to a component-based architecture (Figure 3). When *Rosette* was available, we tested it using the original Javy 2 hierarchy, and the initial component distribution was quite acceptable when compared with the human-made one. We could have saved a significant amount of time if it had been available on time.

In the long term, our goal is to support the up-front development of games with a component-based architecture where entities are connected to a logical hierarchical view. In this paper we have shown how we allow an iterative process when defining the class hierarchy, so operators applied to the early versions of the component distribution are automatically reapplied in the late ones. Nevertheless, more work must be done in the code generation phase to do it reversible. Changes in the autogenerated source code

are still, unfortunately, out of the scope of *Rosette* so they must be manually redone for each class hierarchy iteration.

References

1. K. Beck. Embracing change with extreme programming. *Computer*, 32:70–77, October 1999.
2. K. Beck and C. Andres. *Extreme Programming Explained: Embrace Change (2nd Edition)*. Addison-Wesley Professional, 2004.
3. G. Birkhoff. *Lattice Theory, third editon*. American Math. Society Coll. Publ. 25, Providence, R.I, 1973.
4. W. Buchanan. *Game Programming Gems 5*, chapter A Generic Component Library. Charles River Media, 2005.
5. M. Chady. Theory and practice of game object component architecture. In *Game Developers Conference*, 2009.
6. M. Dao, M. Huchard, T. Libourel, A. Pons, and J. Villerd. Proposals for Multiple to Single Inheritance Transformation. In *MASPEGHI'04: 3rd Workshop on Managing SPEcialization/Generalization Hierarchies*, pages 21–26, Oslo (Norway), 2004.
7. S. Ducasse, O. Nierstrasz, N. Schärli, R. Wuyts, and A. P. Black. Traits: A mechanism for fine-grained reuse. *ACM Trans. Program. Lang. Syst.*, 28:331–388, March 2006.
8. B. Ganter and R. Wille. Formal concept analysis. *Mathematical Foundations*, 1997.
9. S. Garcés. *AI Game Programming Wisdom III*, chapter Flexible Object-Composition Architecture. Charles River Media, 2006.
10. R. Godin and P. Valtchev. *Formal Concept Analysis*, chapter Formal Concept Analysis-Based Class Hierarchy Design in Object-Oriented Software Development, pages 304–323. Springer Berlin / Heidelberg, 2005.
11. P. P. Gómez-Martín, M. A. Gómez-Martín, P. A. González-Calero, and P. Palmier-Campos. Using metaphors in game-based education. In K. chuen Hui, Z. Pan, R. C. kit Chung, C. C. Wang, X. Jin, S. Göbel, and E. C.-L. Li, editors, *Technologies for E-Learning and Digital Entertainment. Second International Conference of E-Learning and Games (Edutainment'07)*, volume 4469 of *Lecture Notes in Computer Science*, pages 477–488. Springer Verlag, 2007.
12. W. Hesse and T. A. Tilley. *Formal Concept Analysis used for Software Analysis and Modelling*, volume 3626 of *LNAI*, pages 288–303. Springer, 2005.
13. A. Lienhard, S. Ducasse, and G. Arévalo. Identifying traits with formal concept analysis. In *Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering*, ASE '05, pages 66–75, New York, NY, USA, 2005. ACM.
14. D. Llansó, M. A. Gómez-Martín, P. P. Gómez-Martín, and P. A. González-Calero. Explicit domain modelling in video games. In *International Conference on the Foundations of Digital Games (FDG)*, Bordeaux, France, June 2011. ACM.
15. B. Rene. *Game Programming Gems 5*, chapter Component Based Object Management. Charles River Media, 2005.
16. K. Schwaber and M. Beedle. *Agile Software Development with Scrum*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 2001.
17. T. Tourwe and K. Mens. Mining aspectual views using formal concept analysis. In *Proceedings of the Source Code Analysis and Manipulation, Fourth IEEE International Workshop*, pages 97–106, Washington, DC, USA, 2004. IEEE Computer Society.
18. P. Valtchev, D. Grosser, C. Roume, and M. R. Hacene. Galicia: An open platform for lattices. In *In Using Conceptual Structures: Contributions to the 11th Intl. Conference on Conceptual Structures (ICCS'03*, pages 241–254. Shaker Verlag, 2003.
19. M. West. Evolve your hiearchy. *Game Developer*, 13(3):51–54, Mar. 2006.

Fuzzy-Valued Triadic Implications

Cynthia Vera Glodeanu

Technische Universität Dresden,
01062 Dresden, Germany
Cynthia_Vera.Glodeanu@mailbox.tu-dresden.de

Abstract. We present a new approach for handling fuzzy triadic data in the setting of Formal Concept Analysis. The starting point is a fuzzy-valued triadic context (K_1, K_2, K_3, Y), where K_1, K_2 and K_3 are sets and Y is a ternary fuzzy relation between these sets. First, we generalise the methods of Triadic Concept Analysis to our setting and show how they fit other approaches to Fuzzy Triadic Concept Analysis. Afterwards, we develop the fuzzy-valued triadic implications as counterparts of the various triadic implications studied in the literature. These are of major importance for the integrity of Fuzzy and Fuzzy-Valued Triadic Concept Analysis.

Keywords: Formal Concept Analysis, fuzzy data, three-way data

1 Introduction

So far, the fuzzy approaches to Triadic Concept Analysis considered all three components of a triadic concept as fuzzy sets. In [1] the methods from Triadic Concept Analysis were generalised to the fuzzy setting. A more general approach was presented in [2], where different residuated lattices were considered for each fuzzy set. A somehow different strategy was considered in [3] using alpha-cuts.

Our approach differs from the other ones in considering just two components as fuzzy and one as crisp in a triadic concept. This is motivated by the fact that in some situations it is not appropriate to regard all sets as fuzzy. For example, it is not natural to say that *half of a person is old*, however we may say *a person is half old*. First, we translate methods of Triadic Concept Analysis to our setting. Compared to other works, we generalise all triadic derivation operators and show how they change for the fuzzy approaches considered by other authors. Besides these results, the main achievement of this paper is the generalisation of the various triadic implications presented in [4].

Due to the large amount of results in this paper, we concentrate on giving an intuition of the methods and omit proofs whenever they do not influence the understanding. The missing proofs, further results concerning fuzzy-valued triadic concepts and trilattices can be found in [5]. There, we also study the fuzzy-valued triadic approach to Factor Analysis.

The paper is structured as follows: In Section 2 we give brief introductions to Triadic and Formal Fuzzy Concept Analysis. In Section 3 we develop our

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

fuzzy-valued setting, defining context, concept, derivation operators and show how they correspond other to approaches to Fuzzy Triadic Concept Analysis. We also comment on the reasons why our setting is a proper generalisation. In Section 4 we present the fuzzy-valued triadic implications. The developed methods are accompanied by illustrative examples. The last section contains concluding remarks and further topics of research.

2 Preliminaries

We assume basic familiarities with Formal Concept Analysis and refer the reader to [6]. In the following we give brief introductions to Triadic Concept Analysis [7, 8] and Formal Fuzzy Concept Analysis [9, 10].

2.1 Triadic Concept Analysis

As introduced in [7], the underlying structure of Triadic Concept Analysis is a **triadic context** defined as a quadruple (K_1, K_2, K_3, Y) where K_1, K_2 and K_3 are sets and Y is a ternary relation, i.e., $Y \subseteq K_1 \times K_2 \times K_3$. The elements of K_1, K_2 and K_3 are called **(formal) objects, attributes** and **conditions**, respectively, and $(g, m, b) \in Y$ is read: *object g has attribute m under condition b*. A **triadic concept** (shortly **triconcept**) of a triadic context (K_1, K_2, K_3, Y) is defined as a triple (A_1, A_2, A_3) with $A_i \subseteq K_i$, $i \in \{1, 2, 3\}$ that is maximal with respect to component-wise set inclusion. For a triconcept (A_1, A_2, A_3), the components A_1, A_2 and A_3 are called the **extent**, the **intent**, and the **modus** of (A_1, A_2, A_3), respectively.

Small triadic contexts can be represented through three-dimensional cross tables (see Example 1). Pictorially, a triconcept is a rectangular box full of crosses in the three-dimensional cross table representation of (K_1, K_2, K_3, Y), where this "box" is maximal under proper permutation of rows, columns and layers of the cross table.

For $\{i, j, k\} = \{1, 2, 3\}$ with $j < k$ and for $X \subseteq K_i$ and $Z \subseteq K_j \times K_k$, the $(-)^{(i)}$**-derivation operators** are defined by

$$X \mapsto X^{(i)} := \{(k_j, k_k) \in K_j \times K_k \mid (k_i, k_j, k_k) \in Y \text{ for all } k_i \in X\}, \quad (1)$$

$$Z \mapsto Z^{(i)} := \{k_i \in K_i \mid (k_i, k_j, k_k) \in Y \text{ for all } (k_j, k_k) \in Z\}. \quad (2)$$

These derivation operators correspond to the derivation operators of the dyadic contexts defined by $\mathbb{K}^{(i)} := (K_i, K_j \times K_k, Y^{(i)})$ for $\{i, j, k\} = \{1, 2, 3\}$, where $k_1 Y^{(1)} (k_2, k_3) :\Longleftrightarrow k_2 Y^{(2)} (k_1, k_3) :\Longleftrightarrow k_3 Y^{(3)} (k_1, k_2) :\Longleftrightarrow (k_1, k_2, k_3) \in Y$. Due to the structure of triadic contexts further derivation operators can be defined. For $\{i, j, k\} = \{1, 2, 3\}$ and for $X_i \subseteq K_i$, $X_j \subseteq K_j$ and $X_k \subseteq K_k$ the $(-)^{X_k}$**-derivation operators** are defined by

$$X_i \mapsto X_i^{X_k} := \{k_j \in K_j \mid (k_i, k_j, k_k) \in Y \text{ for all } (k_i, k_k) \in X_i \times X_k\}, \quad (3)$$

$$X_j \mapsto X_j^{X_k} := \{k_i \in K_i \mid (k_i, k_j, k_k) \in Y \text{ for all } (k_j, k_k) \in X_j \times X_k\}. \quad (4)$$

These derivation operators correspond to the derivation operators of the dyadic contexts defined by $\mathbb{K}^{ij}_{X_k} := (K_i, K_j, Y^{ij}_{X_k})$ where $(k_i, k_j) \in Y^{ij}_{X_k}$ if and only if $(k_i, k_j, k_k) \in Y$ for all $k_k \in X_k$. The structure on the set of all triconcepts $\mathfrak{T}(\mathbb{K})$ is the set inclusion in each component of the triconcept. For each $i \in \{1, 2, 3\}$ there is a quasiorder $\lesssim_i$ and its corresponding equivalence relation $\sim_i$ defined by

$$(A_1, A_2, A_3) \lesssim_i (B_1, B_2, B_3) :\Longleftrightarrow A_i \subseteq B_i \text{ and}$$
$$(A_1, A_2, A_3) \sim_i (B_1, B_2, B_3) :\Longleftrightarrow A_i = B_i \ (i = 1, 2, 3).$$

The triconcepts ordered in this way form complete *trilattices*, the triadic counterparts of concept lattices, as proved in the Basic Theorem of Triadic Concept Analysis [8]. However, unlike the dyadic case, the extents, intents and modi, respectively, do not form a closure system in general.

Example 1. The triadic context displayed below consists of the object set $K_1 = \{1, 2, 3\}$, the attribute set $K_2 = \{a, b, c\}$ and the condition set $K_3 = \{A, B\}$. The context has 12 triconcepts which are displayed in the same figure on the right. For example, the first concept means that object 1 has attributes a and b under

	A			B		
	a	b	c	a	b	c
1	×	×		×	×	
2		×	×	×		×
3		×	×	×	×	×

No.	Extent	Intent	Modus	No.	Extent	Intent	Modus
1	$\{1\}$	$\{a, b\}$	$\{K_3\}$	7	$\{3\}$	$\{K_2\}$	$\{B\}$
2	$\{K_1\}$	$\{b\}$	$\{A\}$	8	$\{K_1\}$	$\{a\}$	$\{B\}$
3	$\{2, 3\}$	$\{b, c\}$	$\{A\}$	9	$\{2, 3\}$	$\{c\}$	$\{K_3\}$
4	$\{\varnothing\}$	$\{K_2\}$	$\{K_3\}$	10	$\{3\}$	$\{b, c\}$	$\{K_3\}$
5	$\{1, 3\}$	$\{a, b\}$	$\{B\}$	11	$\{K_1\}$	$\{K_2\}$	$\{\varnothing\}$
6	$\{2, 3\}$	$\{a, c\}$	$\{B\}$	12	$\{K_1\}$	$\{\varnothing\}$	$\{K_3\}$

Fig. 1. Triadic context and the associated triconcepts

all conditions from K_3. However, as two components of a triconcept are necessary to determine the third one, $\{a, b\}$ is also an intent of another triconcept, namely of the fifth one.

2.2 Formal Fuzzy Concept Analysis

A **complete residuated lattice** $\mathbf{L} := (L, \wedge, \vee, \otimes, \to, 0, 1)$ is an algebra such that: (1) $(L, \wedge, \vee, 0, 1)$ is a complete lattice, (2) $(L, \otimes, 1)$ is a commutative monoid, (3) 0 is the least and 1 the greatest element, (4) the adjointness property holds for all $a, b, c \in L$, i.e., $a \otimes b \leq c \Leftrightarrow a \leq b \to c$. Then, $\otimes$ is called **multiplication**, $\to$ **residuum** and $(\otimes, \to)$ **adjoint couple**. Each of the following adjoint couples make $\mathbf{L}$ a complete residuated lattice:

Lukasiewicz: $a \otimes b := max(0, a + b - 1)$ with $a \to b := min(1,\ 1 - a + b)$

Gödel: $a \otimes b := min(a, b)$ with $a \to b := \begin{cases} 1, & a \leq b \\ b, & a \not\geq b \end{cases}$

Product: $a \otimes b := ab$ with $a \to b := \begin{cases} 1, & a \leq b \\ b/a, & a \not\geq b \end{cases}$

The **hedge operator** is defined as a unary function $^* : L \to L$ which satisfies the following properties: (1) $1^* = 1$, (2) $a^* \leq a$, (3) $(a \to b)^* \leq a^* \to b^*$, and (4) $a^{**} = a^*$. Typical examples are the *identity*, i.e., for all $a \in L$ it holds that $a^* = a$, and the *globalization*, i.e., $a^* = 0$ for all $a \in L \setminus \{1\}$ and $a^* = 1$ if and only if $a = 1$.

A triple (G, M, I) is called a **formal fuzzy context** if $I : G \times M \to L$ is a fuzzy relation between the sets G and M and L is the support set of some residuated lattice. Elements from G and M are called **objects** and **attributes**, respectively. The fuzzy relation I assigns to each $g \in G$ and each $m \in M$ a truth degree $I(g, m) \in L$ to which the object g has the attribute m. For fuzzy sets $A \in L^G$ and $B \in L^M$ the **derivation operators** are defined by

$$A^{\shortmid}(m) := \bigwedge_{g \in G} (A(g)^* \to I(g, m)),\ B^{\shortmid}(g) := \bigwedge_{m \in M} (B(m) \to I(g, m)), \qquad (5)$$

for $g \in G$ and $m \in M$. Then, $A^{\shortmid}(m)$ is the truth degree of the statement "m is shared by all objects from A" and $B^{\shortmid}(g)$ is the truth degree of "g has all attributes from B". For now, we take for * the identity. It plays an important role in the computation of the stem base, as we will see later.

A **fuzzy concept** is a tuple $(A, B) \in L^G \times L^M$ such that $A^{\shortmid} = B$ and $B^{\shortmid} = A$. Then, A is called the **(fuzzy) extent** and B the **(fuzzy) intent** of (A, B). Fuzzy concepts represent maximal rectangles with truth values different from zero in the fuzzy context. The fuzzy concepts ordered by the fuzzy set inclusion form fuzzy concept lattices [9, 10]. Taking in (5) for * hedges different from the identity, we obtain the so-called fuzzy concept lattices with hedges [11].

Example 2. The fuzzy context displayed below has the object set $G = \{x, y, z\}$, the attribute set $M = \{a, b, c, d\}$ and the set of truth values is the 3-element chain $L = \{0, 0.5, 1\}$. Using the Gödel logic and the derivation operators defined in Equation 5 with the hedge * being the identity we obtain 10 fuzzy concepts. For example $(\{1, 0.5, 0\}, \{1, 1, 0, 0\})$ is a fuzzy concept. The extent contains the truth values of each object belonging to the extent, i.e., in this case x belongs fully to the set, y belongs to it with a truth value 0.5 and z does not belong to the extent. Similar affirmations can be done for the intent. Using the Lukasiewicz logic in the same setting we obtain 13 fuzzy concepts. On this set of truth values the only possible hedge operators are the identity and globalization. As one of the major roles of the hedge operators is to control the size of the fuzzy concept lattice, the

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

number of fuzzy concepts will be smaller, when using in (5) a hedge different from the identity. In our example, using the globalization operator as the hedge, we obtain 6 fuzzy concepts both with the Gödel and Lukasiewicz logic. As we will see immediately, the hedges play also an important role for the attribute implications, especially for the stem base.

Fuzzy implications were studied in a series of papers by R. Belohlavek and V. Vychodil, as for example in [12, 13]. For fuzzy sets $A, B \in L^X$ the **subsethood degree** of A being a subset of B is given by $tv(A \subseteq B) = \bigwedge_{x \in X}(A(x) \to B(x))$. Let A and B be fuzzy attribute sets, then the truth value of the implication $A \to B$ is given by

$$\begin{aligned} tv(A \to B) &:= tv(\forall g \in G((\forall m \in A, (g,m) \in I) \to (\forall n \in B, (g,n) \in I))) \\ &= \bigwedge_{g \in G} \Big(\bigwedge_{m \in M} (A(m) \to I(g,m)) \to \bigwedge_{n \in M} (B(n) \to I(g,n)) \Big) \\ &= tv(B \subseteq A^{\shortparallel}). \end{aligned}$$

Example 3. Let us go back to our fuzzy context from Example 2. Consider the Gödel logic and the derivation operators from (5) with * being the identity. Then, $b(1)^{\shortparallel} = \{1, 0.5, 0\}^{\shortparallel} = \{1, 1, 0, 0\}$. Now, $tv(b(1) \to a(1)) = tv(\{a(1)\} \subseteq b(1)^{\shortparallel}) = 1$ and $tv(b(1) \to \{a(1), c(0.5)\}) = 0$ because $c(0.5) \notin b(1)^{\shortparallel}$. On the other hand, considering in (5) the globalization as the hedge, we obtain $b(1)^{\shortparallel} = \{1, 0.5, 0\}^{\shortparallel} = \{1, 1, 0.5, 0\}$ and therefore $tv(b(1) \to \{a(1), c(0.5)\}) = 1$. Yet another example is $tv(b(0.5) \to b(1)) = tv(\{b(1)\} \subseteq b(0.5)^{\shortparallel}) = tv(\{b(1)\} \subseteq \{1, 0.5, 0, 0\}) = 0.5$.

Due to the large number of implications in a fuzzy and even in a crisp formal context, one is intrested in the stem base of the implications. The **stem base** is a set of implications which is non-redundant and complete. The existence and construction of the stem base for the discrete case was studied in [14], see also [6]. The problem for the fuzzy case was studied in [13]. There, the authors showed that using in (5) the globalization, the stem base of a fuzzy context is uniquely determined. Using hedges different from the globalization, a fuzzy context may have more than one stem base.

3 Fuzzy-Valued Triadic Concept Analysis

Now, we are ready to develop our fuzzy-valued triadicsetting. We will define fuzzy-valued triadic contexts, concepts and derivation operators.

For a triadic context $\mathbb{K} = (K_1, K_2, K_3, Y)$ a **dyadic-cut** (shortly **d-cut**) is defined as $c^i_\alpha := (K_j, K_k, Y^{jk}_\alpha)$, where $\{i, j, k\} = \{1, 2, 3\}$ and $\alpha \in K_i$. A d-cut is actually a special case of $\mathbb{K}^{ij}_{X_k} = (K_i, K_j, Y^{ij}_{X_k})$ for $X_k \subseteq K_k$ and $|X_k| = 1$. Each d-cut is itself a dyadic context.

Definition 1. *A **fuzzy-valued triadic context** (**f-valued triadic context**) is a quadruple $\mathbb{K} := (K_1, K_2, K_3, Y)$, where Y is a ternary fuzzy relation between the sets K_i with $i \in \{1, 2, 3\}$, i.e., $Y : K_1 \times K_2 \times K_3 \to L$ and L is the support set*

of some residuated lattice. The elements of K_1, K_2 and K_3 are called ***objects****,* ***attributes*** *and* ***conditions****, respectively. To every triple $(k_1, k_2, k_3) \in K_1 \times K_2 \times K_3$, Y assigns a truth value $tv_{k_3}(k_1, k_2)$ to which object k_1 has attribute k_2 under condition k_3.*

The f-valued triadic context can be represented as a three-dimensional table, the entries of which are fuzzy values (see Example 4). In $\mathbb{K}$ one can interchange the roles played by the sets K_1, K_2 and K_3 requiring, for example, that Y assigns to every triple (k_2, k_3, k_1) a truth value $tv_{k_1}(k_2, k_3)$ to which attribute k_2 exists under condition k_3 having object k_1.

Definition 2. *A* ***fuzzy-valued triadic concept*** *(shortly* ***f-valued triconcept****) of an f-valued triadic context (K_1, K_2, K_3, Y) is a triple (A_1, A_2, A_3) with $A_1 \subseteq L^{K_1}$, $A_2 \subseteq L^{K_2}$ and $A_3 \subseteq K_3$ that is maximal with respect to componentwise set inclusion. The components A_1, A_2 and A_3 are called (****f-valued****) **extent**, (****f-valued****) **intent**, and the* ***modus*** *of (A_1, A_2, A_3), respectively. We denote by $\mathfrak{T}(\mathbb{K})$ the set of all f-valued triconcepts.*

This definition immediately implies that the d-cut $(K_1, K_2, Y^{12}_{k_3})$ is a fuzzy context for every $k_3 \in K_3$.

Example 4. We consider an f-valued triadic context with values from the 3-element chain $\{0, 0.5, 1\}$. The object set $K_1 = \{1, 2, 3, 4, 5\}$ contains 5 groups of students, the attribute set $K_2 = \{f, s, v\}$ contains 3 feelings, namely, fevered (f), serious (s), vigilant (v) and the condition set $K_3 = \{E, P, F\}$ contains the events: Doing an exam (E), giving a presentation (P) and meeting friends (F). Using the Lukasiewicz logic, we obtain 30 f-valued triconcepts and with the Gödel

	E			P			F		
	f	s	v	f	s	v	f	s	v
1	1	1	1	1	0.5	0.5	0	0.5	1
2	1	0.5	1	0.5	0	0	0	0	0.5
3	0.5	0.5	0.5	0.5	0.5	0	0	0	0.5
4	0.5	0	0.5	0.5	0.5	0.5	0	0.5	0.5
5	1	1	1	1	0.5	0.5	0	0.5	1

Fig. 2. F-valued triadic context

logic 34. For example, $(\{1, 1, 0.5, 0, 0\}, \{1, 0.5, 1\}, \{E\})$ is an f-valued triconcept meaning that while doing an exam the first two student groups and half of the third one are fevered, vigilant and moderately serious. Another example is $(\{1, 1, 1, 1, 1\}, \{0.5, 0, 0\}, \{E, P\})$ meaning that all students are moderately fevered while giving a presentation. Yet another example is $(\{1, 0, 0, 0.5, 1\}, \{1, 0, 0.5\}, \{E, P\})$ signifying that the first, the last and half of the 4-th group of students are fevered and moderately vigilant while doing an exam and giving a presentation.

Lemma 1. *Every f-valued triadic context is isomorphic to a triadic context.*

Proof. According to [9], every fuzzy context is isomorphic to a formal context, namely to its double-scaled context. Every condition d-cut is a fuzzy context. By double-scaling each condition d-cut we obtain the corresponding double-scaled triadic context $\widetilde{\mathbb{K}}$ for an f-valued triadic context $\mathbb{K}$.

Formally, suppose that $\widetilde{\mathbb{K}} := (K_1^+, K_2^+, K_3, \widetilde{Y})$ is the double-scaled triadic context of $\mathbb{K} := (K_1, K_2, K_3, Y)$, the construction of which is given below. We have to show that the considered isomorphism is given by

$$\varphi : \underline{\mathfrak{T}}(\mathbb{K}) \to \underline{\mathfrak{T}}(\widetilde{\mathbb{K}}) \text{ with } \varphi(A_1, A_2, A_3) := (A_1^+, A_2^+, A_3)$$

and that the inverse map is given by

$$\psi : \underline{\mathfrak{T}}(\widetilde{\mathbb{K}}) \to \underline{\mathfrak{T}}(\mathbb{K}) \text{ with } \psi(A_1, A_2, A_3) := (A_1^\diamondsuit, A_2^\diamondsuit, A_3).$$

Therefore, we have to prove the following statements: For all f-valued triconcepts $(A_1, A_2, A_3), (B_1, B_2, B_3) \in \underline{\mathfrak{T}}(\mathbb{K})$ and for all $(X_1, X_2, X_3) \in \underline{\mathfrak{T}}(\widetilde{\mathbb{K}})$:

$$\begin{aligned}
&\varphi(A_1, A_2, A_3) \in \underline{\mathfrak{T}}(\widetilde{\mathbb{K}}),\\
&\psi(X_1, X_2, X_3) \in \underline{\mathfrak{T}}(\mathbb{K}),\\
&\psi\varphi(A_1, A_2, A_3) = (A_1, A_2, A_3),\\
&\varphi\psi(X_1, X_2, X_3) = (X_1, X_2, X_3),\\
&(A_1, A_2, A_3) \lesssim_i (B_1, B_2, B_3) \Leftrightarrow \varphi(A_1, A_2, A_3) \lesssim_i \varphi(B_1, B_2, B_3),
\end{aligned}$$

for all $i \in \{1, 2, 3\}$. These statements can be proven by basic properties of fuzzy sets and triadic derivation operators. Due to limitation of space, we skip the proof. □

We present the construction of $\widetilde{\mathbb{K}} := (K_1^+, K_2^+, K_3, \widetilde{Y})$, the double-scaled triadic context, for a given f-valued triadic context $\mathbb{K} = (K_1, K_2, K_3, Y)$. Let $X_i \subseteq L^{K_i}$ with $i \in \{1, 2\}$ and let L be the support set of some residuated lattice. We define

$$\begin{aligned}
X_i^+ &:= \{(k_i, \mu) \mid k_i \in K_i, \mu \in L, \mu \leq X_i(k_i)\} \subseteq K_i^* := K_i \times L,\\
X_i^\diamondsuit &:= \bigvee \{\mu \mid (k_i, \mu) \in X_i\} \subseteq L^{K_i}.
\end{aligned}$$

Then, $\widetilde{Y} \subseteq K_1^+ \times K_2^+ \times K_3$ and

$$((k_1, \mu), (k_2, \lambda), k_3) \in \widetilde{Y} :\Longleftrightarrow \mu \otimes \lambda \leq tv_{k_3}(k_1, k_2) \Longleftrightarrow \mu \otimes \lambda \leq Y(k_1, k_2, k_3).$$

According to the above lemma, the f-valued triadic contexts fulfill all the properties the triadic contexts have. The f-valued triconcepts ordered by the (fuzzy) set inclusion form a complete fuzzy trilattice. Due to limitation of space we omit the proofs.

For our f-valued setting we want to obtain the corresponding $(-)^{A_k}$ and $(-)^{(i)}$ derivation operators. However, these can be defined in various ways. We

distinguish between more cases for the $(-)^{(i)}$-derivation operators. In case of the $(-)^{(i)}$**-derivation operators** with $Z = X_j \times X_3 \subseteq L^{K_j} \times K_3$ and $X_i \subseteq L^{K_i}$ for $\{i, j\} = \{1, 2\}$ the situation is easy. They are defined as

$$Z \mapsto Z^{(i)} := \bigwedge_{k_j \in K_j} \{X_j(k_j) \to Y^{(i)}(k_i, (k_j, k_3)) \mid \forall k_3 \in K_3\},$$
$$X_i \mapsto X_i^{(i)} := (T_{l_3}, \{k_3 \in K_3 \mid T_{k_3} \subseteq T_{l_3}\}) \text{ for } l_3 \in K_3,$$

where $T_{l_3} := \bigwedge_{k_i \in K_i} (X_i(k_i) \to Y^{(i)}(k_i, (k_j, l_3))$ with the derivation operators from the fuzzy dyadic context $\mathbb{K}^{(i)} := (K_i, K_j \times K_3, Y^{(i)})$ and $Y^{(i)}(k_i, (k_j, k_3)) := Y(k_i, k_j, k_3)$. The $(-)^{(3)}$**-derivation operator** for $Z := X_1 \times X_2 \subseteq L^{K_1} \times L^{K_2}$ and $X_3 \subseteq K_3$ is defined by

$$Z \mapsto Z^{(3)} : = \{k_3 \in K_3 \mid k_1 \otimes k_2 \leq tv_{k_3}(k_1, k_2), \forall (k_1, k_2) \in Z\} \tag{6}$$
$$= \bigwedge_{(k_1,k_2) \in K_1 \times K_2} (Z(k_1, k_2) \to Y^{(3)}((k_1, k_2), k_3))^*, \tag{7}$$

where $Z(k_1, k_2) := X_1(k_1) \otimes X_2(k_2)$, * is the globalization in order to assure that $Z^{(3)}$ is crisp and we have the dyadic fuzzy context $\mathbb{K}^{(3)} := (K_1 \times K_2, K_3, Y^{(3)})$ with $Y^{(3)}((k_1, k_2), k_3) := Y(k_1, k_2, k_3)$. We search for the conditions which contain the maximal rectangle generated by Z.

The situation for $X_3^{(3)}$ is quite tricky. Applying the derivation operators in $\mathbb{K}^{(3)}$ for X_3, we get a truth value $l \in L$ such that $l = k_1 \otimes k_2$ instead of a tuple (k_1, k_2). To obtain such a tuple, we first have to compute the double-scaled context $\widetilde{\mathbb{K}}$. Afterwards, we use the crisp $(-)^{(3)}$-derivation operator in $\widetilde{\mathbb{K}}$ to find the components of the triconcept. Finally, we transform these into fuzzy sets as described in the construction of $\widetilde{\mathbb{K}}$. This way, we obtain the tuples $((k_1, \mu), (k_2, \nu))$ consisting of objects and attributes with their truth values instead of the truth value $k_1 \otimes k_2$.

For other approaches of fuzzy triadic data the derivation operators given in (7) and the above construction suffice for any $(-)^{(i)}$ derivation operator.

Proposition 1. *The $(-)^{(i)}$-derivation operators with $i \in \{1, 2, 3\}$ yield f-valued triconcepts.*

Proof. Suppose $X_1 \subseteq L^{K_1}, X_2 \subseteq L^{K_2}$ and $X_3 \subseteq K_3$. We have $X_1^{(1)} = (T_{l_3}, \{k_3 \mid T_{k_3} \subseteq T_{l_3}\})$, where

$$T_{l_3} = \bigwedge_{k_1 \in K_1} (X_1(k_1) \to Y^{(1)}(k_1, (k_2, l_3)))$$
$$= \bigwedge_{k_1 \in K_1} (X_1(k_1) \to Y_{l_3}^{12}(k_1, k_2)).$$

Since $\mathbb{K}_{l_3}^{12}$ is a dyadic fuzzy context, $(X_1, T_{l_3}) =: (X_1, A_2)$ is a fuzzy preconcept in $\mathbb{K}_{l_3}^{12}$, i.e., $X_1^{\downarrow} \subseteq A_2$ and $A_2^{\downarrow} \subseteq X_1$ with the derivation operators of $\mathbb{K}_{l_3}^{12}$ given

by Equation (5). In particular we have $X_1^{\shortmid} \subseteq A_2$. For any $k_3 \in K_3$ if $T_{k_3} \subseteq T_{l_3}$, then (X_1, A_2) is a fuzzy preconcept also in $\mathbb{K}^{12}_{l_3 \cup k_3}$. Proceeding alike, we obtain the largest set $A_3 \subseteq K_3$ containing l_3 such that $T_{A_3} \subseteq T_{l_3}$. Then, (X_1, A_2) is a fuzzy preconcept in $\mathbb{K}^{12}_{A_3}$. So far, we obtained the last two components of the f-valued triconcept and apply on them the $(-)^{(1)}$-derivation operator to obtain the first one. Now, we have

$$\begin{aligned}(A_2 \times A_3)^{(1)} &= \bigwedge_{k_2 \in K_2} \{A_2(k_2) \to Y^1(k_1, (k_2, k_3)) \mid \forall k_3 \in A_3\} \\ &= \bigwedge_{k_2 \in K_2} (A_2(k_2) \to Y^{12}_{A_3}(k_1, k_2)),\end{aligned}$$

which is A_2 derivated in $\mathbb{K}^{12}_{A_3}$, i.e., the fist component of the triconcept, namely A_1. Since (A_1, A_2) is a fuzzy concept, it is a maximal rectangle and A_3 is the largest set containing this maximal rectangle.

We still have to check the other pair of derivation operators. Let $X_3 \subseteq K_3$, then the maximality of $X_3^{(3)} = (A_1, A_2)$ is automatically satisfied, as we obtain $X_3^{(3)}$ from the double scaled context. The maximality of $(A_1 \times A_2)^{(3)}$ follows analogously to the first case. □

As a direct consequence of this proposition, we have the following statement:

Proposition 2. *For an f-valued triconcept (A_1, A_2, A_3) it holds that $A_i = (A_j \times A_k)^{(i)}$ for $\{i, j, k\} = \{1, 2, 3\}$ with $j < k$.* □

For the $(-)^{A_k}$-derivation operators we also distinguish between two cases, namely when A_k is a crisp set and when it is fuzzy. When A_k is crisp, i.e., $A_k := A_3$ we proceed as follows: For $X_i \subseteq L^{K_i}$ with $i \in \{1, 2\}$ and $A_3 \subseteq K_3$ we define

$$X_1 \mapsto X_1^{A_3} := \bigwedge_{k_1 \in K_1} (X_1(k_1)^\bullet \to Y^{12}_{A_3}(k_1, k_2)), \tag{8}$$

$$X_2 \mapsto X_2^{A_3} := \bigwedge_{k_2 \in K_2} (X_2(k_2) \to Y^{12}_{A_3}(k_1, k_2)) \tag{9}$$

for the dyadic fuzzy context $\mathbb{K}^{12}_{A_3} := (K_1, K_2, Y^{12}_{A_3})$. where

$$\begin{aligned}&Y^{12}_{A_3} : K_1 \times K_2 \times A_3 \to L, \\ &\qquad Y^{12}_{A_3}(k_1, k_2) := \bigwedge \{tv_{k_3}(k_1, k_2) \mid \forall (k_1, k_2, k_3) \in K_1 \times K_2 \times A_3\}.\end{aligned}$$

These derivation operators are the fuzzy counterparts of the $(-)^{A_k}$-derivation operators, because A_k is crisp. In the discrete case we have $(k_i, k_j) \in Y^{i,j}_{A_k}$ if and only if for all $k_k \in A_k$ it holds that $(k_i, k_j, k_k) \in Y$. Therefore, in the fuzzy setting for $Y^{ij}_{A_3}(k_i, k_j)$, we take the minimum of the values $tv_{k_3}(k_i, k_j)$. Since $\mathbb{K}^{12}_{A_3}$ is a fuzzy context, the $(-)^{A_3}$-derivation operators form fuzzy Galois connections.

In (8) we will need the hedge $^\bullet$ for the computation of the unique stem base, however in general we take the identity for this hedge.

For the $(-)^{A_j}$-derivation operators with $\{i,j\} = \{1,2\}$ the situation is different, because A_j is a fuzzy set. In the following we discuss more possibilities to obtain these derivation operators. In such cases we are interested in the relation between K_i and K_3 for the values of A_j. This means that we are interested in just a part of the double-scaled context $\widetilde{\mathbb{K}}$, namely in $\widetilde{\mathbb{K}}_{A_j} := \bigwedge_{a_j \in A_j} (K_i^+, K_3, a_j, \widetilde{Y})$. So, we could use discrete derivation operators to compute the concepts of $\widetilde{\mathbb{K}}_{A_j}$ and afterwards transform them into fuzzy concepts. However, this is a laborious task and was presented just for a better understanding of the problem.

Another approach for the $(-)^{A_j}$**-derivation operators** is the following:

$$X_i \mapsto X_i^{A_j} := \{k_3 \in K_3 \mid k_i \otimes k_j \leq tv_{k_3}(k_i, k_j), \forall (k_i, k_j) \in X_i \times A_j\},$$
$$X_3 \mapsto X_3^{A_j} := \bigvee \{k_i \in L^{K_i} \mid k_i \otimes k_j \leq tv_{k_3}(k_i, k_j), \forall (k_3, k_j) \in X_3 \times A_j\}.$$

In this case we do not need to double-scale the context. We compute the fuzzy concept induced by X_i and A_j and check under which conditions it exists. This way we obtain $X_i^{A_j}$, i.e., the third component of the f-valued triconcept that is induced by X_i and A_j. To obtain $X_3^{A_j}$ we consider each $k_i \in L^{K_i}$ and check whether the maximal rectangle $k_i \otimes A_j$ exists under the fixed conditions of X_3. Afterwards, we take the maximum of these k_i's due to the maximality property of f-valued triconcepts. This approach is laborious, especially the computation of $X_3^{A_j}$ due to the large number of k_i's we have to check.

We will consider a more straight-forward approach by computing the fuzzy context induced by A_j. A similar approach was presented in [1]. For $X_i \in L^{K_i}$, $A_j \in L^{K_j}$ with $\{i,j\} = \{1,2\}$ and $A_3 \subseteq K_3$ we have

$$X_i \mapsto X_i^{A_j} := \bigwedge_{k_i \in K_i} (X_i(k_i)^\bullet \to Y^{i3}_{A_j}(k_i, k_3))^*, \tag{10}$$
$$X_3 \mapsto X_3^{A_j} := \bigwedge_{k_j \in K_j} (X_j(k_j) \to Y^{i3}_{A_j}(k_i, k_3)), \tag{11}$$

where $^\bullet$ and * are hedge operators. The $^\bullet$ operator is optional, as it is needed just for the computation of the stem base. It is the identity if $i = 1$. The * hedge is always a compulsory globalization in order to assure that $X_i^{A_j}$ yields a crisp set. Then, (10) and (11) are the derivation operators of the fuzzy context $(K_i, K_3, Y^{i3}_{A_j})$ where $Y^{i3}_{A_j}(k_i, k_3) := \bigwedge_{k_j \in K_j} (A_j(k_j) \to Y(k_i, k_j, k_k))$.

Considering in (10) and (11) all values for the indices, i.e., instead of $(-)^{A_j}$ we take $(-)^{A_k}$ for $\{i,j,k\} = \{1,2,3\}$, and ignoring *, these derivation operators suffice for other approaches to Fuzzy Triadic Concept Analysis. This happens due to the fact that such derivation operators yield triconcepts in which all three components are fuzzy sets.

Proposition 3. *For $\{i,j,k\} = \{1,2,3\}$ there are (fuzzy) sets $X_i \in L^{K_i}$ ($X_i \in K_i$, if $i = 3$) and $X_k \in L^{K_k}$ ($X_k \in K_k$, if $k = 3$) such that $A_j := X_i^{X_k}$, $A_i :=$*

$A_j^{X_k}$ *and* $A_k := (A_i \times A_j)^{(k)}$ *(if* $i < j$*) or* $A_k := (A_j \times A_i)^{(k)}$ *(if* $i > j$*). Then,* (A_1, A_2, A_3) *is an f-valued triconcept denoted by* $b_{ik}(X_i, X_k)$ *having the smallest* k*-th component under all f-valued triconcepts* (B_1, B_2, B_3) *with the largest* j*-th component satisfying* $X_i \subseteq B_i$ *and* $X_k \subseteq B_k$*. Particularly,* $b_{ik}(A_i, A_k) = (A_1, A_2, A_3)$ *for each f-valued triconcept* (A_1, A_2, A_3) *of* $\mathbb{K}$*.*

Proof. Without loss of generality we can assume $(i, j, k) = (1, 2, 3)$. Obviously, $X_1 \subseteq A_1$ and $X_3 \subseteq A_3$. We start by proving that (A_1, A_2, A_3) is indeed an f-valued triconcept. From Proposition 1 we have $A_3 = (A_1 \times A_2)$. Then, $A_2 \subseteq A_1^{(A_1 \times A_2)^{(3)}} = A_1^{A_3} \subseteq X_1^{X_3} = A_2$. Hence, $A_2 = A_1^{A_3} = (A_1 \times A_3)^{(2)}$, similarly $A_1 = (A_2 \times A_3)^{(1)}$ and together with Proposition 2 they yield an f-valued triconcept. The rest of the proof is analogous to the crisp case. Let $(B_1, B_2, B_3) \in \mathfrak{T}(\mathbb{K})$ with $X_1 \subseteq B_1$ and $X_3 \subseteq B_3$. Then, $B_2 \subseteq A_2$, because $B_2 = (B_1 \times B_3)^{(2)} = B_1^{B_3} \subseteq X_1^{X_3} = A_2$. If $B_2 = A_2$, by similar consideration as before, we obtain $B_1 \subseteq A_1$. Therefore, we have $A_3 = (A_1 \times A_2)^{(3)} \subseteq (B_1 \times B_2)^{(3)} = B_3$, finishing the first part of the proof. Now, if (A_1, A_2, A_3) is an f-valued triconcept, then $A_1^{A_3} = (A_1 \times A_3)^{(2)} = A_2$ and $A_2^{A_3} = (A_2 \times A_3)^{(1)} = A_1$. Therefore, $b_{ik}(A_1, A_3) = (A_1, A_2, A_3)$ follows by the first part of the proposition. □

4 F-valued Implications

In this section we will study f-valued implications, as generalisations of those elaborated for the discrete case in [4]. There, the authors presented various triadic implications, which are stronger than the ones developed in [15]. For a given discrete triadic context $\mathbb{K} = (K_1, K_2, K_3, Y)$ and for $R, S \subseteq K_2$ and $C \subseteq K_3$ the expression $R \xrightarrow{C} S$ was called *conditional attribute implication*. For $R, S \subseteq K_3$ and $C \subseteq K_2$ the expression $R \xrightarrow{C} S$ was called *attributional condition implication*. Implications of the form $R \to S$ with $R, S \subseteq K_2 \times K_3$ were called *attribute×condition implications*. Our main aim in the upcoming subsections is to generalise such implications to our setting.

4.1 F-valued Conditional Attribute vs. Attributional Condition Implications

In this subsection we study implications of the form: *If we are moderately vigilant during an exam, then we are also fevered* and *If we are serious during an exam, then we feel the same during our presentation.*

Definition 3. *For* $R, S \subseteq L^{K_2}, C \subseteq K_3$ *and globalization* $^\bullet$ *we call the expression* $R \xrightarrow{C} S$ ***f-valued conditional attribute implication*** *and its truth value is given by*

$$\begin{aligned} R \xrightarrow{C} S &:= tv(\forall g \in K_1((\forall m \in R, (g,m) \times C \in Y)^\bullet \to (\forall n \in S, (g,n) \times C \in Y))) \\ &= \bigwedge_{g \in K_1} \Big(\bigwedge_{m \in K_2} (R(m) \to Y_C^{12}(g,m)) \to \bigwedge_{n \in K_2} (S(n) \to Y_C^{12}(g,n)) \Big) \\ &= tv(S \subseteq R^{CC}). \end{aligned}$$

Note that these implications are ordinary fuzzy implications since we are working in the fuzzy context $\mathbb{K}_C^{12}$.

Example 5. For the context given in Figure 2 we have, for example, the f-valued conditional attribute implication $s(0.5) \overset{E}{\to} f(1) = s(0.5) \overset{P}{\to} f(1) = 0.5$ and yet another is $s(0.5) \overset{F}{\to} f(1) = 0$. The first implication means that whenever the students are partially serious during an exam then they are also fevered. The same holds for this implication during a presentation given by the students. The implication does not hold when they are meeting their friends. In such situations the students can be serious but have a relaxed attitude.

For an f-valued triadic context $\mathbb{K}$ we denote by

$$Imp(K_2) := \{R \to S \mid R, S \in L^{K_2}\}$$

the set of all fuzzy implications on K_2. We construct the dyadic context

$$\mathfrak{C}_{imp}(\mathbb{K}) := (Imp(K_2), K_3, I)$$

where $Imp(K_2)$ is a fuzzy set, K_3 is a crisp set and $I(R \to S, c) := R \overset{c}{\to} S$. In order to keep the condition set crisp, we use in $\mathfrak{C}_{imp}(\mathbb{K})$ a slightly different version of the dyadic fuzzy derivation operators defined in (5), namely

$$A^{\shortmid}(m) := \bigwedge_{g \in Imp(K_2)} (A(g)^* \to I(g,m)),\ B^{\shortmid}(g) := (\bigwedge_{m \in K_3} (B(m) \to I(g,m)))^{\bullet}$$

for $A \in Imp(K_2)$, $B \in K_3$ and * is the globalization. Then, $(A, B) \in \underline{\mathfrak{B}}(\mathfrak{C}_{imp}(\mathbb{K}))$ contains in its extent all the implications that hold under all conditions of B. As in the crisp case, each extent is an implicational theory and hence, every extent has a stem base. In the concept lattice of $\mathfrak{C}_{imp}(\mathbb{K})$ the implicational theories are hierarchically ordered by the conditions under which they hold. The extent A is the set of all implications that hold in (K_1, K_2, Y_c^{12}) with $c \in C$.

The number of fuzzy implications can be very large, since we have all implications $A \to B$ with $A, B \subseteq L^{K_2}$. In the crisp case an implication either holds or not, whereas in the fuzzy case an implication holds with a given truth value, i.e., with $tv(A \to B)$. We have $tv(a \to b, c) = \bigwedge\{tv(a \to b), tv(a \to c)\}$ and $tv(a, b \to c) = \bigwedge\{tv(a \to c), tv(b \to c)\}$ for all $a, b, c \in L^{K_2}$. Hence, for the structure of $\mathfrak{C}_{imp}(\mathbb{K})$ it is enough to compute implications of the form $a \to b$ and $a(\mu) \to a(\nu)$ for all $a, b \in L^{K_2}$ with $b \neq a$ and $\mu, \nu \in L$ with $\mu \lneq \nu$. As discussed before, the other implications are infimum reducible elements in the lattice.

In accordance with the idea presented in [4] we label the concept lattice of $\mathfrak{C}_{imp}(\mathbb{K})$ as follows: The attribute labelling is done in the usual way. For the object labelling the situation is more cumbersome. Each set of implications from $Imp(K_2)$ is an extent of $\mathfrak{C}_{imp}(\mathbb{K})$ and an implicational theory, as discussed above. The object labels shall be distributed such that every extent is generated as an implicational theory by the labels attached to it and to its subconcepts. Therefore, the bottom element of the lattice will contain the stem base of all f-valued conditional attribute implications.

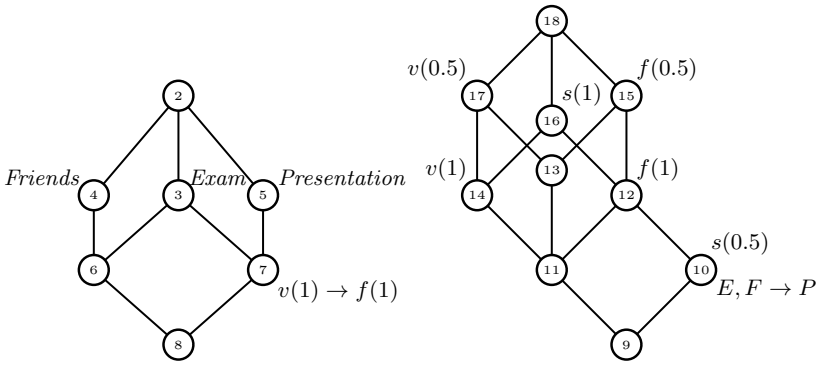


Fig. 3. Conditional attribute vs. attributional condition implications

On the left part in Figure 3 the lattice of $\mathfrak{C}_{imp}(\mathbb{K})$ is displayed. For better legibility we used just the attribute labels (the conditions) and one object label (conditional attribute implication). The implication $v(1) \to f(1)$ from the lattice means that whenever the students are vigilant in degree (truth value) 1 during an exam and presentation they are also fevered in degree 1 in these situations.

An implication $C \to D$ between the intents of $\mathfrak{C}_{imp}(\mathbb{K})$ means that if $R \overset{C}{\to} S$ holds, then $R \overset{D}{\to} S$ must hold as well. For our example the stem base of $\mathfrak{C}_{imp}(\mathbb{K})$ is $P, F \to E$. We could perform a condition attribute exploration as proposed in [4] for the discrete case, however this would go beyond the scope of this paper.

In a triadic context we may arbitrarily interchange the roles of objects, attributes and conditions. Therefore, a triadic context has a sixfold symmetry. By interchanging attributes with conditions in Definition 3, we obtain the *attributional condition implications* defined as follows:

Definition 4. *For $R, S \subseteq K_3$ and $M \subseteq L^{K_2}$ the expression*

$$\begin{aligned} R \overset{M}{\to} S &:= tv(\forall g \in K_1((\forall a \in R, g \times M \times a \in Y^*) \to (\forall b \in S, g \times M \times b \in Y^*))) \\ &= \bigwedge_{g \in K_1} \Big(\bigwedge_{a \in K_3} (R(a) \to Y_M^{13}(g,a))^* \to \bigwedge_{b \in K_3} (S(b) \to Y_M^{13}(g,a))^* \Big), \end{aligned}$$

*is called **f-valued attributional condition implication**, where * is the globalization.*

We use the globalization hedge operator because this time $R \overset{M}{\to} S$ is a crisp implication. For example, for the f-valued triadic context from Table 2 we have the attributional condition implication $P \overset{v(1)}{\to} E, F = 1$, meaning that students who are vigilant during a presentation are also vigilant during an exam and while meeting friends. On the other hand, $P \overset{f(1)}{\to} E, F = 0$ means that a student being fevered during a presentation does not imply that he/she is fevered during an exam and while spending time with friends.

In analogy to the conditional attribute implications, we can also build the context $\mathfrak{C}_{imp}(\widetilde{\mathbb{K}}) := (Imp(K_3), K_2 \times L, I)$ for the attributional condition implications. This time we have $Imp(K_3) := \{R \to S \mid R, S \in K_3\}$, i.e., all implications on K_3 and $I(R \to S, m) := R \overset{m}{\to} S$. The extents of $\mathfrak{C}_{imp}(\widetilde{\mathbb{K}})$ consist of all implications that hold in (K_1, K_3, Y^{13}_m) with $m \in K_2$. The concept lattice is displayed on the right in Figure 3. For example the implication $E, F \to P$ means that if the students during an exam and while meeting friends are (partially) fevered and (partially) serious, then they have the same feelings during their presentation.

The connection between the two classes of implications is an open question even for the discrete case and it remains open for the f-valued triadic case as well.

4.2 F-valued Attribute×Condition Implications

As presented for the discrete case, the two classes of implications studied so far are not powerful enough to express all possible kinds of implications in a triadic context. Therefore, we will generalise the so-called attribute×condition implications to our setting. These express implications of the form *If we are serious during our presentation, then we are moderately fevered during the exam.*

Definition 5. *For $R, S \subseteq L^{K_2} \times K_3$ the expression $R \to S$ is an **f-valued attribute×condition implication** and its truth value is given by*

$$\bigwedge_{g \in K_1} \Big(\bigwedge_{(m,b) \in K_2 \times K_3} (R(m,b) \to Y(g,m,b))^\bullet \to \bigwedge_{(n,c) \in K_2 \times K_3} (S(n,c) \to Y(g,n,c)) \Big),$$

where $^\bullet$ is the globalization, if we want to compute the unique stem base, otherwise the identity.

These are the attribute implications of the fuzzy context $(K_1, K_2 \times K_3, Y^{(1)})$. Their stem base is given by the stem base of the attribute implications from $(K_1, K_2 \times K_3, Y^{(1)})$.

Obviously, such implications can be easily obtained by the f-valued conditional attribute and attributional condition implications, i.e., if we have $R \overset{C}{\to} S$ for $R, S \subseteq L^{K_2}, C \subseteq K_3$, then we can compute $R \times \{c\} \to S \times \{c\}$ for all $c \in C$. Going the other way around, namely transforming the f-valued attribute×condition implications into f-valued conditional attribute and attributional condition implications, is of course also possible.

One could also be interested in f-valued object×attribute or object×condition implications. For our example this would mean *If the first group of students is fevered, then the second one is serious.*

5 Conclusion and Further Research

First, we presented a new framework for treating triadic fuzzy data. For this setting we generalised the notions of the $(-)^{A_k}$ and $(-)^{(i)}$ derivation operators,

triconcepts and trilattices. We also showed how our notions can be translated into different approaches to Fuzzy Triadic Concept Analysis studied by other authors. One of our main results is the generalisation of the $(-)^{(i)}$ derivation operator for the f-valued triadic and fuzzy triadic setting, since it is absent in other works dealing with fuzzy triadic data. Second, we generalised triadic implications to our f-valued setting. These are of major importance for the development of Fuzzy and Fuzzy-Valued Triadic Concept Analysis.

Future research will focus on the connection between the different classes of f-valued triadic implications. As mentioned at the beginning, [5] is an extended version of this paper including the factorization problem. In the future we want to apply the f-valued triadic factorization to real world data.

References

1. Belohlávek, R., Osicka, P.: Triadic concept analysis of data with fuzzy attributes. In Hu, X., Lin, T.Y., Raghavan, V.V., Grzymala-Busse, J.W., Liu, Q., Broder, A.Z., eds.: GrC, IEEE Computer Society (2010) 661–665
2. Osicka, P., Konecny, J.: General approach to triadic concept analysis 116-126. In Kryszkiewicz, M., Obiedkov, S.A., eds.: Proc. CLA 2010, University of Sevilla (2010) 116–126
3. Clara, N.: Hierarchies generated for data represented by fuzzy ternary relations. In: Proceedings of the 13th WSEAS international conference on Systems, Stevens Point, Wisconsin, USA, World Scientific and Engineering Academy and Society (WSEAS) (2009) 121–126
4. Ganter, B., Obiedkov, S.A.: Implications in triadic formal contexts. In: ICCS. (2004) 186–195
5. Glodeanu, C.: Fuzzy-valued triadic concept analysis and its applications. Technical Report MATH-AL-07-2011, Technische Universitat Dresden (September 2011)
6. Ganter, B., Wille, R.: Formale Begriffsanalyse: Mathematische Grundlagen. (1996)
7. Lehmann, F., Wille, R.: A triadic approach to formal concept analysis. In Ellis, G., Levinson, R., Rich, W., Sowa, J.F., eds.: ICCS. Volume 954 of Lecture Notes in Computer Science., Springer (1995) 32–43
8. Wille, R.: The basic theorem of triadic concept analysis. Order **12** (1995) 149–158
9. Pollandt, S.: Fuzzy Begriffe. Springer Verlag, Berlin Heidelberg New York (1997)
10. Belohlávek, R.: Fuzzy Relational Systems: Foundations and Principles. Volume 20 of IFSR Int. Series on Systems Science and Engineering. Kluwer Academic/Plenum Press (2002)
11. Belohlávek, R., Vychodil, V.: Fuzzy concept lattices constrained by hedges. JACIII **11**(6) (2007) 536–545
12. Belohlávek, R., Vychodil, V.: Attribute implications in a fuzzy setting. In: ICFCA. (2006) 45–60
13. Belohlávek, R., Vychodil, V., Chlupová, M.: Implications from data with fuzzy attributes. In: AISTA 2004 in Cooperation with the IEEE Computer Society Proceedings. (2004)
14. Guigues, J.L., Duquenne, V.: Familles minimales d'implications informatives resultant d'un tableau de donnes binaires. Math. Sci. Humaines (95) (1986)
15. Biedermann, K.: A Foundation of the Theory of Trilattices. Shaker Verlag, Aachen (1988)

Mining Biclusters of Similar Values with Triadic Concept Analysis

Mehdi Kaytoue[1], Sergei O. Kuznetsov[2], Juraj Macko[3], Wagner Meira Jr.[1] and Amedeo Napoli[4]

[1] Universidade Fereral de Minas Gerais – Belo Horizonte – Brazil
[2] HSE – Pokrovskiy Bd. 11 – 109028 Moscow – Russia
[3] Palacky University – 17. listopadu – 77146 Olomouc – Czech Republic
[4] INRIA/LORIA – Campus Scientifique, B.P. 239 – Vandœuvre-lès-Nancy – France
kaytoue@dcc.ufmg.br, kuznetsovs@yandex.ru, juraj.macko@upol.cz, meira@dcc.ufmg.br, napoli@loria.fr

Abstract. Biclustering numerical data became a popular data-mining task in the beginning of 2000's, especially for analysing gene expression data. A bicluster reflects a strong association between a subset of objects and a subset of attributes in a numerical object/attribute data-table. So called biclusters of similar values can be thought as maximal sub-tables with close values. Only few methods address a complete, correct and non redundant enumeration of such patterns, which is a well-known intractable problem, while no formal framework exists. In this paper, we introduce important links between biclustering and formal concept analysis. More specifically, we originally show that Triadic Concept Analysis (TCA), provides a nice mathematical framework for biclustering. Interestingly, existing algorithms of TCA, that usually apply on binary data, can be used (directly or with slight modifications) after a preprocessing step for extracting maximal biclusters of similar values.

Keywords: Triadic concept analysis, numerical biclustering, scaling

1 Introduction

Numerical data biclustering mainly appeared in the beginning of 2000's as a first answer to new challenges raised by biological data analysis, and especially gene expression data analysis [13]. Starting from an object/attribute numerical data-table (e.g. Table 1), the goal is to group together some objects with some attributes according to the values taken by these attributes for these objects [13]. Accordingly, a bicluster is formally defined as a pair composed of a set of objects and a set of attributes. Such pair can be represented as a rectangle in the numerical table, modulo lines and columns permutations. Table 1 is a numerical dataset with objects in lines and attributes in columns, while each table entry corresponds to the value taken by the attribute in column for the object in line. Table 2 illustrates bicluster $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\})$ as a grey rectangle.

There are several types of biclusters in the literature (see [13] for a survey), depending on the relation between the values taken by their attributes for their

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

objects. The most simple case can be understood as rectangles of equal values: a bicluster corresponds to a set of objects whose values taken by a same set of attributes are exactly the same, e.g. $(\{g_1, g_2, g_3\}, \{m_5\})$. Constant biclusters only appear in idyllic situations: generally numerical data are noisy. Accordingly, a straightforward generalization of such biclusters lies in so called biclusters of similar values: they are represented by rectangles with almost identical, say similar, values [13, 1, 7]. Table 2 illustrates a bicluster of similar values $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\})$ where two values are said to be similar if their difference is no more than 1. Moreover, this bicluster is maximal: neither an object nor an attribute can be added without violating the similarity condition.

Only few methods address a complete, correct and non redundant enumeration of such patterns [1, 7], which is a well-known intractable problem [13], while no formal framework exists. In this paper, we show that Formal Concept Analysis (FCA) [3], and especially Triadic Concept Analysis (TCA) [12] provides a suitable and well defined framework for this task: Basically, an object has an attribute under a condition (a value). After a simple scaling procedure (turning original data into binary), a bicluster is represented as a triadic concept, composed of a set of objects, a set of attributes (both characterizing the corresponding "rectangle") and a set of values. All sets are maximal thanks to existing concept forming derivation operators of TCA. This comes with several advantages:

- Two values w_1, w_2 of the original data are said to be similar iff their difference does not exceed a given parameter θ. In this case, we write $w_1 \simeq_\theta w_2 \iff |w_1 - w_2| \leq \theta$. Otherwise, we write $w_1 \not\simeq_\theta w_2$. The trilattice produced with TCA after scaling gives all maximal biclusters of similar values for any θ ordered w.r.t. similarity of their values.
- The well known notion of *frequency* takes a semantics w.r.t. similarity of values. For example, let (A, B, C) be a triconcept, where A is a set of objects, B a set of attributes, and C a set of similar values. Assume (A, B) to be the corresponding bicluster. The higher $|C|$, the more similar are the values of the bicluster. If all $|A|$, $|B|$, and $|C|$ are high we obtain a bicluster represented as a large rectangle of close values.
- Existing algorithms from TCA [4] and n-ary closed set mining [2] can be used directly after scaling. We also provide a new algorithm to compute biclusters maximal only for a given θ (see algorithm TriMax later on).
- Both scaling procedure and algorithm TriMax computations can be directly distributed to several computing cores.
- The method can be adapted to n-ary numerical datasets. For example, with $n = 3$, a n-cluster would be a maximal $3D$-box of similar values. It can be applied to 3D gene expression data, monitoring the behaviour of genes in different samples over time. It follows that mining n-dimensional clusters can be achieved with $n + 1$-adic concept analysis.

The paper is organized as follows. Firstly, preliminaries regarding TCA are presented in Section 2. Then Section 3 formally states the problem. It is followed by the description of our two methods, respectively in Section 4 and 5. The

first shows how TCA can help characterizing all maximal biclusters for any θ, while the second restricts the problem to a user-given θ. This is followed by experiments on the proposed approaches. Finally, the paper ends with a discussion and perspectives of further research.

Table 1: A numerical dataset

	m_1	m_2	m_3	m_4	m_5
g_1	1	2	2	1	6
g_2	2	1	1	0	6
g_3	2	2	1	7	6
g_4	8	9	2	6	7

Table 2: A bicluster of similar values

	m_1	m_2	m_3	m_4	m_5
g_1	**1**	**2**	**2**	1	6
g_2	**2**	**1**	**1**	0	6
g_3	**2**	**2**	**1**	7	6
g_4	8	9	2	6	7

2 Triadic Concept Analysis

We assume that the reader is familiar with basic notions of Formal Concept Analysis [3]. Lehmann and Wille introduced Triadic Concept Analysis (TCA [12]). Data are represented by a triadic context, given by (G, M, B, Y). G, M, and B are respectively called sets of objects, attributes and conditions, and $Y \subseteq G \times M \times B$. The fact $(g, m, b) \in Y$ is interpreted as the statement "Object g has the attribute m under condition b".

A (triadic) concept of (G, M, B, Y) is a triple (A_1, A_2, A_3) with $A_1 \subseteq G$, $A_2 \subseteq M$ and $A_3 \subseteq B$ satisfying the two following statements: (i) $A_1 \times A_2 \times A_3 \subseteq Y$, $X_1 \times X_2 \times X_3 \subseteq Y$ and (ii) $A_1 \subseteq X_1$, $A_2 \subseteq X_2$ and $A_3 \subseteq X_3$ implies $A_1 = X_1$, $A_2 = X_2$ and $A_3 = X_3$. If (G, M, B, Y) is represented by a three dimensional table, (i) means that a concept stands for a 3-dimensional rectangle full of crosses while (ii) characterises component-wise maximality of concepts. For a triadic concept (A_1, A_2, A_3), A_1 is called the extent, A_2 the intent and A_3 the modus.

To describe the derivation operators, it is convenient to alternatively represent a triadic context as (K_1, K_2, K_3, Y). Then, for $\{i, j, k\} = \{1, 2, 3\}$, $j < k$, $X \subseteq K_i$ and $Z \subseteq K_j \times K_k$, (i)-derivation operators are defined by:

$$\Phi : X \to X^{(i)} : \{(a_j, a_k) \in K_j \times K_k \mid (a_i, a_j, a_k) \in Y \text{ for all } a_i \in X\}$$
$$\Phi' : Z \to Z^{(i)} : \{a_i \in K_i \mid (a_i, a_j, a_k) \in Y \text{ for all } (a_j, a_k) \in Z\}$$

This definition leads to derivation operator $\mathbf{K}^{(3)}$ and dyadic context $\mathbf{K}^{(3)} = \langle K_3, K_1 \times K_2, Y^{(3)} \rangle$. Further derivation operators are defined as follows: for $\{i, j, k\} = \{1, 2, 3\}$, $X_i \subseteq K_i$, $X_j \subseteq K_j$ and $A_k \subseteq K_k$, the (i, j, A_k)-derivation operators are defined by:

$$\Psi : X_i \to X_i^{(i,j,A_k)} : \{a_j \in K_j \mid (a_i, a_j, a_k) \in Y \text{ for all } (a_i, a_k) \in X_i \times A_k\}$$
$$\Psi' : X_j \to X_j^{(i,j,A_k)} : \{a_i \in K_i \mid (a_i, a_j, a_k) \in Y \text{ for all } (a_j, a_k) \in X_j \times A_k\}$$

Operators Φ and Φ' will be called outer operators, pair of both operators outer closure and dyadic operators Ψ and Ψ' inner operators or inner closure when pair of both is used. Derivation operators of dyadic context are defined by $\mathbf{K}_{A_k}^{ij} = \langle K_i, K_j, Y_{A_k}^{ij} \rangle$, where $(a_i, a_j) \in Y_{A_k}^{ij}$ iff a_i, a_j, a_k are related by Y for all $a_k \in A_k$.

From a computational point of view, [4] developed the algorithm Trias for extracting frequent triadic concepts, i.e. whose extent, intent and modus cardinalities are higher than user-defined thresholds (see also [5]). Cerf et al. presented

a more efficient algorithm called DATA-PEELER able to handle n-ary relations [2] while formal definitions lie in so called Polyadic Concept Analysis [14].

3 Notations and problem settings

A numerical dataset is realized by a many-valued context [3] and we define accordingly (maximal) biclusters of similar values.

Definition 1 (Many-valued context). *Let G be a set of objects, M be a set of attributes, W be the set of attribute values and I be a ternary relation defined on the Cartesian product $G \times M \times W$. The fact $(g, m, w) \in I$, also written $m(g) = w$, means that "Attribute m takes the value w for the object g". The tuple (G, M, W, I) is called many-valued context, or simply numerical dataset in this paper.*

Example 1. Table 1 is a numerical dataset, or many-valued context, with objects $G = \{g_1, g_2, g_3, g_4\}$, attributes $M = \{m_1, m_2, m_3, m_4, m_5\}$, $W = \{0, 1, 2, 6, 7, 8, 9\}$ and for example $m_5(g_2) = 6$.

Definition 2 (Bicluster). *In a numerical dataset (G, M, W, I), a bicluster is a tuple (A, B) with $A \subseteq G$ and $B \subseteq M$.*

Definition 3 (Similarity relation and bicluster of similar values). *Let $w_1, w_2 \in W$ be two attribute values and $\theta \in \mathbb{N}$ be a user-defined parameter, called similarity parameter. w_1 and w_2 are said to be similar iff $|w_1 - w_2| \leq \theta$ and we note $w_1 \simeq_\theta w_2$. (A, B) is bicluster of similar values if $m(g) \simeq_\theta n(h)$ for all $g, h \in A$ and for all $m, n \in B$.*

Definition 4 (Maximal bicluster of similar values). *A bicluster of similar values (A, B) is maximal if adding either an object in A or an attribute in B does not result in a bicluster of similar values.*

Example 2 (From Table 1). $(\{g_1, g_4\}, \{m_2, m_4\})$ is a bicluster. $(\{g_1, g_2\}, \{m_2\})$ is a bicluster of similar values with $\theta \geq 1$. However, it is not maximal. With $1 \leq \theta < 5$, $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\})$ is maximal. Finally, with $\theta = 7$ the bicluster $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3, m_4, m_5\})$ is maximal. Note that a constant (maximal) bicluster is a (maximal) bicluster of similar values with $\theta = 0$.

Thus the problem that we address in this paper is the extraction of all maximal biclusters of similar values from a numerical dataset. We desire the extraction to be complete, correct and non-redundant compared to several existing methods of the literature based on heuristics [13]. For that matter, we propose in the next section a first method aiming at extracting biclusters for any similarity parameter θ. This method establishes new links between biclustering and FCA in general, and TCA in particular. Then, the present methodology is adapted to characterize and extract biclusters that are maximal for a given θ only as usually done in the literature [1, 7, 13].

4 Biclusters of similar values in Triadic Concept Analysis

Firstly, we consider the problem of generating maximal biclusters for any θ. Starting from a numerical dataset (G, M, W, I), the basic idea lies in building a triadic context (G, M, T, Y) where the two first dimensions remain formal objects and formal attributes, while W is scaled into a third dimension denoted by T. This new dimension T is called the *scale dimension*: intuitively, it gives different "spaces of values" that each object-attribute pair $(g, m) \in G \times M$ can take. Once the scale is given, a triadic context is derived from which triadic concepts are characterized.

We use the *interordinal scaling* [3] to build the scale dimension. It allows to encode in 2^T all possible intervals of values in W. This scale allows to derive a triadic context from which any bicluster of similar values can be characterized as a triadic concept. We made more precise these statements and illustrate the whole procedure with examples.

Definition 5 (Interordinal Scaling). *A scale is a binary relation $J \subseteq W \times T$ associating original elements from the set of values W to their derived elements in T. In the case of interordinal scaling, $T = \{[min(W), w], \forall w \in W\} \cup \{[w, max(W)], \forall w \in W\}$. Then $(w, t) \in J$ iff $w \in t$.*

Example 3. Table 3 gives the tabular representation of the interordinal scale for Table 1. Intuitively, each line describes a single value, while dyadic concepts represent all possible intervals over W. An example of dyadic concept in this table is given by $(\{6, 7, 8\}, \{t_6, t_7, t_8, t_9, t_{10}\})$, rewritten as $(\{6, 7, 8\}, \{[6, 8]\})$ since $\{t_6, t_7, t_8, t_9, t_{10}\}$ represents the interval $[0, 8] \cap [0, 9] \cap [1, 9] \cap [2, 9] \cap [6, 9] = [6, 8]$.

J	t_1 = [0, 0]	t_2 = [0, 1]	t_3 = [0, 2]	t_4 = [0, 6]	t_5 = [0, 7]	t_6 = [0, 8]	t_7 = [0, 9]	t_8 = [1, 9]	t_9 = [2, 9]	t_{10} = [6, 9]	t_{11} = [7, 9]	t_{12} = [8, 9]	t_{13} = [9, 9]
0	×	×	×	×	×	×	×						
1		×	×	×	×	×	×	×					
2			×	×	×	×	×	×	×				
6				×	×	×	×	×	×	×			
7					×	×	×	×	×	×	×		
8						×	×	×	×	×	×	×	
9							×	×	×	×	×	×	×

Table 3: Interordinal scale of the set of attribute values W.

Once the scale is defined, we can derive the triadic context w.r.t. this scale.

Definition 6 (Triadic scaled context). *Let Y be ternary relation $Y \subseteq G \times M \times T$. Then $(g, m, t) \in Y$ iff $(m(g), t) \in J$, or simply $m(g) \in t$. We call the tuple (G, M, T, Y) the triadic scaled context of the numerical dataset (G, M, W, I).*

Example 4. The object-attribute pair (g_1, m_1) taking value $m_1(g_1) = 1$ is scaled into triples $(g_1, m_1, t) \in Y$ where t takes any interval in $\{[0, 1], [0, 2], [0, 6], [0, 7],$

$[0,8],[0,9],[1,9]\}$. The intersection of intervals in this set is the original value itself, i.e. $m_1(g_1) = 1$, a basic property of interordinal scaling. As a result, Table 4 illustrates the whole scaled triadic context derived from the numerical dataset given in Table 1 using interordinal scale. The very first cross ($\times$) in this table (upper left) represents the tuple (g_2, m_4, t_1), meaning that $m_4(g_2) \in [0,0]$.

We present now our first main result: there is a one-to-one correspondence between (i) the set of maximal biclusters of similar values in a given numerical dataset for any similarity parameter θ and (ii) the set of all triadic concepts in the triadic context derived with interordinal scaling.

Proposition 1. *Tuple $\langle A, B, U\rangle$, where $A \subseteq G$, $B \subseteq G$ and $U \subseteq T$ is triadic concept iff (A, B) is a maximal bicluster of similar values for some $\theta \geq 0$.*

Proof. We leave the proof in the Appendix of the paper since we need to introduce notations and propositions not necessary in the rest of the paper.

Example 5. For example, $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\}, \{t_3, t_4, t_5, t_6, t_7, t_8\})$ is a triadic concept from the context depicted in Table 4. It corresponds to the maximal bicluster $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\})$ with $\theta = 1$. $\theta = 1$ since $\{t_3, t_4, t_5, t_6, t_7, t_8\}$ is maximal (it is a modus), it corresponds to interval $[1, 2]$ and naturally $2-1 = 1$ is the length of this interval.

	$t_1 = [0,0]$					$t_2 = [0,1]$					$t_3 = [0,2]$					$t_4 = [0,6]$					$t_5 = [0,7]$				
	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5
g_1						×			×		×	×	×	×		×	×	×	×	×	×	×	×	×	×
g_2				×			×	×	×		×	×	×	×		×	×	×	×	×	×	×	×	×	×
g_3								×			×	×	×			×	×	×		×	×	×	×	×	×
g_4													×						×	×				×	×	×

	$t_6 = [0,8]$					$t_7 = [0,9]$					$t_8 = [1,9]$					$t_9 = [2,9]$					$t_{10} = [6,9]$				
	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5
g_1	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×		×	×		×					×
g_2	×	×	×	×	×	×	×	×	×	×	×	×	×		×	×				×					×
g_3	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×				×	×
g_4	×		×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×		×	×

	$t_{11} = [7,9]$					$t_{12} = [8,9]$					$t_{13} = [9,9]$				
	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5
g_1															
g_2															
g_3				×											
g_4	×	×			×	×	×					×			

Table 4: Triadic scaled context for Table 1 with interordinal scaling.

Hence we showed that extracting biclusters of similar values for any θ in a numerical dataset can be achieved by (i) scaling the attribute value dimension and (ii) extracting the triadic concepts in the resulting derived triadic context.

Interestingly, triadic concepts (A, B, U) with the largest sets A, B or C represent large biclusters of close values. Indeed, the larger $|A|$ and $|B|$ the larger the data covering of the corresponding bicluster. Furthermore, the larger $|U|$, the more similar values for bicluster (A, B). Indeed, by the properties of interordinal

scaling, the more intervals in U, the smaller their interval intersection. Mining so called top-k frequent triadic concepts can accordingly be achieved with the existing algorithm DATA-PEELER [2].

On another hand, extracting maximal biclusters for all θ may be neither efficient nor effective with large numerical data: their number tends to be very large and all biclusters are not relevant for a given analysis. Furthermore, both size and density of contexts derived with interordinal scaling are known to be problematic w.r.t algorithmic scalability, see e.g. [9]. In existing methods of the literature, θ is set *a priori*. We show now how to handle this case with slight modifications, our second main result.

5 Extracting biclusters of similar values for a given θ

In this section we consider the problem of extracting maximal biclusters of similar values in TCA for a given θ only. It comes with slight modifications of the methodology presented in last section. Intuitively, consider the previous scaling applied on a numerical dataset (G, M, W, I). It scales W into dimension T and subsets of T characterize all intervals of values over W. To get maximal biclusters for a given θ only, we should not consider all possible intervals in W, but rather all intervals (i) having a range size that is less or equal than θ to avoid biclusters with non similar values, and (ii) having a range size the closest as possible to θ to avoid non-maximal biclusters. For example, if we set $\theta = 2$, it is probably not interesting to consider interval $[0, 8]$ in the scale dimension since $8 - 0 > \theta$. Similarly, considering the interval $[6, 6]$ may not be interesting as well, since a bicluster with all its values equal to 6 may not be maximal. As introduced in [6], those maximal intervals of similar values used for the scale are called blocks of tolerance over the set of numbers W with respect to the tolerance relation $\simeq_\theta$.

Therefore we firstly recall basics on tolerance relations over a set of numbers. It allows us to define a simpler scaling procedure. The resulting triadic context is then mined with a new TCA algorithm called TRIMAX to extract maximal biclusters of similar values for a given θ.

Blocks of tolerance over W are defined as maximal sets of pairwise similar values from W:

Definition 7 (Tolerance blocks from a set of numbers). *The similarity relation $\simeq_\theta$ is called a tolerance relation, i.e. reflexive, symmetric but not transitive. Given a set W of values, a subset $V \subseteq W$, and a tolerance relation $\simeq_\theta$ over W, V is a* block of tolerance *if:*

(i) $\forall w_1, w_2 \in V,\ w_1 \simeq_\theta w_2$ *(pairwise similarity)*
(ii) $\forall w_1 \notin V, \exists w_2 \in V,\ w_1 \not\simeq_\theta w_2$ *(maximality).*

From Table 1 we have $W = \{0, 1, 2, 6, 7, 8, 9\}$. With $\theta = 2$, one has $0 \simeq_2 2$ but $2 \not\simeq_2 6$. Accordingly, one obtains 3 blocks of tolerance, namely the sets $\{0, 1, 2\}$, $\{6, 7, 8\}$ and $\{7, 8, 9\}$. These three sets can be renamed as the convex hull of their elements on $\mathbb{N}$: respectively, $[0, 2]$, $[6, 8]$ and $[7, 9]$: any number lying between the

minimal and the maximal elements (w.r.t. natural number ordering) of a block of tolerance is naturally similar to any other element of the block.

To derive a triadic context from a numerical dataset, we simply use tolerance blocks over W to define the scale dimension.

Definition 8 (TRIMAX scale relation). *The scale relation is a binary relation $J \subseteq W \times C$, where C is the set of blocks of tolerance over W renamed as their convex hulls. Then, $(w, c) \in J$ iff $w \in c$.*

Example 6. From Table 1 we have: $C = \{[0,1],[1,2],[6,7],[7,8],[8,9]\}$ with $\theta = 1$, and $C = \{[0,2],[6,8],[7,9]\}$ with $\theta = 2$.

Then, we can apply the same context derivation as in previous section: scaling is still based on intervals, but this time it uses tolerance blocks.

Definition 9 (TRIMAX triadic scaled context). *Let $Y \subseteq G \times M \times C$ be a ternary relation. Then $(g, m, c) \in Y$ iff $(m(g), c) \in J$, or simply $m(g) \in c$, where J is the scale relation. (G, M, C, Y) is called the* TRIMAX *triadic scaled context.*

Example 7. Table 5 is the TRIMAX triadic scaled concept derived from the numerical dataset lying in Table 1 with $\theta = 1$.

	label 1					label 2					label 3					label 4					label 5				
	[0, 1]					[1, 2]					[6, 7]					[7, 8]					[8, 9]				
	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5	m_1	m_2	m_3	m_4	m_5
g_1	×				×	×	×	×	×						×										
g_2		×	×	×		×	×	×							×										
g_3			×			×	×	×					×	×					×						
g_4								×					×	×		×				×	×	×			

Table 5: Triadic scaled context using tolerance blocks over W and $\theta = 1$.

Definition 10 (Dyadic context associated with a block of tolerance). *Consider a block of tolerance $c \in C$. The dyadic context associated with this block is given by (G, M, Z) where $z \in Z$ denotes all $(g, m) \in G \times M$ such as $m(g) \in c$.*

Example 8. In Table 5, each such dyadic context is labelled by its corresponding block of tolerance.

Now, remark that blocks of tolerance over W are totally ordered: let $[v_1, v_2]$ and $[w_1, w_2]$ be two blocks of tolerance, one has $[v_1, v_2] < [w_1, w_2]$ iff $v_1 < w_1$. Hence, associated dyadic contexts are also totally ordered and we use a corresponding indexing set to label them. In Table 5, contexts for blocks $\langle[0,1],[1,2],[6,7],[7,8],[8,9]\rangle$ are respectively labelled $\langle 1, 2, 3, 4, 5\rangle$.

We now present our second main results: The scaled triadic context supports the extraction of maximal biclusters of similar values for a given θ. In this case however, existing algorithms of TCA cannot be applied directly. For example, in Table 5, the triconcept $(\{g_3\}, \{m_4\}, \{3, 4\})$ corresponds to a bicluster of similar values which is not maximal. Hence we present hereafter a new TCA algorithm for this task, called TRIMAX.

The basic idea of TriMax relies on the following facts. Firstly, since each dyadic context corresponds to a block of tolerance, we do not need to compute intersections of contexts, such as classically done in TCA. Hence each dyadic context is processed separately. Secondly, a dyadic concept of a dyadic context necessarily represents a bicluster of similar values, but we cannot be sure it is maximal (see previous example). Hence, we need to check if a concept is still a concept in other dyadic contexts, corresponding to other classes of tolerance. This is made precise with the following proposition.

Proposition 2. *Let* (A, B, U) *be a triadic concept from* TriMax *triadic scaled context* (G, M, C, Y)*, such that* U *is the outer closure of a singleton* $\{c\} \subseteq C$*. If* $|U| = 1$*,* (A, B) *is a maximal bicluster of similar values. Otherwise,* (A, B) *is a maximal bicluster of similar values iff* $\nexists y \in [min(U); max(U)]$*,* $y < c$ *s.t.* $(A, B) \neq \Psi_y^{'}(\Psi_y((A, B)))$*, where* $\Psi_y^{'}(.)$ *and* $\Psi_y(.)$ *correspond to inner derivation operators associated with* y^{th} *dyadic context.*

Proof. When $|U| = 1$, (A, B) is a dyadic concept only in one dyadic context corresponding to a block of tolerance. By properties of tolerance blocks, (A, B) is a maximal bicluster. If $|U| \neq 1$, (A, B) is a dyadic concept in $|U|$ dyadic contexts. Since the tolerance block set is totally ordered, it directly implies that modus U is an interval $[min(U); max(U)]$. Hence, if $\exists y \in [min(U); max(U)]$ s.t. $(A, B) = \Psi_y^{'}(\Psi_y((A, B)))$ this means that (A, B) is not a maximal bicluster of similar values.

Description of the TriMax **algorithm.** TriMax starts with scaling initial numerical data into several dyadic contexts, each one standing for a block of tolerance over W with given θ. The set of all dyadic contexts forms accordingly a triadic context. Then, each dyadic context is mined with any FCA algorithm (or closed itemset mining algorithm), and all formal concepts are extracted. For a given concept (A, B), we compute outer derivation $\Phi^{'}((A, B))$, i.e. to obtain the set of dyadic contexts labels in which the current dyadic concept holds. If it results in a singleton, this means that (A, B) is a concept for the current block of tolerance only, i.e. it is a maximal bicluster of similar values, and it has been, or will never be, generated twice. Otherwise, (A, B) is a concept in other contexts, and can be generated accordingly several times (as much as the number of contexts in which it holds). Then, we only consider (A, B) if we are sure it is the last time it is computed. Finally, we need to check if current concept represents a maximal bicluster, i.e. there should not exist a context from the modus where (A, B) is not a dyadic concept.

Proposition 3. TriMax *outputs a (i) complete, (ii) correct and (iii) non redundant collection of all maximal biclusters of similar values for a given numerical dataset and similarity parameter* θ*.*

Proof. (i) and (ii) follow directly from Proposition 2. Statement (iii) is ensured by the second *if* condition of the algorithm: a dyadic concept (or equivalently bicluster) is considered iff it has been extracted in the last dyadic context in which it holds.

Algorithm 1: TriMax

input : Numerical dataset (G, M, W, I), tolerance parameter θ
output: Maximal biclusters of similar values

Let $C = \{[a_i, b_i]\}$ be the totally ordered set of all blocks over W for given θ.
Indices i form an indexing set.
forall the $[a_i, b_i] \in C$ **do**
  Build context (G, M, Z_i) such that $(g, m) \in Z_i \Leftrightarrow m(g) \in [a_i, b_i]$
forall the (G, M, Z_i) **do**
  Use any FCA algorithm to extract all its concepts (A, B)
  forall the *dyadic concepts* (A, B) *in the current context* (G, M, Z_i) **do**
    if $|\Phi^{'}((A, B))| = 1$ **then**
      print (A, B)
    else if $max(\Phi^{'}((A, B)) = i$ **then**
      $x \leftarrow min(\Phi^{'}((A, B))$
      if $\nexists y \in [x, i[$ *s.t.* $(A, B) \neq \Psi_y^{'}(\Psi_y((A, B)))$ **then**
        print (A, B)

6 Computer experiments

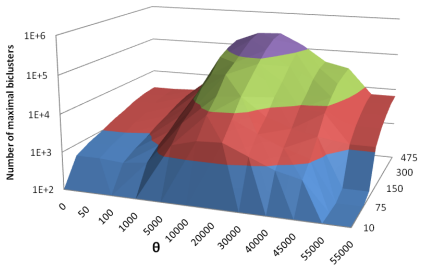
In this section, we experiment with the algorithm TriMax and highlight various aspects of its practical complexity.

Data. We explore a gene expression dataset of the species *Laccaria bicolor* available at NCBI[5]. More details on this dataset can be found in [9]. This gene expression dataset monitors the behaviour of $11,930$ genes in 12 biological situations, reflecting various stages of *Laccaria bicolor* biological cycle. Attribute values in W vary between 0 and $60,000$.

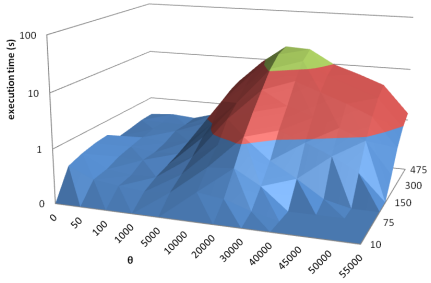
TriMax **implementation**. TriMax is written in C++. It uses the boost library 1.42 for data structures and the implementation of InClose from its authors[6] for dyadic concepts extraction. At each iteration of the main loop, i.e. each tolerance block, the current scaled dyadic context is produced: We do not generated the whole triadic context which cannot fit into memory for large databases. It turns out that the modus computation for a given dyadic concept requires to compute scaling "on the fly", i.e. when computing the set of dyadic contexts in which a current concept holds. The experiments were carried out on an Intel CPU 2.54 Ghz machine with 8 GB RAM running under Ubuntu 11.04.

Experiment settings. The goal of the present experiments is not to give a qualitative evaluation of the present approach (say biological interpretation), but rather a quantitative evaluation. Indeed, the present work aims at showing

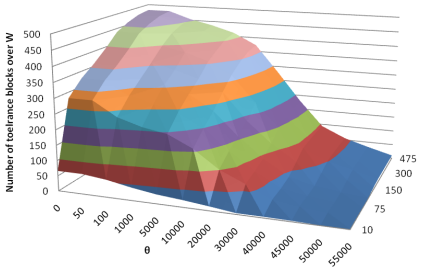
[5] http://www.ncbi.nlm.nih.gov/geo/ as series GSE9784
[6] http://sourceforge.net/projects/inclose/



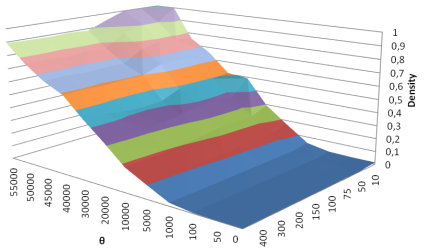
(i) Numbers of patterns (Y-axis) w.r.t. θ (X-axis) and $|G|$ (Z-axis)



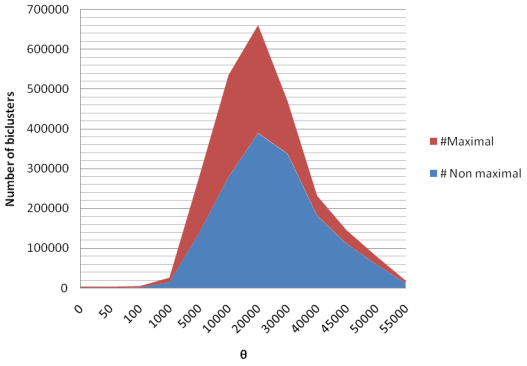
(ii) Execution times in seconds (Y-axis) w.r.t. θ (X-axis) and $|G|$ (Z-axis)



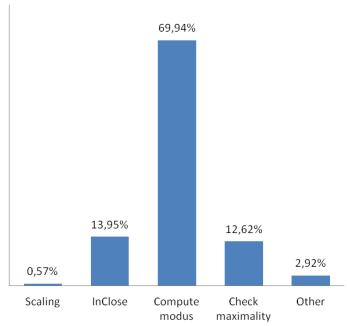
(iii) Numbers of blocks of tolerance (Y-axis) w.r.t. θ (X-axis) and $|G|$ (Z-axis)



(iv) Density of triadic contexts (Y-axis) w.r.t. θ (X-axis) and $|G|$ (Z-axis)



(v) Comparing the number of generated dyadic concepts w.r.t. the actual number of maximal biclusters varying θ with $|G| = 500$



(vi) Repartition of execution time w.r.t main steps of TriMax with $\theta = 33,000$ and $|G| = 500$

Fig. 1: Monitoring with different settings (i) the number of maximal biclusters, (ii) the execution times of TriMax, (iii) the number of tolerance blocks, (iv) the derived triadic context density, (v) the number of non-maximal biclusters generated as dyadic-concepts w.r.t. the number of maximal biclusters, and (vi) repartition of execution time in the TriMax algorithm.

how an existing type of biclusters can be mined with Triadic Concept Analysis. For a qualitative evaluation, the reader may refer for example to [1, 9].

Accordingly, we designed the following experiments to monitor various aspects of the TriMax algorithm. For most of the experiments, the dataset used is composed of an increasing number of objects and all attributes. The objects are chosen randomly once and for all so that the different experiment results can be compared. We also vary the parameter θ in the same way across all experiments. Then, we monitor the following aspects, as presented in Figure 1:

i. Number of maximal biclusters of similar values
ii. Execution time (in seconds)
iii. Number of tolerance blocks
iv. Density of the triadic context, where density is defined as $d(G, M, C, Y) = |Y|/(|G| \times |M| \times |C|)$. This information is important, since contexts with high density are known to be hard to process with FCA algorithms [11], and we use the InClose algorithm for dyadic contexts processing.
v. Comparison between the number of non-maximal biclusters produced by TriMax (i.e. dyadic concepts that do not corresponds to maximal biclusters) with the number of maximal biclusters.
vi. Execution time profiling of the main procedures of TriMax. This is achieved with the tool GNU GProf and gives us what parts of the algorithm are the most time consuming.

Experiment results. Figure 1 presents the results of our experiments with different settings. In these settings, we vary the number of objects $|G|$ and the parameter θ. A first observation arises from graph (i): the number of biclusters is the highest when $\theta \simeq 30,000$. A first explanation is that $30,000$ is the half of the maximal value of W and almost all multiples of 100 in $[0; 60,000]$ belongs to W. In graph (ii), execution time has the same behaviour as graph (i). These results can be understood by paying attention to the next graphs (iii) and (iv). In (iii) is monitored the number of tolerance blocks. The maximal number is reached when $\theta = 0$, i.e. $|C| = |W|$. When $\theta = max(W)$, we have $|C| = 1$. Now we observe in (iv) that the density follows a reverse behaviour: When $\theta = 0$, the density tends towards 0%; when $\theta = max(W)$, then density exactly equal 1%. Combining both graph (iii) and (iv), the worst cases happen when both density and tolerance bloc count are high.

Another observation, which explains also the execution times, arises from graph (v). Here are compared the number of maximal biclusters and the number of non-maximal biclusters generated as dyadic concepts. Here again, worst case is reached when $\theta \simeq 30,000$. Looking at graph (vi), we learn that this is however not the major problem. The mostly consuming procedure of TriMax is the computation of the modus of a dyadic concept. The explanation is that we compute modus with "on the fly scaling".

Therefore, the bottleneck of our algorithm reveals itself to be the modus computation. In practical applications however, the analyst is not interested in all biclusters of similar values. Some constraints are generally defined, such as a minimal (resp. maximal) number of objects (resp. attributes) in a bicluster

(A, B), or a minimal area $|A| \times |B|$, etc. Interestingly, most of those constraints can be evaluated on a generated dyadic concept. Therefore, before computing the modus of such concept, we can check such properties and discard the concept if not respecting the constraints. Although not reflected in this paper, we tested how adding minimal (resp. maximal) size constraints on a bicluster affects both number of biclusters and execution times. The results are very interesting: for example with $\theta = 33,000$, $|G| = 500$, and minimal (resp. maximal) size for $|A|$ set to 10 (resp. 40), TriMax produces only $5,332$ maximal biclusters in 2.1 seconds compared to $104,226$ maximal biclusters extracted in 16.130 seconds without any constraint.

Finally, the most interesting aspect of TriMax is its direct distributed computation capacity. Indeed, each iteration, i.e. for each block of tolerance, can be achieved independently from the others. Furthermore, the core of TriMax consisting in extracting dyadic contexts can also be distributed, see e.g. [10]. A deeper investigation remains to be done in this case. Note that although the method description involves W as a set of natural numbers, TriMax can directly handle numerical data real numbers, and has been implemented as such.

Comparison with existing methods. Two existing methods in the literature also consider the problem of extracting all maximal biclusters of similar values from a numerical dataset. The first method is called *Numerical Biset Miner* (NBS-Miner [1]). The second method is based on *interval pattern structures* (IPS [7, 8]). Limited by space, we do not detail these methods. Both NBS-Miner and IPS algorithms have been implemented in C++. First experiments show that NBS-Miner is not scalable compared to IPS and TriMax. On another hand, it seems that TriMax outperforms IPS, but a deeper investigation is required. The main problem in IPS is to find an efficient algorithm able to compute tolerance blocks over a set of intervals.

7 Conclusion

We addressed the problem of biclustering numerical data with Formal Concept Analysis. So called (maximal) biclusters of similar values can be characterized and extracted with Triadic Concept Analysis, which turns out to be a novel mathematical framework for this task. We properly defined a scaling procedure turning original numerical data into triadic contexts from which biclusters can be extracted as triadic concepts with existing algorithms. This approach allows a correct, complete and non-redundant extraction of all maximal biclusters, for any similarity parameter θ and can be extended to n-ary numerical datasets while their computation can be directly distributed. The interpretation of triadic concepts is very rich: both extent and intent allow to characterize a bicluster (i.e. the rectangle), while the modus gives the range of values of the biclusters, and for which θ is the bicluster maximal. Moreover, the larger the modus, the more similar the values within current bicluster. It follows a perspective of research, aiming at extracting the top-k frequent tri-concepts with Data-Peeler [2], which can help to handle the problem of top-k biclusters extraction. We also adapted the

TCA machinery with algorithm TriMax to extract maximal biclusters for a user-defined θ, which is classical in the existing literature. It appears that TriMax is a fully customizable algorithm: any concept extraction algorithm can be used inside its core (along with several constraints on produced dyadic concepts), while its distributed computation is direct. Among several other experiments, it remains now to determine which are the best core algorithms for a given θ parameter, the very last directly influencing derived contexts density.

Acknowledgements. Authors would like to thank Dmitry Andreevich Morozov for implementing the algorithms NBS-Miner and IPS. The second author was supported by the project of the Russian Foundation for Basic Research, grant no. 08-07-92497-NTsNIL_a. Juraj Macko acknowledges support by Grant No. 202/10/0262 of the Czech Science Foundation.

A Proof of the Proposition 1.

Before proving this proposition, we need to introduce the following. For sake of simplicity, we now consider W as the set of all natural numbers from a numerical dataset that are greater or equal than the minimal value and lower or equal than the maximal value, i.e. $W = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ with the example of Table 1.

Definition 11 (Scale value and scale relation). *We call scale value* $s = q - r$ *where* $r = min(W)$ *and* $q = max(W)$. *The scale relation is a binary relation* $J \subseteq W \times T$, *where* $T = \{t_1, \ldots, t_{2s+1}\}$ $r \leq w \leq q$ *and* $\langle w, t_i \rangle \in J$ *iff* $i \in [w - r + 1, w - r + 1 + s]$.

Note that J is equivalent to interordinal scale of W previously given, but this notations are used for the proof.

Definition 12 ($E_{\theta w}$ - cluster base). *We introduce* $E_{\theta w} \subseteq T$ *defined as* $E_{\theta w} = [t_{w+\theta-r+1}; t_{w-r+1+s}]$ *for given* θ *and* $w \in W$.

Example 9 ($E_{\theta w}$ - cluster base). $E_{12} = [t_{2+1-0+1}; t_{2-0+1+9}] = [t_4; t_{12}]$.

Proposition 4. $(w_b = m(g)) \simeq_\theta (n(h) = w_c)$ *iff* $(\langle g, m \rangle \in Y^{12}_{E_{\theta b}}$ *and* $\langle h, n \rangle \in Y^{12}_{E_{\theta b}})$.

Proof. Let $E_b, E_c \subseteq T$ and $w_c \geq w_b$. According to the definition $(g, m) \in Y^{12}_{E_{\theta b}}$ iff m, g, t are related by Y for all $t \in E_{\theta b}$. Using scaling and definition we have $[t_{w_b-r+1}; t_{w_b-r+1+s}] = E_b \supseteq E_{\theta b} = [t_{w_b+\theta-r+1}; t_{w_b-r+1+s}]$ which is straightforward. We just need to show that $(h, n) \in Y^{12}_{E_{\theta b}}$ holds as well. With scaling definition and previous definition we get $[t_{w_c-r+1}; t_{w_c-r+1+s}] = E_c \supseteq E_{\theta b} = [t_{w_b+\theta-r+1}; t_{w_b-r+1+s}]$ holding iff $w_c - w_b \leq \theta$, which is equal to the definition of $\simeq_\theta$.

Moreover we can easily see as a corollary that $w_c - w_b \leq \theta$ holds iff $E_b \cap E_c \supseteq E_{\theta b}$ and $w_c - w_b = \theta$ holds iff $E_b \cap E_c = E_{\theta b}$. Now we can prove the Proposition 1 from the main text.

Proposition 1. *Tuple* $\langle A_1, A_2, U\rangle$*, where* $A_1 \subseteq G$*,* $A_2 \subseteq M$ *and* $U \subseteq T$ *is triadic concept iff* (A_1, A_2) *is a maximal bicluster of similar values for some* $\theta \geq 0$*. Furthermore the value of* θ *is defined as* $\theta = s - |U| + 1$*.*

Proof. Let $U = E_{\theta b}$ and consider dyadic context $Y_U^{12} = Y_{E_{\theta b}}^{12}$ for some w_b. Using dyadic closure operator $\Psi^{'}(\Psi((A_1))$ we get (A_1, A_2). From definition of triconcept we know that $A_1 \subseteq B_1$ implies $A_1 = B_1$ (the same for A_2). From definition of maximal bicluster of similar values we know that $\langle A_1, A_2\rangle$ is maximal when it does not exists $\langle B_1, B_2\rangle$ s.t. $B_1 \supseteq A_1$ (the same applies for A_2). It is obvious that both sets are maximal from definition and when we have the same dyadic context $Y_U^{12} = Y_{E_{\theta b}}^{12}$. Now we need to look at dyadic context $Y_U^{12} = Y_{E_{\theta b}}^{12}$. In $|U| = |E_{\theta b}| = |[t_{w_b+\theta-r+1}; t_{w_b-r+1+s}]|$ we can easily see that $|U| = s - \theta + 1$, which gives $\theta = s - |U| + 1$.

Finally, U is maximal (as being modus of a triconcept) and $E_{\theta b}$ is maximal as well because $w_c - w_b \leq \theta$ holds iff $E_b \cap E_c \supseteq E_{\theta b}$. All facts mentioned in this proof leads to equality of the triconcept and maximal bicluster of similar values.

References

1. Besson, J., Robardet, C., Raedt, L.D., Boulicaut, J.F.: Mining bi-sets in numerical data. In: Dzeroski, S., Struyf, J. (eds.) KDID. Lecture Notes in Computer Science, vol. 4747, pp. 11–23. Springer (2007)
2. Cerf, L., Besson, J., Robardet, C., Boulicaut, J.F.: Closed patterns meet n-ary relations. TKDD 3(1) (2009)
3. Ganter, B., Wille, R.: Formal Concept Analysis. Springer (1999)
4. Jäschke, R., Hotho, A., Schmitz, C., Ganter, B., Stumme, G.: Trias - an algorithm for mining iceberg tri-lattices. In: ICDM. pp. 907–911 (2006)
5. Ji, L., Tan, K.L., Tung, A.K.H.: Mining frequent closed cubes in 3d datasets. In: Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB). pp. 811–822. ACM (2006)
6. Kaytoue, M., Assaghir, Z., Napoli, A., Kuznetsov, S.O.: Embedding tolerance relations in formal concept analysis: an application in information fusion. In: CIKM. pp. 1689–1692. ACM (2010)
7. Kaytoue, M., Kuznetsov, S.O., Napoli, A.: Biclustering numerical data in formal concept analysis. In: Valtchev, P., Jäschke, R. (eds.) ICFCA. LNCS, vol. 6628, pp. 135–150. Springer (2011)
8. Kaytoue, M., Kuznetsov, S.O., Napoli, A.: Revisiting numerical pattern mining with formal concept analysis. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI). IJCAI/AAAI (2011)
9. Kaytoue, M., Kuznetsov, S.O., Napoli, A., Duplessis, S.: Mining gene expression data with pattern structures in formal concept analysis. Inf. Sci. 181(10), 1989–2001 (2011)
10. Krajca, P., Vychodil, V.: Distributed algorithm for computing formal concepts using map-reduce framework. In: IDA. pp. 333–344. Springer (2009)
11. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. J. Exp. Theor. Artif. Intell. 14(2-3), 189–216 (2002)
12. Lehmann, F., Wille, R.: A triadic approach to formal concept analysis. In: ICCS. LNCS, vol. 954, pp. 32–43. Springer (1995)

13. Madeira, S., Oliveira, A.: Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Transactions on Computational Biology and Bioinformatics 1(1), 24–45 (2004)
14. Voutsadakis, G.: Polyadic concept analysis. Order 19(3), 295–304 (2002)

Fast Mining of Iceberg Lattices: A Modular Approach Using Generators

Laszlo Szathmary[1], Petko Valtchev[1], Amedeo Napoli[2], Robert Godin[1], Alix Boc[1], and Vladimir Makarenkov[1]

[1] Dépt. d'Informatique UQAM, C.P. 8888,
Succ. Centre-Ville, Montréal H3C 3P8, Canada
Szathmary.L@gmail.com, {valtchev.petko, godin.robert}@uqam.ca,
{makarenkov.vladimir, boc.alix}@uqam.ca
[2] LORIA UMR 7503, B.P. 239, 54506 Vandœuvre-lès-Nancy Cedex, France
napoli@loria.fr

Abstract. Beside its central place in FCA, the task of constructing the concept lattice, i.e., concepts plus Hasse diagram, has attracted some interest within the data mining (DM) field, primarily to support the mining of association rule bases. Yet most FCA algorithms do not pass the scalability test fundamental in DM. We are interested in the iceberg part of the lattice, alias the frequent closed itemsets (FCIs) plus precedence, augmented with the respective generators (FGs) as these provide the starting point for nearly all known bases. Here, we investigate a modular approach that follows a workflow of individual tasks that diverges from what is currently practiced. A straightforward instantiation thereof, *Snow-Touch*, is presented that combines past contributions of ours, *Touch* for FCIs/FGs and *Snow* for precedence. A performance comparison of *Snow-Touch* to its closest competitor, *Charm-L*, indicates that in the specific case of dense data, the modularity overhead is offset by the speed gain of the new task order. To demonstrate our method's usefulness, we report first results of a genome data analysis application.

1 Introduction

Association discovery [1] in data mining (DM) is aimed at pinpointing the most frequent patterns of *items*, or *itemsets*, and the strongest *associations* between items dug in a large *transaction* database. The main challenge here is the potentially huge size of the output. A typical way out is to focus on a *basis*, i.e. a reduced yet lossless representation of the target family (see a list in [2]). Many popular bases are either formulated in terms of FCA or involve structures that do. For instance, the *minimal non-redundant association rules* [3] require the computation of the *frequent closed itemsets* (FCI) and their respective *frequent generators* (FGs), while the *informative basis* involves the inclusion-induced precedence links between FCIs.

We investigate the computation of iceberg lattices, i.e., FCIs plus precedence, together with the FGs. In the DM literature, several methods exist that

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

target FCIs by first discovering the associated FGs (e.g. the levelwise FCI miners *A-Close* [4] and *Titanic* [5]). More recently, a number of extensions of the popular FCI miner *Charm* [6] have been published that output two or all three of the above components. The basic one, *Charm-L* [7], produces FCIs with precedence links (and could be regarded as a lattice construction procedure). Further extensions to *Charm-L* produce the FGs as well (see [8,9]).

In the FCA field, the frequency aspect of concepts has been mostly ignored whereas generators have rarely been explicitly targeted. Historically, the first method whose output combines closures, generators and precedence has been presented in [10] yet this fact is covered by a different terminology and a somewhat incomplete result (see explanations below). The earliest method to explicitly target all three components is to be found in [11] while an improvement was published in [12]. Yet all these FCA-centered methods have a common drawback: They scale poorly on large datasets due to repeated scans of the entire database (either for closure computations or as an incremental restructuring technique). In contrast, *Charm-L* exploits a *vertical* encoding of the database that helps mitigate the cost of the impact of the large object (a.k.a. transaction) set.

Despite a diverging *modus operandi*, both FCA and data mining methods follow the same overall algorithmic schema: they first compute the set of concepts/FCIs and the precedence links between them and then use these as input in generator/FG calculation.

However efficient *Charm-L* is, its design is far from optimal: For instance, FCI precedence is computed at FCI discovery, benefiting from no particular insight. Thus, many FCIs from distant parts of the search space are compared. We therefore felt that there is space for improvement, e.g., by bringing in techniques operating locally. An appealing track seemed to lay in the exploration of an important duality from hypergraph theory to inverse the computation dependencies between individual tasks (and thus define a new overall workflow). To clarify this point, we chose to investigate a less intertwined algorithmic schema, i.e. by a modular design so that each individual task could be targeted by the best among a pool of alternative methods.

Here, we describe a first step in our study, *Snow-Touch*, which has been assembled from existing methods by wiring them w.r.t. our new schema. Indeed, our method relies on *Charm* for mining FCIs and on the vertical FG miner *Talky-G*, which are put together into a combined miner, *Touch* [13], by means of an FGs-to-FCIs matching mechanism. The *Snow* method [14] extracts the precedence links from FCIs and FGs.

The pleasant surprise with *Snow-Touch* was that, when a Java implementation thereof was experimentally compared to *Charm-L* (authors' version in C++) on a wide range of data, our method prevailed on all dense datasets. This was not readily anticipated as the modular design brought a computational overhead, e.g. the extra matching step. Moreover, *Snow-Touch* proved to work well with real-world data, as the first steps of a large-scale analysis of genomic data indicate.

In summary, we contribute here a novel computation schema for iceberg lattices with generators (hence a **new lattice construction approach**). Moreover, we derive an efficient FCI/FG/precedence miner (especially on dense sets). We also demonstrate the practical usefulness of *Snow-Touch* as well as of the global approach for association mining based on generic rules.

The remainder of the paper is as follows: Background on pattern mining, hypergraphs, and vertical pattern mining is provided in Section 2. In Section 3 we present the different modules of the *Snow-Touch* algorithm. Experimental evaluations are provided in Section 4 and conclusions are drawn in Section 5.

2 Background

In the following, we summarize knowledge about relevant structures from frequent pattern mining and hypergraph theory (with parallels drawn with similar notions from FCA) as well as about efficient methods for mining them.

2.1 Basic facts from pattern mining and concept analysis

In pattern mining, the input is a database (comparable to an FCA formal context). Records in the database are called *transactions* (alias objects), denoted here $\mathcal{O} = \{o_1, o_2, \ldots, o_m\}$. A transaction is basically subsets of a given total set of *items* (alias *attributes*), denoted here $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$. Except for its *itemset*, a transaction is explicitly identified through a unique identifier, a *tid* (a set of identifiers is thus called a *tidset*). Throughout this paper, we shall use the following database as a running example (the **"dataset $\mathcal{D}$"**): $\mathcal{D} = \{(1,\ ACDE)$, $(2,\ ABCDE)$, $(3,\ AB)$, $(4,\ D)$, $(5,\ B)\}$.

The standard $'$ derivation operators from FCA are denoted differently in this context. Thus, given an itemset X, the tidset of all transactions comprising X in their itemsets is the *image* of X, denoted $t(X)$ (e.g. $t(AB) = 23$). We recall that an itemset of length k is called a k-itemset. Moreover, the (absolute) *support* of an itemset X, $supp : \wp(\mathcal{A}) \to \mathbb{N}$, is $supp(X) = |t(X)|$. An itemset X is called *frequent*, if its support is not less than a user-provided *minimum support* (denoted by *min_supp*). Recall as well that, in $[X]$, the equivalence class of X induced by $t()$, the extremal elements w.r.t. set-theoretic inclusion are, respectively, the unique maximum X'' (a.k.a. *closed* itemset or the *concept intent*), and its set of minimums, a.k.a. the *generator* itemsets. In data mining, an alternative definition is traditionally used stating that an itemset X is *closed* (*a generator*) if it has no proper superset (subset) with the same support. For instance, in our dataset $\mathcal{D}$, B and C are generators, whose respective closures are B and $ACDE$.

In [6], a subsumption relation is defined as well: X *subsumes* Z, iff $X \supset Z$ and $supp(X) = supp(Z)$. Obviously, if Z *subsumes* X, then Z cannot be a generator. In other words, if X is a generator, then all its subsets Y are generators as well[3]. Formally speaking, the generator family forms a downset within the Boolean lattice of all itemsets $\langle \wp(\mathcal{A}), \subseteq \rangle$.

[3] Please notice that the dual property holds for non generators.

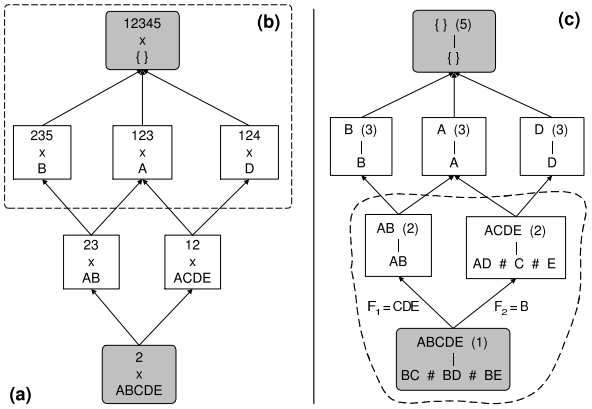


Fig. 1. Concept lattices of dataset $\mathcal{D}$. **(a)** The entire concept lattice. **(b)** An iceberg part of (a) with $min\_supp = 3$ (indicated by a dashed rectangle). **(c)** The concept lattice with generators drawn within their respective nodes

The FCI and FG families losslessly represent the family of all frequent itemsets (FIs) [15]. They jointly compose various non-redundant bases of valid association rules, e.g. the *generic basis* [2]. Further bases require the inclusion order $\leq$ between FCIs or its transitive reduction $\prec$, i.e. the precedence relation.

In Fig. 1 (adapted from [14]), views (a) and (b) depict the concept lattice of dataset $\mathcal{D}$ and its iceberg part, respectively. Here we investigate the efficient computation of the three components of an association rule basis, or what could be spelled as the generator-decorated iceberg (see Fig. 1 (c)).

2.2 Effective mining methods for FCIs, FGs, and precedence links

Historically, the first algorithm computing all closures with their generators and precedence links can be found in [10] (although under a different name in a somewhat incomplete manner). Yet the individual tasks have been addressed separately or in various combinations by a large variety of methods.

First, the construction of all concepts is a classical FCA task and a large number of algorithms exist for it using a wide range of computing strategies. Yet they scale poorly as FCI miners due to their reliance on object-wise computations (e.g. the incremental acquisition of objects as in [10]). These involve to a large number of what is called *data scans* in data mining that are known to seriously deteriorate the performances. In fact, the overwhelming majority of FCA algorithms would suffer on the same drawback as they have been designed under the assumption that the number of objects and the number of attributes remain in the same order of magnitude. Yet in data mining, there is usually a much larger number of transactions than there are items.

As to generators, they have attracted significantly less attraction in FCA as a standalone structure. Precedence links, in turn, are sometimes computed by

concept mining FCA algorithms beside the concept set. Here again, objects are heavily involved in the computation hence the poor scaling capacity of these methods. The only notable exception to this rule is the method described in [16] which was designed to deliberately avoid referring to objects by relying exclusively on concept intents.

When all three structures are considered together, after [10], efficient methods for the combined task have been proposed, among others, in [11,12].

In data mining, mining FCIs is also a popular task [17]. Many FCI miners exist and a good proportion thereof would output FGs as a byproduct. For instance, levelwise miners such as *Titanic* [5] and *A-Close* [17], use FGs as entry points into the equivalence classes of their respective FCIs. In this field, the FGs, under the name of free-sets [15], have been targeted by dedicated miners. Precedence links do not seem to play a major role in pattern mining since few miners would consider them. In fact, to the best of our knowledge, the only mainstream FCI miner that would also output the Hasse diagram of the iceberg lattice is *Charm-L* [8]. In order to avoid multiple data scans, *Charm-L* relies on a specific encoding of the transaction database, called *vertical*, that takes advantage of the aforementioned asymmetry between the number of transactions and the number of items. Moreover, two ulterior extensions thereof [7,9] would also cover the FGs for each FCI, making them the primary competitors for our own approach.

Despite the clear discrepancies in their modus operandi, both FCA-centered algorithms and FCI/FG miners share their overall algorithmic schema. Indeed, they first compute the set of concepts/FCIs and the precedence links between them and then use these as input in generator/FG calculation. The latter task can be either performed along a levelwise traversal of the equivalence class of a given closure, as in [8] and [10], or shaped as the computation of the minimal transversals of a dedicated hypergraph[4], as in [11,12] and [9].

While such a schema could appear more intuitive from an FCA point of view (first comes the lattice, then the generators which are seen as an "extra"), it is less natural and eventually less profitable for data mining. Indeed, while a good number of association rule bases would require the precedence links in order to be constructed, FGs are used in a much larger set of such bases and may even constitute a target structure of their own (see above). Hence, a more versatile mining method would only output the precedence relation (and compute it) as an option, which is not possible with the current design schema. More precisely, the less rigid order between the steps of the combined task would be: (1) FCIs, (2) FGs, and (3) precedence. This basically means that precedence needs to be computed at the end, independently from FG and FCI computations (but may rely on these structures as input). Moreover, the separation of the three steps insures a higher degree of modularity in the design of the concrete methods following our schema: Any combination of methods that solve an individual task could be used, leaving the user with a vast choice. On the reverse side of the coin,

[4] Termed alternatively as *(minimal) blockers* or *hitting sets*.

total modularity comes with a price: if FGs and FCIs are computed separately, an extra step will be necessary to match an FCI to its FGs.

We describe hereafter a method fitting the above schema which relies exclusively on existing algorithmic techniques. These are combined into a single global procedure, called *Snow-Touch* in the following manner: The FCI computation is delegated to the *Charm* algorithm which is also the basis for *Charm-L*. FGs are extracted by our own vertical miner *Talky-G*. The two methods together with an FG-to-FCI matching technique form the *Touch* algorithm [13]. Finally, precedence is retrieved from FCIs with FGs by the *Snow* algorithm [14] using a ground duality result from hypergraph theory.

In the remainder of this section we summarize the theoretical and the algorithmic background of the above methods which are themselves briefly presented and illustrated in the next section.

2.3 Hypergraphs, transversals, and precedence in closure semi-lattices

The generator computation in [11] exploits the tight interdependence between the intent of a concept, its generators and the intents of its immediate predecessor concepts. Technically speaking, a generator is a *minimal blocker* for the family of *faces* associated to the concept intent and its predecessor intents[5].

Example. Consider the closed itemset (CI) lattice in Figure 1 (c). The CI $ABCDE$ has two faces: $F_1 = ABCDE \setminus AB = CDE$ and $F_2 = ABCDE \setminus ACDE = B$.

It turns out that *blocker* is a different term for the widely known *hypergraph transversal* notion. We recall that a hypergraph [18] is a generalization of a graph where edges can connect arbitrary number of vertices. Formally, it is a pair $(V,\mathcal{E})$ made of a basic vocabulary $V = \{v_1, v_2, \ldots, v_n\}$, the *vertices*, and a family of sets $\mathcal{E}$, the *hyper-edges*, all drawn from V.

A set $T \subseteq V$ is called a *transversal* of $\mathcal{H}$ if it has a non-empty intersection with all the edges of $\mathcal{H}$. A special case are the minimal transversals that are exploited in [11].

Example. In the above example, the minimal transversals of $\{CDE, B\}$ are $\{BC, BD, BE\}$, hence these are the generators of $ABCDE$ (see Figure 1 (c)).

The family of all minimal transversals of $\mathcal{H}$ constitutes the *transversal hypergraph* of $\mathcal{H}$ ($Tr(\mathcal{H})$). A duality exists between a simple hypergraph and its transversal hypergraph [18]: For a simple hypergraph $\mathcal{H}$, $Tr(Tr(\mathcal{H})) = \mathcal{H}$. Thus, the faces of a concept intent are exactly the minimal transversals of the hypergraph composed by its generators.

Example. The bottom node in Figure 1 (c) labelled $ABCDE$ has three generators: BC, BD, and BE while the transversals of the corresponding hypergraph are $\{CDE, B\}$.

[5] A face is the set-theoretic difference between the intents of two concepts bound by a precedence link.

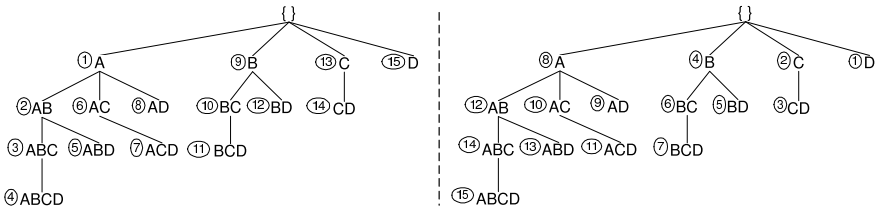


Fig. 2. Left: pre-order traversal with *Eclat*; **Right:** reverse pre-order traversal with *Talky-G*

2.4 Vertical Itemset Mining

Miners from the literature, whether for plain FIs or FCIs, can be roughly split into breadth-first and depth-first ones. Breadth-first algorithms, more specifically the *Apriori*-like [1] ones, apply levelwise traversal of the pattern space exploiting the anti-monotony of the frequent status. Depth-first algorithms, e.g., *Closet* [19], in contrast, organize the search space into a prefix-tree (see Figure 2) thus factoring out the effort to process common prefixes of itemsets. Among them, the *vertical* miners use an encoding of the dataset as a set of pairs (item, tidset), i.e., $\{(i, t(i))|i \in \mathcal{A}\}$, which helps avoid the costly database re-scans.

Eclat [20] is a plain FI miner relying on a vertical encoding at a depth-first traversal of a tree structure, called IT-tree, whose nodes are $X \times t(X)$ pairs. *Eclat* traverses the IT-tree in a pre-order way, from left-to-right [20] (see Figure 2). *Charm* adapts the computing schema of *Eclat* to the rapid construction of the FCIs [6]. It is knowingly one of the fastest FCI-miners, hence its adoption as a component in *Touch* as well as the idea to look for similar technique for FGs. However, a vertical FG miner would be closer to *Eclat* than to *Charm* as it requires no specific speed-up during the traversal (recall that FGs form a downset). In contrast, there is a necessary shift in the test focus w.r.t. *Eclat*: Instead of supersets, subsets need to be examined to check candidate FGs. This, in turn, requires that all such subsets are already tested at the moment an itemset is examined. In other terms, the IT-tree traversal order needs to be a linear extension of $\subseteq$ order between itemsets.

3 The Snow-Touch Algorithm

We sketch below the key components of *Snow-Touch* i.e. *Talky-G*, *Touch*, and *Snow*.

3.1 Talky-G

Talky-G is a vertical FG miner that constructs an IT-tree in a depth-first right-to-left manner [13].

Traversal Of The Generator Search Space

Traversing $\wp(\mathcal{A})$ so that a given set X is processed after all its subsets induces a $\subseteq$-complying traversal order, i.e. a linear extension of $\subseteq$.

In FCA, a similar technique is used by the *Next-Closure* algorithm [21]. The underlying *lectic* order is rooted in an implicit mapping of $\wp(\mathcal{A})$ to $[0 \dots 2^{|\mathcal{A}|} - 1]$, where a set image is the decimal value of its characteristic vector w.r.t. an arbitrary ordering $rank : \mathcal{A} \leftrightarrow [1..|\mathcal{A}|]$. The sets are then listed in the increasing order of their mapping values which represents a depth-first traversal of $\wp(\mathcal{A})$. This encoding yields a depth-first *right-to-left* traversal (called *reverse pre-order traversal* in [22]) of the IT-tree representing $\wp(\mathcal{A})$.

EXAMPLE. See Figure 2 for a comparison between the traversal strategies in *Eclat* (left) and in *Talky-G* (right). Order-induced ranks of itemsets are drawn next to their IT-tree nodes.

The Algorithm

The algorithm works the following way. The IT-tree is initialized by creating the root node and hanging a node for each frequent item below the root (with its respective tidset). Next, nodes below the root are examined, starting from the right-most one. A 1-itemset p in such a node is an FG iff $supp(p) < 100\%$ in which case it is saved to a dedicated list. A recursive exploration of the subtree below the current node then ensues. At the end, all FGs are comprised in the IT-tree.

During the recursive exploration, all FGs from the subtree rooted in a node are mined. First, FGs are generated by "joining" the subtree's root to each of its sibling nodes laying to the right. A node is created for each of them and hung below the subtree's root. The resulting node's itemset is the union of its parents' itemsets while its tidset is the intersection of the tidsets of its parents. Then, all the direct children of the subtree's root are processed recursively in a right-to-left order.

When two FGs are joined to form a candidate node, two cases can occur. Either we obtain a new FG, or a valid FG cannot be generated from the two FGs. A candidate FG is the union of the input node itemsets while its tidset is the intersection of the respective tidsets. It can fail the FG test either by insufficient support (non frequent) or by a strict FG-subset of the same support (which means that the candidate is a proper superset of an already found FG with the same support).

EXAMPLE. Figure 3 illustrates *Talky-G* on an input made of the dataset $\mathcal{D}$ and a $min\_supp = 1$ (20%). The node ranks in the traversal-induced order are again indicated. The IT-tree construction starts with the root node and its children nodes: Since no universal item exists in $\mathcal{D}$, all items are FGs and get a node below the root. In the recursive extension step, the node E is examined first: Absent right siblings, it is skipped. Node D is next: the candidate itemset DE

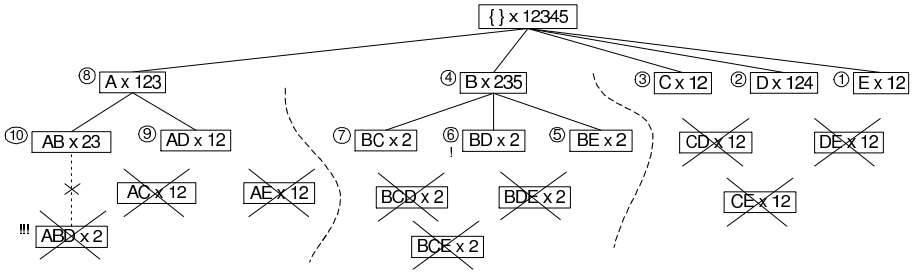


Fig. 3. Execution of *Talky-G* on dataset $\mathcal{D}$ with $min\_supp = 1$ (20%)

fails the FG test since of the same support as E. With C, both candidates CD and CE are discarded for the same reason. In a similar fashion, the only FGs in the subtree below the node of B are BC, BD, and BE. In the case of A, these are AB and AD. ABD fails the FG test because of BD.

Fast Subsumption Checking

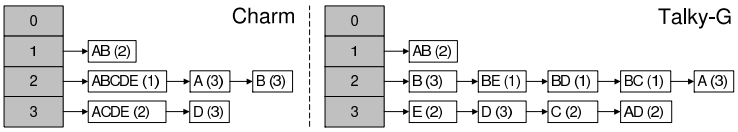
During the generation of candidate FGs, a subsumer itemset cannot be a generator. To speed up the subsumption computation, *Talky-G* adapts the hash structure of *Charm* for storing frequent generators together with their support values. Thus, as tidsets are used for hashing of FGs, two equivalent itemsets get the same hash value. Hence, when tracking a potential subsumee for a candidate X, we check within the corresponding list in the hash table for FGs Y having (i) the same support as X and, if positive outcome, (ii) proper subsets of X (see details in [13]).

Example. The hash structure of the IT-tree in Figure 3 is drawn in Figure 4 (top right). The hash table has four entries that are lists of itemsets. The hash function over a tidset is the modulo 4 of the sum of all tids. For instance, to check whether ABD subsumes a known FG, we take its hash key, $2 \bmod 4 = 2$, and check the content of the list at index 2. In the list order, B is discarded for support mismatch, while BE fails the subset test. In contrast, BD succeeds both the support and the inclusion tests so it invalidates the candidate ABD.

3.2 The Touch Algorithm

The *Touch* algorithm has three main features, namely **(1)** extracting frequent closed itemsets, **(2)** extracting frequent generators, and **(3)** associating frequent generators to their closures, i.e. identifying frequent equivalence classes.

Finally, our method matches FGs to their respective FCIs. To that end, it exploits the shared storage technique in both *Talky-G* and *Charm*, i.e. the hashing on their images (see Figure 4 (top)). The calculation is immediate: as the hash value of a FG is the same as for its FCI, one only needs to traverse



FCI (supp)	FGs
AB (2)	AB
$ABCDE$ (1)	BE; BD; BC
A (3)	A

FCI (supp)	FGs
B (3)	B
$ACDE$ (2)	E; C; AD
D (3)	D

Fig. 4. Top: hash tables for dataset $\mathcal{D}$ with $min\_supp = 1$. **Top left:** hash table of *Charm* containing all FCIs. **Top right:** hash table of *Talky-G* containing all FGs. **Bottom:** output of *Touch* on dataset $\mathcal{D}$ with $min\_supp = 1$

the FG hash and for each itemset lookup the list of FCI associated to its own hash value. Moreover, setting both lists to the same size, further simplifies the procedure as both lists will then be located at the same offset within their respective hash tables.

Example. Figure 4 (top) depicts the hash structures of *Charm* and *Talky-G*. Assume we want to determine the generators of $ACDE$ which is stored at position 3 in the hash structure of *Charm*. Its generators are also stored at position 3 in the hash structure of *Talky-G*. The list comprises three members that are subsets of $ACDE$ with the same support: E, C, and AD. Hence, these are the generators of $ACDE$. The output of *Touch* is shown in Figure 4 (bottom).

3.3 The Snow Algorithm

Snow computes precedence links on FCIs from associated FGs [14]. *Snow* exploits the duality between hypergraphs made of the generators of an FCI and of its faces, respectively to compute the latter as the transversals of the former. Thus, its input is made of FCIs and their associated FGs. Several algorithms can be used to produce this input, e.g. *Titanic* [5], *A-Close* [4], *Zart* [23], *Touch*, etc. Figure 4 (bottom) depicts a sample input of *Snow*.

On such data, *Snow* first computes the faces of a CI as the minimal transversals of its generator hypergraph. Next, each difference of the CI X with a face yields a predecessor of X in the closed itemset lattice.

Example. Consider again $ABCDE$ with its generator family $\{BC, BD, BE\}$. First, we compute its transversal hypergraph: $Tr(\{BC, BD, BE\}) = \{CDE, B\}$. The two faces $F_1 = CDE$ and $F_2 = B$ indicate that there are two predecessors for $ABCDE$, say Z_1 and Z_2, where $Z_1 = ABCDE \setminus CDE = AB$, and $Z_2 = ABCDE \setminus B = ACDE$. Application of this procedure for all CIs yields the entire precedence relation for the CI lattice. □

Table 1. Database characteristics

database name	# records	# non-empty attributes	# attributes (in average)	largest attribute
T25I10D10K	10,000	929	25	1,000
MUSHROOMS	8,416	119	23	128
CHESS	3,196	75	37	75
CONNECT	67,557	129	43	129

4 Experimental Evaluation

In this section we discuss practical aspects of our method. First, in order to demonstrate that our approach is computationally efficient, we compare its performances on a wide range of datasets to those of *Charm-L*. Then, we present an application of *Snow-Touch* to the analysis of genomic data together with an excerpt of the most remarkable gene associations that our method helped to uncover.

4.1 Snow-Touch vs. Charm-L

We evaluated *Snow-Touch* against *Charm-L* [8,9]. The experiments were carried out on a bi-processor Intel Quad Core Xeon 2.33 GHz machine running Ubuntu GNU/Linux with 4 GB RAM. All times reported are real, wall clock times, as obtained from the Unix *time* command between input and output. *Snow-Touch* was implemented entirely in Java. For performance comparisons, the authors' original C++ source of *Charm-L* was used. *Charm-L* and *Snow-Touch* were executed with these options: `./charm-l -i input -s min_supp -x -L -o COUT -M 0 -n`; `./leco.sh input min_supp -order -alg:dtouch -method:snow -nof2`. In each case, the standard output was redirected to a file. The diffset optimization technique [24] was activated in both algorithms.[6]

Benchmark datasets. For the experiments, we used several real and synthetic dataset benchmarks (see Table 1). The synthetic dataset T25, using the IBM Almaden generator, is constructed according to the properties of market basket data. The MUSHROOMS database describes mushrooms characteristics. The CHESS and CONNECT datasets are derived from their respective game steps. The latter three datasets can be found in the UC Irvine Machine Learning Database Repository. Typically, real datasets are very dense, while synthetic data are usually sparse. Response times of the two algorithms on these datasets are presented in Figure 5.

Charm-L. *Charm-L* represents a state-of-the-art algorithm for closed itemset lattice construction [8]. *Charm-L* extends *Charm* to directly compute the lattice while it generates the CIs. In the experiments, we executed *Charm-L* with a switch to compute (minimal) generators too using the *minhitset* method. In [9], Zaki and Ramakrishnan present an efficient method for calculating the generators, which is actually the generator-computing method of Pfaltz and Taylor [25].

[6] *Charm-L* uses diffsets by default, thus no explicit parameter was required.

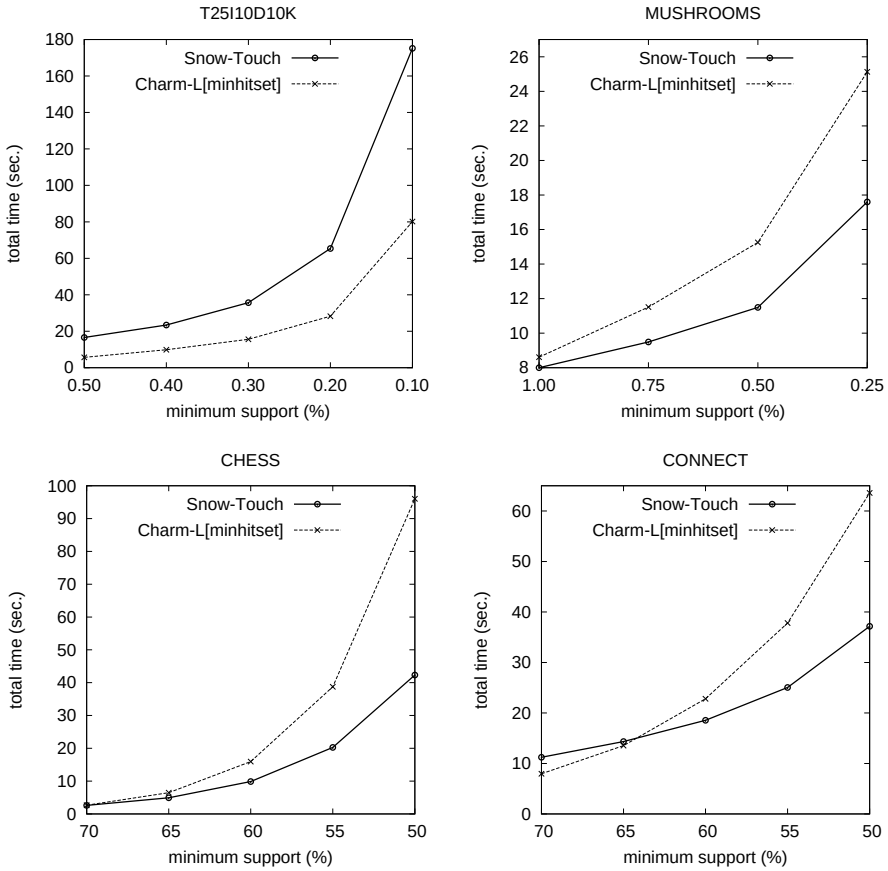


Fig. 5. Response times of *Snow-Touch* and *Charm-L*.

This way, the two algorithms (*Snow-Touch* and *Charm-L*) are *comparable* since they produce *exactly the same output*.

Performance on sparse datasets. On T25, *Charm-L* performs better than *Snow-Touch*. We have to admit that sparse datasets are a bit problematic for our algorithm. The reason is that T25 produces long sparse bitvectors, which gives some overhead to *Snow-Touch*. In our implementation, we use bitvectors to store tidsets. However, as can be seen in the next paragraph, our algorithm outperforms *Charm-L* on all the dense datasets that were used during our tests.

Performance on dense datasets. On MUSHROOMS, CHESS and CONNECT, we can observe that *Charm-L* performs well only for high values of support. Below a certain threshold, *Snow-Touch* gives lower response times, and the gap widens as the support is lowered. When the minimum support is set low enough, *Snow-Touch* can be several times faster than *Charm-L*. Considering that *Snow-Touch* is implemented in Java, we believe that a good C++ implementation could be several orders of magnitude faster than *Charm-L*.

According to our experiments, *Snow-Touch* can construct the concept lattices faster than *Charm-L* in the case of dense datasets. From this, we draw the hypothesis that our direction towards the construction of FG-decorated concept lattices is more beneficial than the direction of *Charm-L*. That is, it is better to extract first the FCI/FG-pairs and then determine the order relation between them than first extracting the set of FCIs, constructing the order between them, and then determining the corresponding FGs for each FCI.

4.2 Analysis of Antibiotic Resistant Genes

We looked at the practical performance of *Snow-Touch* on real-world genomic dataset whereby the goal was to discover meaningful associations between genes in entire genomes seen as items and transactions, respectively.

The genomic dataset was collected from the website of the National Center for Biotechnology Information (NCBI) with a focus on genes from microbial genomes. At the time of writing (June 2011), 1,518 complete microbial genomes were available on the NCBI website.[7] For each genome, its list of genes was collected (for instance the genome with ID CP002059.1 has two genes, *rnpB* and *ssrA*). Only 1,250 genomes out of the 1,518 proved non empty; we put them in a binary matrix of 1,250 rows $\times$ 125,139 columns. With an average of 684 genes per genome we got 0.55% density (i.e., large yet sparse dataset with an imbalance between numbers of rows and of columns).

The initial result of the mining task was the family of *minimal non-redundant association rules* ($\mathcal{MNR}$), which are directly available from the output of *Snow-Touch*. We sorted them according to the confidence. Among all strong associations, the bioinformaticians involved in this study found most appealing the rules describing the behavior of antibiotic resistant genes, in particular, the *mecA* gene. *mecA* is frequently found in bacterial cells. It induces a resistance to antibiotics such as *Methicillin*, *Penicillin*, *Erythromycin*, etc. [26]. The most commonly known carrier of the gene *mecA* is the bacterium known as MRSA (methicillin-resistant *Staphylococcus aureus*).

At a second step, we were narrowing the focus on a group of three genes, *mecA* plus *ampC* and *vanA* [27]. *ampC* is a beta-lactam-resistance gene. AmpC beta-lactamases are typically encoded on the chromosome of many gram-negative bacteria; it may also occur on *Escherichia coli*. *AmpC* type beta-lactamases may also be carried on plasmids [26]. Finally, the gene *vanA* is a vancomycin-resistance gene typically encoded on the chromosome of gram-positive bacteria such as *Enterococcus*. The idea was to relate the presence of these three genes to the presence or absence of any other gene or a combination thereof.

Table 2 shows an extract of the most interesting rules found by our algorithm. These rules were selected from a set of 18,786 rules.

For instance, rule (1) in Table 2 says that the gene *mecA* is present in 85.71% of cases when the set of genes {*clpX*, *dnaA*, *dnaI*, *dnaK*, *gyrB*, *hrcA*, *pyrF*}

[7] http://www.ncbi.nlm.nih.gov/genomes/lproks.cgi

Table 2. An extract of the generated minimal non-redundant association rules. After each rule, the following measures are indicated: support, confidence, support of the left-hand side (antecedent), support of the right-hand side (consequent)

(1) {*clpX*, *dnaA*, *dnaI*, *dnaK*, *gyrB*, *hrcA*, *pyrF*} → {*mecA*} (supp=96 [7.68%]; **conf=0.857 [85.71%]**; suppL=112 [8.96%]; suppR=101 [8.08%])
(2) {*clpX*, *dnaA*, *dnaI*, *dnaK*, *nusG*} → {*mecA*} (supp=96 [7.68%]; **conf=0.835 [83.48%]**; suppL=115 [9.20%]; suppR=101 [8.08%])
(3) {*clpX*, *dnaA*, *dnaI*, *dnaJ*, *dnaK*} → {*mecA*} (supp=96 [7.68%]; **conf=0.828 [82.76%]**; suppL=116 [9.28%]; suppR=101 [8.08%])
(4) {*clpX*, *dnaA*, *dnaI*, *dnaK*, *ftsZ*} → {*mecA*} (supp=96 [7.68%]; **conf=0.828 [82.76%]**; suppL=116 [9.28%]; suppR=101 [8.08%])
(5) {*clpX*, *dnaA*, *dnaI*, *dnaK*} → {*mecA*} (supp=97 [7.76%]; **conf=0.815 [81.51%]**; suppL=119 [9.52%]; suppR=101 [8.08%])
(6) {*greA*, *murC*, *pheS*, *rnhB*, *ruvA*} → {*ampC*} (supp=99 [7.92%]; **conf=0.227 [22.71%]**; suppL=436 [34.88%]; suppR=105 [8.40%])
(7) {*murC*, *pheS*, *pyrB*, *rnhB*, *ruvA*} → {*ampC*} (supp=99 [7.92%]; **conf=0.221 [22.15%]**; suppL=447 [35.76%]; suppR=105 [8.40%])
(8) {*dxs*, *hemA*} → {*vanA*} (supp=29 [2.32%]; **conf=0.081 [8.15%]**; suppL=356 [28.48%]; suppR=30 [2.40%])
(9) {*dxs*} → {*vanA*} (supp=30 [2.40%]; **conf=0.067 [6.73%]**; suppL=446 [35.68%]; suppR=30 [2.40%])

is present in a genome. The above rules have a direct practical use. In one such scenario, they could be used to suggest which antibiotic should be taken by a patient depending on the presence or absence of *certain* genes in the infecting microbe.

5 Conclusion

We presented a new design schema for the task of mining the iceberg lattice and the corresponding generators out of a large context. The target structure directly involved in the construction of a number of association rule bases and hence is of a certain importance in the data mining field. While previously published algorithms follow the same schema, i.e., construction of the iceberg lattice (FCIs plus precedence links) followed by the extraction of the FGs, our approach consists in inferring precedence links from the previously mined FCIs with their FGs.

We presented an initial and straightforward instanciation of the new algorithmic schema that reuses existing methods for the three steps: the popular *Charm* FCI miner, our own method for FG extraction, *Talky-G* (plus an FGs-to-FCIs matching procedure), and the Hasse diagram constructor *Snow*. The resulting iceberg plus FGs miner, *Snow-Touch*, is far from an optimal algorithm, in particular due to redundancies in the first two steps. Yet an implementation thereof within the Coron platform (in Java) has managed to outperform its natural

competitor, *Charm-L* (in C++) on a wide range of datasets, especially on dense ones.

To level the playing ground, we are currently re-implementing *Snow-Touch* in C++ and expect the new version to be even more efficient. In a different vein, we have tested the capacity of our approach to support practical mining task by applying it to the analysis of genomic data. While a large number of associations usually come out of such datasets, many of the redundant with respect to each other, by limiting the output to only the generic ones, our method helped focus the analysts' attention to a smaller number of significant rules.

As a next step, we are studying a more integrated approach for FCI/FG construction that requires no extra matching step. This should result in substantial efficiency gains. On the methodological side, our study underlines the duality between generators and order w.r.t. FCIs: either can be used in combination with FCIs to yield the other one. It rises the natural question of whether FCIs alone, which are output by a range of frequent pattern miners, could be used to efficiently retrieve first precedence, and then FGs.

References

1. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules in Large Databases. In: Proc. of the 20th Intl. Conf. on Very Large Data Bases (VLDB '94), San Francisco, CA, Morgan Kaufmann (1994) 487–499
2. Kryszkiewicz, M.: Concise Representations of Association Rules. In: Proc. of the ESF Exploratory Workshop on Pattern Detection and Discovery. (2002) 92–109
3. Bastide, Y., Taouil, R., Pasquier, N., Stumme, G., Lakhal, L.: Mining Minimal Non-Redundant Association Rules Using Frequent Closed Itemsets. In: Proc. of the Computational Logic (CL '00). Volume 1861 of LNAI., Springer (2000) 972–986
4. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering Frequent Closed Itemsets for Association Rules. In: Proc. of the 7th Intl. Conf. on Database Theory (ICDT '99), Jerusalem, Israel (1999) 398–416
5. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., Lakhal, L.: Computing Iceberg Concept Lattices with Titanic. Data and Knowl. Eng. **42**(2) (2002) 189–222
6. Zaki, M.J., Hsiao, C.J.: CHARM: An Efficient Algorithm for Closed Itemset Mining. In: SIAM Intl. Conf. on Data Mining (SDM' 02). (Apr 2002) 33–43
7. Zaki, M.J.: Mining Non-Redundant Association Rules. Data Mining and Knowledge Discovery **9**(3) (2004) 223–248
8. Zaki, M.J., Hsiao, C.J.: Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure. IEEE Trans. on Knowl. and Data Eng. **17**(4) (2005) 462–478
9. Zaki, M.J., Ramakrishnan, N.: Reasoning about Sets using Redescription Mining. In: Proc. of the 11th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining (KDD '05), Chicago, IL, USA (2005) 364–373
10. Godin, R., Missaoui, R.: An incremental concept formation approach for learning from databases. Theoretical Computer Science Journal (133) (1994) 387–419
11. Pfaltz, J.L.: Incremental Transformation of Lattices: A Key to Effective Knowledge Discovery. In: Proc. of the First Intl. Conf. on Graph Transformation (ICGT '02), Barcelona, Spain (Oct 2002) 351–362
12. Le Floc'h, A., Fisette, C., Missaoui, R., Valtchev, P., Godin, R.: JEN : un algorithme efficace de construction de générateurs pour l'identification des règles d'association. Nouvelles Technologies de l'Information **1**(1) (2003) 135–146

13. Szathmary, L., Valtchev, P., Napoli, A., Godin, R.: Efficient Vertical Mining of Frequent Closures and Generators. In: Proc. of the 8th Intl. Symposium on Intelligent Data Analysis (IDA '09). Volume 5772 of LNCS., Lyon, France, Springer (2009) 393–404
14. Szathmary, L., Valtchev, P., Napoli, A., Godin, R.: Constructing Iceberg Lattices from Frequent Closures Using Generators. In: Discovery Science. Volume 5255 of LNAI., Budapest, Hungary, Springer (2008) 136–147
15. Calders, T., Rigotti, C., Boulicaut, J.F.: A Survey on Condensed Representations for Frequent Sets. In Boulicaut, J.F., Raedt, L.D., Mannila, H., eds.: Constraint-Based Mining and Inductive Databases. Volume 3848 of Lecture Notes in Computer Science., Springer (2004) 64–80
16. Baixeries, J., Szathmary, L., Valtchev, P., Godin, R.: Yet a Faster Algorithm for Building the Hasse Diagram of a Galois Lattice. In: Proc. of the 7th Intl. Conf. on Formal Concept Analysis (ICFCA '09). Volume 5548 of LNAI., Darmstadt, Germany, Springer (May 2009) 162–177
17. Pasquier, N.: Mining association rules using formal concept analysis. In: Proc. of the 8th Intl. Conf. on Conceptual Structures (ICCS '00), Shaker-Verlag (Aug 2000) 259–264
18. Berge, C.: Hypergraphs: Combinatorics of Finite Sets. North Holland, Amsterdam (1989)
19. Pei, J., Han, J., Mao, R.: CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets. In: ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery. (2000) 21–30
20. Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: New Algorithms for Fast Discovery of Association Rules. In: Proc. of the 3rd Intl. Conf. on Knowledge Discovery in Databases. (August 1997) 283–286
21. Ganter, B., Wille, R.: Formal concept analysis: mathematical foundations. Springer, Berlin/Heidelberg (1999)
22. Calders, T., Goethals, B.: Depth-first non-derivable itemset mining. In: Proc. of the SIAM Intl. Conf. on Data Mining (SDM '05), Newport Beach, USA. (Apr 2005)
23. Szathmary, L., Napoli, A., Kuznetsov, S.O.: ZART: A Multifunctional Itemset Mining Algorithm. In: Proc. of the 5th Intl. Conf. on Concept Lattices and Their Applications (CLA '07), Montpellier, France (Oct 2007) 26–37
24. Zaki, M.J., Gouda, K.: Fast vertical mining using diffsets. In: Proc. of the 9th ACM SIGKDD Intl. Conf. on Knowledge discovery and data mining (KDD '03), New York, NY, USA, ACM Press (2003) 326–335
25. Pfaltz, J.L., Taylor, C.M.: Scientific Knowledge Discovery through Iterative Transformation of Concept Lattices. In: Proc. of the SIAM Workshop on Data Mining and Discrete Mathematics, Arlington, VA, USA (2002) 65–74
26. Philippon, A., Arlet, G., Jacoby, G.A.: Plasmid-Determined AmpC-Type β-Lactamases. Antimicrobial Agents and Chemotherapy **46**(1) (2002) 1–11
27. Schwartz, T., Kohnen, W., Jansen, B., Obst, U.: Detection of antibiotic-resistant bacteria and their resistance genes in wastewater, surface water, and drinking water biofilms. Microbiology Ecology **43**(3) (2003) 325–335

Boolean factors as a means of clustering of interestingness measures of association rules *

Radim Belohlavek[1], Dhouha Grissa[2,4,5], Sylvie Guillaume[3,4], Engelbert Mephu Nguifo[2,4], Jan Outrata[1]

[1] Data Analysis and Modeling Lab
Department of Computer Science , Palacky University, Olomouc
17. listopadu 12, CZ-77146 Olomouc, Czech Republic
radim.belohlavek@acm.org,jan.outrata@upol.cz
[2] Clermont Université, Université Blaise Pascal, LIMOS, BP 10448, F-63000 Clermont-Ferrand, France
[3] Clermont Université, Université d'Auvergne, LIMOS, BP 10448, F-63000 Clermont-Ferrand, France
[4] CNRS, UMR 6158, LIMOS, F-63173 Aubiére, France
[5] URPAH, Département d'Informatique, Faculté des Sciences de Tunis, Campus Universitaire, 1060 Tunis, Tunisie
dgrissa@isima.fr,guillaum@isima.fr,mephu@isima.fr

Abstract. Measures of interestingness play a crucial role in association rule mining. An important methodological problem is to provide a reasonable classification of the measures. Several papers appeared on this topic. In this paper, we explore Boolean factor analysis, which uses formal concepts corresponding to classes of measures as factors, for the purpose of classification and compare the results to the previous approaches.

1 Introduction

An important problem in extracting association rules, well known since the early stage of association rule mining [32], is the possibly huge number of rules extracted from data. A general way of dealing with this problem is to define the concept of rule interestingness: only association rules that are considered interesting according to some measure are presented to the user. The most widely used measures of interestingness are based on the concept of support and confidence. However, the suitability of these measures to extract interesting rules was challenged by several studies, see e.g. [34]. Consequently, several other interestingness measures of association rules were proposed, see e.g. [35], [23], [12], [38]. With the many existing measures of interestingness arises the problem of selecting an appropriate one.

* We acknowledge support by the ESF project No. CZ.1.07/2.3.00/20.0059, the project is co-financed by the European Social Fund and the state budget of the Czech Republic (R. Belohlavek); Grant No. 202/10/P360 of the Czech Science Foundation (J. Outrata); and by Grant No. 11G1417 of the French-Tunisian cooperation PHC Utique (D. Grissa).

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

To understand better the behavior of various measures, several studies of the properties of measures of interestingness appeared, see e.g. [12], [27], [23], [16]. Those studies explore various properties of the measures that are considered important. For example, Vaillant et al. [37] evaluated twenty interestingness measures according to eight properties. To facilitate the choice of the user-adapted interestingness measure, the authors applied the clustering methods on the decision matrix and obtained five clusters. Tan et al. [35] studied twenty-one interestingness measures through eight properties and showed that no measure is adapted to all cases. To select the best interestingness measure, they use both a support-based pruning and standardization methods. By applying a new clustering approach, Huynh et al. [21] classifyed thirty-four interestingness measures with a correlation analysis. Geng and Hamilton [12] made a survey of thirty-eight interestingness measures for rules and summaries with eleven properties and gived strategies to select the appropriate measures. D. R. Feno [10] evaluated fifteen interestingness measures with thirteen properties to describe their behaviour. Delgato et al. [9] provided a new study of the interestingness measures by means of the logical model. In addition, the authors proposed and justified the addition of two new principles to the three proposed by Piatetsky-Shapiro [32]. Finally, Heravi and Zaiane [22] studied fifty-three objective measures for associative classification rules according to sixteen properties and explained that no single measure can be introduced as an obvious winner.

The assessment of measures according to their properties results in a measure-property binary matrix. Two studies of this matrix were conducted. Namely, [17] describes how FCA can highlight interestingness measures with similar behavior in order to help the user during his choice. [16] and [14] attempted to find natural clusters of measures using widely used clustering methods, the agglomerative hierarchical method (AHC) and the K-means method. A common feature of these methods is that they only produce disjoint clusters of measures. On the other hand, one could naturally expect overlapping clusters. The aim of this paper is to explore the possibility of obtaining overlapping clusters of measures using factor analysis of binary data and to compare the results with the results of other studies. In particular, we use the recently developed method from [3] and take the discovered factors for clusters. The method uses formal concepts as factors that makes it possible to interpret the factors easily.

2 Preliminaries

2.1 Binary (Boolean) data

Let X be a set of objects (such as a set of customers, a set of functions or the like) and Y be a set of attributes (such as a set of products that customers may buy, a set of properties of functions). The information about which objects have which attributes may formally be represented by a binary relation I between X and Y, i.e. $I \subseteq X \times Y$, and may be visualized by a table (matrix) that contains 1s and 0s, according to whether the object corresponding to a row has the attribute corresponding to a column (for this we suppose some orders of

objects and attributes are fixed). We denote the entries of such matrix by I_{xy}. A data of this type is called *binary data* (or Boolean data). The triplet $\langle X, Y, I\rangle$ is called a *formal context* in FCA but other terms are used in other areas.

Such type of data appears in two roles in our paper. First, association rules, whose interestingness measures we analyze, are certain dependencies over the binary data. Second, the information we have about the interestingness measures of association rules is in the form of binary data: the objects are interestingness measures and the attributes are their properties.

2.2 Association rules

An *association rule* [36] over a set Y of attributes is a formula

$$A \Rightarrow B \tag{1}$$

where A and B are sets of attributes from Y, i.e. $A, B \subseteq Y$. Let $\langle X, Y, I\rangle$ be a formal context. A natural measure of interestingness of association rules is based on the notions of confidence and support. The *confidence* and *support* of an association rule $A \Rightarrow B$ in $\langle X, Y, I\rangle$ is defined by

$$\mathrm{conf}(A \Rightarrow B) = \frac{|A^{\downarrow} \cap B^{\downarrow}|}{|A^{\downarrow}|} \quad \text{and} \quad \mathrm{supp}(A \Rightarrow B) = \frac{|A^{\downarrow} \cap B^{\downarrow}|}{|X|},$$

where $C^{\downarrow}$ for $C \subseteq Y$ is defined by $C^{\downarrow} = \{x \in X \mid \text{ for each } y \in C : \langle x, y\rangle \in I\}$. An association rule is considered interesting if its confidence and support exceed some user-specified thresholds. However, the support-confidence approach reveals some weaknesses. Often, this approach as well as algorithms based on it lead to the extraction of an exponential number of rules. Therefore, it is impossible to validate it by an expert. In addition, the disadvantage of the support is that sometimes many rules that are potentially interesting, have a lower support value and therefore can be eliminated by the pruning threshold *minsupp*. To address this problem, many other measures of interestingness have been proposed in the literature [13], mainly because they are effective for mining potentially interesting rules and capture some aspects of user interest. The most important of those measures are subject to our analysis and are surveyed in Section 3.1. Note that association rules are attributed to [1]. However, the concept of association rule itself as well as various measures of interestingness are particular cases of what is investigated in depth in [18], a book that develops logico-statistical foundations of the GUHA method [19].

2.3 Factor analysis of binary (Boolean) data

Let I be an $n \times m$ binary matrix. The aim in Boolean factor analysis is to find a decomposition

$$I = A \circ B \tag{2}$$

of I into an $n \times k$ binary matrix A and a $k \times m$ binary matrix B with $\circ$ denoting the Boolean product of matrices, i.e.

$$(A \circ B)_{ij} = \max_{l=1}^{k} \min(A_{il}, B_{lj}).$$

The inner dimension, k, in the decomposition may be interpreted as the number of factors that may be used to describe the original data. Namely, $A_{il} = 1$ if and only if the lth factor applies to the ith object and $B_{lj} = 1$ if and only if the jth attribute is one of the manifestations of the lth factor. The factor model behind (2) has therefore the following meaning: The object i has the attribute j if and only if there exists a factor l that applies to i and for which j is one of its particular manifestations. We refer to [3] for further information and references to papers that deal with the problem of factor analysis and decompositions of binary matrices.

In [3], the following method for finding decompositions (2) with the number k of factors as small as possible has been presented. The method utilizes formal concepts of the formal context $\langle X, Y, I \rangle$ as factors, where $X = \{1, \ldots, n\}$, $Y = \{1, \ldots, m\}$ (objects and attributes correspond to the rows and columns of I). Let

$$\mathcal{F} = \{\langle C_1, D_1 \rangle, \ldots, \langle C_k, D_k \rangle\}$$

be a set of formal concepts of $\langle X, Y, I \rangle$, i.e. $\langle C_l, D_l \rangle$ are elements of the concept lattice $\mathcal{B}(X, Y, I)$ [11]. Consider the $n \times k$ binary matrix $A_{\mathcal{F}}$ and a $k \times m$ binary matrix $B_{\mathcal{F}}$ defined by

$$(A_{\mathcal{F}})_{il} = 1 \text{ iff } i \in C_l \quad \text{and} \quad (B_{\mathcal{F}})_{lj} = 1 \text{ iff } j \in D_l. \tag{3}$$

Denote by $\rho(I)$ the smallest number k, so-called Schein rank of I, such that a decomposition of I exists with k factors. The following theorem shows that using formal concepts as factors as in (3) enables us to reach the Schein rank, i.e. is optimal [3]:

Theorem 1. *For every binary matrix I, there exists $\mathcal{F} \subseteq \mathcal{B}(X, Y, I)$ such that $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$ and $|\mathcal{F}| = \rho(I)$.*

As has been demonstrated in [3], a useful feature of using formal concepts as factors is the fact that formal concepts may easily be interpreted. Namely, every factor, i.e. a formal concept $\langle C_l, D_l \rangle$, consists of a set C_l of objects (objects are measures of interestingness in our case) and a set D_l of attributes (properties of measures in our case). C_l contains just the objects to which all the attributes from D_l apply and D_l contains all attributes shared by all objects from C_l. From a clustering point of view, the factors $\langle C_l, D_l \rangle$ may thus be seen as clusters C_l with their descriptions by attributes from D_l. The factors thus have a natural, easy to understand meaning. Since the problem of computing the smallest set of factors is NP-hard, a greedy approximation algorithm was proposed in [3, Algorithm 2]. This algorithm is utilized below in our paper.

3 Clustering interestingness measures using Boolean factors

3.1 Measures of interestingness

In the following, we present the interestingness measures reported in the literature and recall nineteen of their most important properties that were proposed in the literature.

To identify interesting association rules and to enable the user to focus on what is interesting for him, about sixty interestingness measures [20], [35], [10] were proposed in the literature. All of them are defined using the following parameters: $p(XY)$, $p(\bar{X}Y)$, $p(X\bar{Y})$ and $p(\bar{X}\bar{Y})$, where $p(XY) = \frac{n_{XY}}{n}$ represents the number of objects satisfying XY (the intersection of X and Y), and $\bar{X}$ is the negation of X. The following are important examples of interestingness measures:

Lift *[6]:* Given a rule $X \rightarrow Y$, *lift* is the ratio of the probability that X and Y occur together to the multiple of the two individual probabilities for X and Y, i.e.,

$$Lift(X \rightarrow Y) = \frac{p(XY)}{p(X) \times p(Y)}.$$

If this value is 1, then X and Y are independent. The higher this value, the more likely that the existence of X and Y together in a transaction is not just a random occurrence, but because of some relationship between them.

Correlation coefficient *[31]: Correlation* is a symmetric measure evaluating the strength of the itemsets' connection. It is defined by

$$Correlation = \frac{p(XY) - p(X)p(Y)}{\sqrt{p(X)p(Y)p(\bar{X})p(\bar{Y})}}.$$

A correlation around 0 indicates that X and Y are not correlated. The lower is its value, the more negatively correlated X and Y are. The higher is its value, the more positively correlated they are.

Conviction *[6]: Conviction* is one of the measures that favor counter-examples. It is defined by

$$Conviction = \frac{p(X)p(\bar{Y})}{p(X\bar{Y})}$$

Conviction which is not a symmetric measure, is used to quatify the deviation from independence. If its value is 1, then X and Y are independent.

M_{GK} [15]: M_{GK} is an interesting measure, which allows the extraction of negative rules.

$$M_{GK} = \frac{p(Y/X)-p(Y)}{1-p(Y)}, \quad \text{if } X \text{ favorise } Y$$
$$M_{GK} = \frac{p(Y/X)-p(Y)}{p(Y)}, \quad \text{if } X \text{ defavorise } Y$$

It takes into account several situations of references: in the case where the rule is situated in the attractive zone (i.e. $p(Y/X) > p(Y)$), this measure evaluates the distance between independence and logical implication. Thus, the higher the value of M_{GK} is close to 1, the more the rule is close to the logical implication and the higher the value of M_{GK} is close to 0, the more the rule is close to the independence. In the case where the rule is located in the repulsive zone (i.e. $p(Y/X) < p(Y)$), M_{GK} evaluates this time a distance between the independence and the incompatibility. Thus, the closer the value of M_{GK} is to -1, the more similar to incompatibility the rule is; and the closer the value of M_{GK} is to 0, the closer to the independence the rule is.

As was mentioned above, several studies [35], [23], [25], [13] were reported in the literature on the various properties of interestingess measures to be able to characterize and evaluate the interestingness measures. The main goal of researchers in the domain is then to provide a user assistance in choosing the best interestingness measure meeting his needs. For that, formal properties have been developed [32], [24], [35], [12], [4] in order to evaluate the interestingness measures and to help users understanding their behavior. In the following, we present nineteen properties reported in the literature.

3.2 Properties of the measures

Figure 1 lists 19 properties of interestingness measures. The properties are described in detail in [16]; we omit details due to lack of space.

The authors in [14] proposed an evaluation of 61 interestingness measures according to the 19 properties (P_3 to P_{21}). Properties P_1 and P_2 were not taken into account in this study because of their subjective character. The measures and their properties result in a binary measure-property matrix that is used for clustering the measures according to their properties. The clustering performed in [14] using the agglomerative hierarchical method and the K-means method revealed 7 clusters of measures which will be used in the next section in a comparison with the results obtained by Boolean factor analysis applied on the same measure-property matrix.

3.3 Clustering using Boolean factors

The measure-property matrix describing interestingness measures by their properties is depicted in Figure 2. It consists of 62 measures (61 measures from [14] plus one more that has been studied recently) described by 21 properties because the three-valued property P_{14} is represented by three yes-no properties

No.	Property	Ref.
P_1	Intelligibility or comprehensibility of measure	[25]
P_2	Easiness to fix a threshold to the rule	[23]
P_3	Asymmetric measure.	[35], [23]
P_4	Asymmetric measure in the sense of the conclusion negation.	[23], [35]
P_5	Measure assessing in the same way $X \rightarrow Y$ and $\bar{Y} \rightarrow \bar{X}$ in the logical implication case.	[23]
P_6	Measure increasing function the number of examples or decreasing function the number of counter-examples.	[32], [23]
P_7	Measure increasing function the data size.	[12], [35]
P_8	Measure decreasing function the consequent/antecedent size.	[23], [32]
P_9	Fixed value a in the independence case.	[23], [32]
P_{10}	Fixed value b in the logical implication case.	[23]
P_{11}	Fixed value c in the equilibrium case.	[5]
P_{12}	Identified values in the attraction case between X and Y.	[32]
P_{13}	Identified values in the repulsion case between X and Y.	[32]
P_{14}	Tolerance to the first counter-example.	[23], [38]
P_{15}	Invariance in case of expansion of certain quantities.	[35]
P_{16}	Desired relationship between $X \rightarrow Y$ and $\bar{X} \rightarrow Y$ rules.	[35]
P_{17}	Desired relationship between $X \rightarrow Y$ and $X \rightarrow \bar{Y}$ antinomic rules.	[35]
P_{18}	Desired relationship between $X \rightarrow Y$ and $\bar{X} \rightarrow \bar{Y}$ rules.	[35]
P_{19}	Antecedent size is fixed or random.	[23]
P_{20}	Descriptive or statistical measure.	[23]
P_{21}	Discriminant measure.	[23]

Fig. 1. Interestingness measures properties.

$P_{14.1}$, $P_{14.2}$, and $P_{14.3}$. We computed the decomposition of the matrix using Algorithm 2 from [3] and obtained 28 factors (as in the case below, several of them may be disregarded as not very important; we leave the details for a full version of this paper). In addition, we extended the original 62×21 binary matrix by adding for every property its negation, and obtained a 62×42 binary matrix. The reason for adding negated properties is due to our goal to compare the results with the two clustering methods mentioned above and the particular role of the properties and their negations in these clustering methods. From the 62×42 matrix, we obtained 38 factors, denoted $F_1, \ldots, F_{38}$. The factors are presented in Figures 3 and 4. Figure 3 depicts the object-factor matrix describing the interestingness measures by factors, Figure 4 depicts the factor-property matrix explaining factors by properties of measures. Factors are sorted from the most important to the least important, where the importance is determined by the number of 1s in the input measure-property matrix covered by the factor [3]. The first factors cover a large part of the matrix, while the last ones cover only a small part and may thus be omitted [3], see the graph of cumulative cover of the matrix by the factors in Figure 5.

4 Interpretation and comparison to other approaches

The aim of this section is to provide an interpretation of the results described in the previous section and compare them to the results already reported in the literature, focusing mainly on [14]. As was described in the previous section, 38 factors were obtained. The first 21 of them cover 94 % of the input measure-property matrix (1s in the matrix), the first nine cover 72 %, and the first five

	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14.1	P15	P16	P17	P18	P19	P20	P21	P14.2	P14.3
correlation	0	1	1	1	1	1	1	0	0	1	1	0	0	1	1	1	0	0	1	1	0
Cohen	0	1	1	1	1	1	1	0	0	1	1	0	0	0	0	1	0	0	1	1	0
confidence	1	1	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	1	0
causal confidence	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0
Pavillon	1	1	0	1	1	1	1	0	0	1	1	0	0	0	1	0	0	0	1	1	0
Ganascia	1	1	1	1	0	0	0	1	1	0	0	0	0	0	1	0	0	0	1	1	0
causal confirmation	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
descriptive confirmation	1	1	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	1	0
conviction	1	1	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0	0	1	0	1
cosine	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
coverage	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
dependency	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0
causal dependency	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
weighted dependency	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
Bayes factor	1	1	0	1	1	1	1	0	0	1	1	0	1	0	0	0	0	0	1	0	1
Loevinger	1	1	1	1	1	1	1	1	0	1	1	0	0	0	0	0	0	0	1	1	0
collective strength	0	1	1	1	1	1	1	0	0	1	1	0	0	0	0	1	0	0	1	0	1
Fukuda	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
information gain	0	1	0	1	1	1	1	0	0	1	1	1	0	0	0	0	0	0	1	0	0
Goodman	0	1	1	1	1	0	1	1	0	1	1	0	0	1	1	1	0	0	1	1	0
implication index	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0
IPEE	1	1	1	1	0	0	0	0	1	0	0	1	0	0	0	0	1	1	0	0	0
IP3E	1	1	1	1	0	0	0	0	1	0	0	1	0	0	0	0	1	1	1	0	0
PDI	0	1	1	1	0	1	0	0	0	0	0	1	1	0	0	0	1	1	1	0	0
II	1	1	1	1	1	1	1	0	0	1	1	1	1	0	0	0	1	1	0	0	0
EII	1	1	1	1	1	0	0	0	0	0	0	1	0	0	0	0	1	1	1	0	0
REII	1	1	1	1	1	0	1	0	0	1	1	1	0	0	0	0	1	1	1	0	0
likelihood index	0	1	1	1	1	1	1	0	0	1	1	1	0	0	0	0	1	1	0	0	0
interest	0	1	0	1	1	1	1	0	0	1	1	0	0	0	0	0	0	0	1	1	0
Jaccard	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
Jmeasure	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	1
Klosgen	1	1	0	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0	1	0	1
Laplace	1	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0
Mgk	1	1	1	1	1	0	1	1	0	1	1	0	0	0	1	0	0	0	1	1	0
least contradiction	1	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0
Pearl	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	1	1	0
Piatetsky-Shapiro	0	1	1	1	1	1	1	0	0	1	1	0	0	1	1	1	0	1	1	1	0
precision	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0
prevalence	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
YuleQ	0	1	1	1	1	0	1	1	0	1	1	1	1	1	1	1	0	0	1	0	0
recall	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
Gini	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1
relative risk	1	1	0	1	1	1	1	0	0	1	1	0	0	0	0	0	0	0	1	0	1
Sebag	1	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1
support	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
one way support	1	1	0	0	1	1	1	0	0	1	1	0	0	0	0	0	0	0	1	0	1
two way support	0	1	0	0	1	1	1	0	0	1	1	0	0	0	0	0	0	0	1	0	1
examples and counter-examples rate	1	1	1	1	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0
VT100	0	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0	1	1	0
variation support	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	1	0	1
YuleY	0	1	1	1	1	0	1	1	0	1	1	0	1	1	1	1	0	0	1	0	1
Zhang	1	1	1	1	1	0	1	1	0	1	1	1	1	0	1	0	0	0	1	0	0
causal confirmed confidence	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0
Czekanowski	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
negative reliability	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0
mutual information	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
Kulczynski	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
Leverage	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
novelty	0	1	1	1	1	1	1	0	0	1	1	0	0	1	1	1	0	0	1	1	0
odds ratio	0	1	1	1	1	1	1	0	0	1	1	0	0	0	0	1	0	0	1	0	1
specificity	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
causal support	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0

Fig. 2. Input binary matrix describing interestingness measures by their properties.

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24	F25	F26	F27	F28	F29	F30	F31	F32	F33	F34	F35	F36	F37	F38
correlation	1	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0
Cohen	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0
confidence	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0
causal confidence	0	1	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0
Pavillon	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Ganascia	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0
causal confirmation	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0
descriptive confirmation	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0
conviction	1	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1		
cosine	0	1	0	0	1	0	0	0	0	0	1	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	
coverage	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0		
dependency	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0		
causal dependency	0	1	0	0	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0		
weighted dependency	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1
Bayes factor	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1				
Loevinger	1	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0			
collective strength	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1		
Fukuda	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0			
information gain	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1		
Goodman	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	1	0	1	0			
implication index	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0				
IPEE	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	0	0	1	0	1	0	0		
IP3E	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	1	0	1	0	0			
PDI	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	1	0			
II	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0			
EII	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	1	0	1	0	0		
REII	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0		
likelihood index	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	1	0	1	1	0		
interest	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	
Jaccard	0	0	0	0	1	0	1	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1			
Jmeasure	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	1		
Klosgen	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1			
Laplace	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0				
Mgk	1	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0				
least contradiction	0	1	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0				
Pearl	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0			
Piatetsky-Shapiro	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0			
precision	0	1	0	0	0	1	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0		
prevalence	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0			
YuleQ	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	
recall	0	1	0	0	1	0	0	0	0	1	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0			
Gini	0	0	1	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	1		
relative risk	1	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1			
Sebag	0	0	0	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1			
support	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0			
one way support	1	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1			
two way support	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	1			
examples and counter-examples rate	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	1			
VT100	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	1	1	0				
variation support	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	1	1			
YuleY	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	1	0	1	1				
Zhang	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1			
causal confirmed confidence	0	1	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0			
Czekanowski	0	1	0	0	1	0	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0			
negative reliability	0	1	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0			
mutual information	0	0	1	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	1				
Kulczynski	0	0	0	0	1	0	1	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1				
Leverage	0	1	0	0	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0				
novelty	1	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	1	0				
odds ratio	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	1			
specificity	0	1	0	0	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0			
causal support	0	1	0	0	0	1	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0		

Fig. 3. Interestingness measures described by factors obtained by decomposition of the input matrix from Figure 2 extended by negated properties.

	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14.1	P15	P16	P17	P18	P19	P20	P21	P14.2	P14.3	$\overline{\text{P3}}$	$\overline{\text{P4}}$	$\overline{\text{P5}}$	$\overline{\text{P6}}$	$\overline{\text{P7}}$	$\overline{\text{P8}}$	$\overline{\text{P9}}$	$\overline{\text{P10}}$	$\overline{\text{P11}}$	$\overline{\text{P12}}$	$\overline{\text{P13}}$	$\overline{\text{P14.1}}$	$\overline{\text{P15}}$	$\overline{\text{P16}}$	$\overline{\text{P17}}$	$\overline{\text{P18}}$	$\overline{\text{P19}}$	$\overline{\text{P20}}$	$\overline{\text{P21}}$	$\overline{\text{P14.2}}$	$\overline{\text{P14.3}}$
F1	0	1	0	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0
F2	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1	0	0	1	1	0	0	0	1
F3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	1	1	1	1	1	0	1	0	0	0	0	0
F4	1	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	1	0	1	1	0	1	0	0	0	0	0	0
F5	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	1	1	1	1	1	0	0	0
F6	0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	1
F7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	1	0	1	1	0	1	0
F8	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1	0	0	0	1	1
F9	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
F10	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0	1	1	0	0	0	1
F11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	1	1	0	0	0	0	0	0	0
F12	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1
F13	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
F14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1	1	1	1	1	1	0	0	1	1	0	0	0	0
F15	1	1	0	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	1	0	0
F16	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
F17	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1	1	0	1	1	1	0	1	1	0	1	1	1	1	1	1	0	1	1	0	0	0	0
F18	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
F19	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0
F20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1	1	0	1	1	0	0	0	0	0
F21	0	1	0	1	1	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
F22	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	1	
F23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0
F24	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1	1	0	1	1	1	1	1	1	0	1	1	0	0	0	0	0
F25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1
F26	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	0	1
F27	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	1
F28	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	1	1	1	1	0	1	0	1	0	0	0	0	0	0
F29	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	1	0	1	1	1	0	0	1	1	1	0	0	0	1
F30	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	0	0	0	0	0	0	0
F31	0	1	1	1	1	0	1	1	0	1	1	1	1	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	1	1	0	1	1	
F32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	1
F33	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	1	0	1	0	
F34	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
F35	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	
F36	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
F37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	

Fig. 4. Factors obtained by decomposition of the input matrix from Figure 2 extended by negated properties. The factors are described in terms of the original and negated properties.

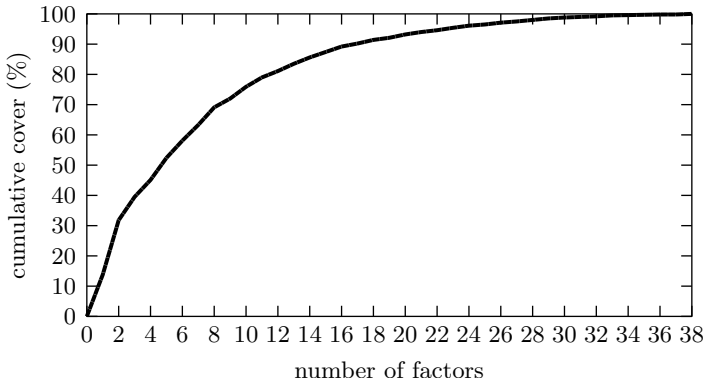


Fig. 5. Cumulative cover of input matrix from Figure 2 extended by negated properties by factors obtained by decomposition of the matrix.

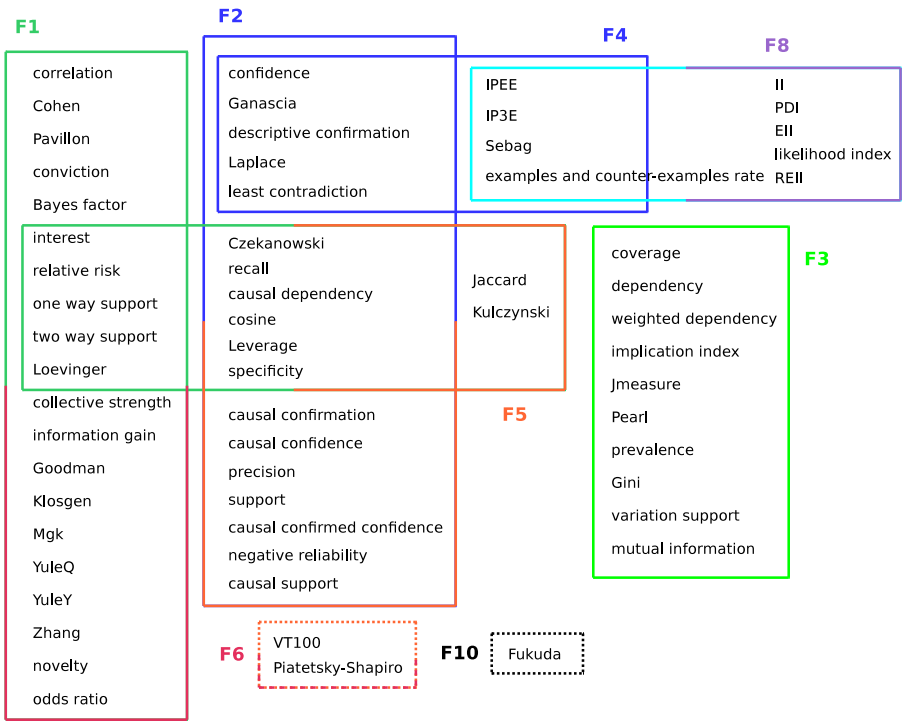


Fig. 6. Venn diagram of the first five factors (plus the eighth and part of the sixth and tenth to cover the whole set of measures) obtained by decomposition of the input matrix from Figure 2 extended by negated properties.

cover 52.4 %. Another remark is that the first ten factors cover the whole set of measures.

Note first that the Boolean factors represent overlapping clusters, contrary to the clustering using the agglomerative hierarchical method and the K-means method performed in [14]. Namely, the clusterings are depicted in Figure 6 describing the Venn diagram of the first five Boolean factors (plus the eighth and part of the sixth and tenth to cover the whole set of measures) and Figure 7, which is borrowed from [14], describing the consensus on the classification obtained by the hierarchical and K-means clusterings. This consensus refunds the classes C_1 to C_7 of the extracted measures, which are common to both techniques.

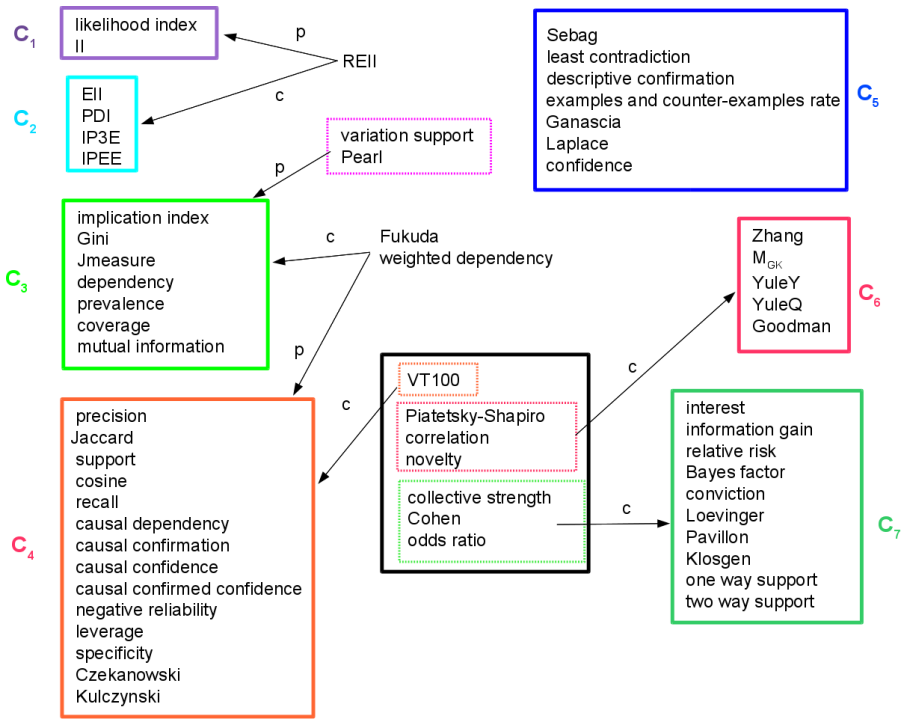


Fig. 7. Classes of measures obtained by the hierarchical and K-means clusterings.

Due to lack of space, we focus on the first four factors since they cover nearly half of the matrix (45.1 %), and also because most of the measures appear at least once in the four factors.

Factor 1. The first factor F_1 applies to 20 measures, see Figure 3, namely: correlation, Cohen, Pavillon, conviction, Bayes factor, Loevinger, collective strength, information gain, Goodman, interest, Klosgen, Mgk, YuleQ, relative risk, one

way support, two way support, YuleY, Zhang, novelty, and odds ratio. These measures share the following 9 properties: P4, P7, P9, not P11, P12, P13, not P19, not P20, P21, see Figure 4.

Interpretation. The factor applies to measures whose evolutionary curve increases w.r.t the number of examples and have a fixed point in the case of independence (this allows to identify the attractive and repulsive area of a rule). The factor also applies only to descriptive and discriminant measures that are not based on a probabilistic model.

Comparison. When looking at the classification results reported in [14], F_1 covers two classes from [14]: C_6 and C_7, which together contain 15 measures. Those classes are closely related within the dendrogram obtained with the agglomerative hierarchical clustering method used in [14]. The 5 missing measures form a class obtained with K-means method in [14] with Euclidian distance.

Factor 2. F_2 applies to 18 measures, namely: confidence, causal confidence, Ganascia, causal confirmation, descriptive confirmation, cosine, causal dependency, Laplace, least contradiction, precision, recall, support, causal confirmed confidence, Czekanowski, negative reliability, Leverage, specificity, and causal support. These measures share the following 11 properties: P4, P6, not P9, not P12, not P13, P14.2, not P15, not P16, not P19, not P20, P21.

Interpretation. The factor applies to measures whose evolutionary curve increases w.r.t. the number of examples and has a variable point in the case of independence, which implies that the attractive and repulsive areas of a rule are not identifiable. The factor also applies only to measures that are not discriminant, are indifferent to the first counter-examples, and are not based on a probabilistic model.

Comparison. F_2 corresponds to two classes, C_4 and C_5 reported in [14]. $C_4 \cup C_5$ contains 22 measures. The missing measures are: Jaccard, Kulczynski, examples and counter-examples rate and Sebag. Those measures are not covered by F_2 since they are not indifferent to the first counter-examples.

Factor 3. F_3 applies to 10 measures, namely: coverage, dependency, weighted dependency, implication index, Jmeasure, Pearl, prevalence, Gini, variation support, and mutual information. These measures share the following 10 properties: not P6, not P8, not P10, not P11, not P13, not P14.1, not P15, not P16, not P17, not P19.

Interpretation. The factor applies to measures whose evolutionary curve does not increase w.r.t. the number of examples.

Comparison. F_3 corresponds to class C_3 reported in [14], which contains 8 measures. The two missing measures, variation support and Pearl, belong to the same classes obtained by both K-means and the hierarchical method. Moreover, these two missing measures are similar to those from C_3 obtained by the hierarchical method since they merge with the measures in C_3 at the next level of the generated dendrogram. Here, there is a strong correspondence between results obtained using Boolean factors and the ones reported in [14].

Factor 4. F_4 applies to 9 measures, namely: confidence, Ganascia, descriptive confirmation, IPEE, IP3E, Laplace, least contradiction, Sebag, and examples and

counter-examples rate. These measures share the following 12 properties: P3, P4, P6, P11, not P7, not P8, not P9, not P12, not P13, not P15, not P16, not P18.

Interpretation. The factor applies to measures whose evolutionary curve increases w.r.t. the number of examples and has a fixed value in the equilibrium case. As there is no fixed value in the independence case, we can not get an identifiable area in the case of attraction or repulsion.

Comparison. F_4 mainly applies to measures of class C_5 obtained in [14]. The two missing measures, IPEE et IP3E, belong to a different class.

5 Conclusions and further issues

We demonstrated that Boolean factors provide us with clearly interpretable meaningful clusters of measures among which the first ones are highly similar to other clusters of measures reported in the literature. Contrary to other clustering methods, Boolean factors represent overlapping clusters. We consider this an advantage because overlapping clusters are a natural phenomenon in human classification. We presented preliminary results on clustering the measures using Boolean factors. Due to limited scope, we presented only parts of the results obtained and leave other results for a full version of this paper.

An interesting feature of the presented method, to be explored in the future, is that the method need not start from scratch. Rather, one or more clusters, that are considered important classes of measures, may be supplied at the start and the method may be asked to complete the clustering. Another issue left for future research is the benefit of the clustering of measures for a user who is interested in selecting a type of measure, rather than a particular measure of interestingness of association rules. In the intended scenario, a user may use various interestingness measures that belong to different classes of measures.

References

1. Agrawal R., Imielinski T., Swami A.: Mining association rules between sets of items in large databases. *Proc. ACM SIGMOD* 1993, 207–216.
2. Agrawal R., Srikant R.: Fast algorithms for mining association rules. *Proc. VLDB Conf.* 1994, 478–499.
3. Belohlavek R., Vychodil V.: Discovery of optimal factors in binary data via a novel method of matrix decomposition. *J. of Computer and System Sciences* **76**(1)(2010), 3–20.
4. Blanchard J., Guillet F., Briand H., Gras R.: Assessing rule with a probabilistic measure of deviation from equilbrium. In *Proc. Of 11th International Symposium on Applied Stochastic Models and Data Analysis ASMDA* 2005, Brest, France, 191–200.
5. Blanchard J., Guillet F., Briand H., Gras R.: IPEE: Indice Probabiliste d'Écart à l'Équilibre pour l'évaluation de la qualité des règles. *Dans l'Atelier Qualité des Données et des Connaissances* 2005, 26–34.
6. Brin S., Motwani R., Silverstein C.: Beyond Market Baskets: Generalizing Association Rules to Correlations. In Proc. of the ACM SIGMOD Conference, Tucson, Arizona, 1997, 265–276.

7. Carpineto C., Romano G.: *Concept Data Analysis. Theory and Applications.* J. Wiley, 2004.
8. Davey B. A., Priestley H.: *Introduction to Lattices and Order.* Cambridge University Press, Oxford, 1990.
9. Delgado M., Ruiz D.-L., Sanchez D.: Studying Interest measures for association rules through a logical model. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* **18**(1)(2010), World Scientific, 87–106.
10. Feno D.R.: Mesures de qualité des règles d'association: normalisation et caractérisation des bases. PhD thesis, Université de La Réunion, 2007.
11. Ganter B., Wille R.: *Formal Concept Analysis. Mathematical Foundations.* Springer, Berlin, 1999.
12. Geng L., Hamilton H.J.: Choosing the Right Lens: Finding What is Interesting in Data Mining. *Quality Measures in Data Mining* 2007, ISBN 978-3-540-44911-9, 3–24.
13. Geng L., Hamilton H. J.: Interestingness measures for data mining: A Survey. *ACM Comput. Surveys* **38**(3)(2006), 1–31.
14. Guillaume S., Grissa D., Mephu Nguifo E.: Catégorisation des mesures d'intérêt pour l'extraction des connaissances. *Revue des Nouvelles Technologies de l'Information*, 2011, to appear (previously available as Technical Report RR-10-14, LIMOS, ISIMA, 2010).
15. Guillaume S.: Traitement des données volumineuses. Mesures et algorithmes d'extraction des règles d'association et règles ordinales. PhD thesis. Université de Nantes, France, 2000.
16. Guillaume S., Grissa D., Mephu Nguifo E.: Propriétés des mesures d'intérêt pour l'extraction des règles. *Dans l'Atelier Qualité des Données et des Connaissances, EGC'2010*, 2010, Hammamet-Tunisie, *http://qdc2010.lri.fr/fr/actes.php*, 15–28.
17. Grissa D., Guillaume S., Mephu Nguifo E.: Combining Clustering techniques and Formal Concept Analysis to characterize Interestingness Measures. *CoRR* **abs/1008.3629**, 2010.
18. Hájek P., Havránek T.: *Mechanizing Hypotheses Formation.* Springer, 1978.
19. Hájek P., Holeňa, Rauch J.: The GUHA method and its meaning for data mining. *J. Computer and System Sciences* **76**(2010), 34–48.
20. Hilderman R. J., Hamilton H. J.: Knowledge Discovery and Measures of Interest, *Volume 638 of The International Series in Engineering and Computer Science* **81**(2)(2001), Kluwer.
21. Huynh X.-H., Guillet F., Briand H.: Clustering Interestingness Measures with Positive Correaltion. **ICEIS** (2) (2005), 248–253.
22. Heravi M. J., Zaïane O. R.: A study on interestingness measures for associative classifiers. **SAC** (2010), 1039–1046.
23. Lallich S., Teytaud, O.: Évaluation et validation de mesures d'intérêt des règles d'association. *RNTI-E-1, numéro spécial* 2004, 193–217.
24. Lenca P, Meyer P., Picouet P., Vaillant B., Lallich S.: Critères d'évaluation des mesures de qualité en ecd. *Revue des Nouvelles Technologies de l'Information (Entreposage et Fouille de données)* (1)(2003), 123–134.
25. Lenca P., Meyer P., Vaillant B., Lallich, S.: A multicriteria decision aid for interestingness measure selection. Technical Report LUSSI-TR-2004-01-EN, Dpt. LUSSI, ENST Bretagne 2004 (chapter 1).
26. Liu J., Mi J.-S.: A novel approach to attribute reduction in formal concept lattices. RSKT 2006, *Lecture Notes in Artificial Intelligence* **4062** (2006), 522–529.

27. Maddouri M., Gammoudi J.: On Semantic Properties of Interestingness Measures for Extracting Rules from Data. *Lecture Notes in Computer Science* **4431** (2007), 148–158.
28. Maier D.: *The Theory of Relational Databases.* Computer Science Press, Rockville, 1983.
29. Pawlak Z.: Rough sets. *Int. J. Information and Computer Sciences* **11**(5)(1982), 341–356.
30. Pawlak Z.: *Rough Sets: Theoretical Aspcets of Reasoning About Data.* Kluwer, Dordrecht, 1991.
31. Pearson K.: Mathematical contributions to the theory of evolution, regression, heredity and panmixia. Philosophical Trans. of the Royal Society A (1896).
32. Piatetsky-Shapiro G.: Discovery, Analysis and Presentation of Strong Rules. In G. Piatetsky-Shapiro & W.J. Frawley, editors: *Knowledge Discovery in Databases.* AAAI Press, 1991, 229–248.
33. Polkowski L.: *Rough Sets: Mathematical Foundations.* Springer, 2002.
34. Sese J., Morishita S.: Answering the most correlated n association rules efficiently. In *Proceedings of the 6th European Conf on Principles of Data Mining and Knowledge Discovery* 2002, Springer-Verlag, 410–422.
35. Tan P.-N., Kumar V., Srivastava J.: Selecting the right objective measure for association analysis. *Information Systems* **29**(4)(2004), 293–313.
36. Tan P.-N., Steinbach M., Kumar V.: *Introduction to Data Mining.* Addison-Wesley, 2005.
37. Vaillant B., Lenca P., Lallich S.: A Clustering of Interestingness Measures. *DS'04, the 7th International Conference on Discovery Science* **LNAI 3245** (2004), 290–297.
38. Vaillant B.: Mesurer la qualité des règles d'association: études formelles et expérimentales. PhD thesis, ENST Bretagne, 2006.
39. Wang X., Ma J.: A novel approach to attribute reduction in concept lattices. RSKT 2006, *Lecture Notes in Artificial Intelligence* **4062** (2006), 522–529.
40. Wille R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Rival I.: *Ordered Sets.* Reidel, Dordrecht, Boston, 1982, 445–470.
41. Zhang W.-X., Wie L., Qi J.-J.: Attribute reduction in concept lattices based on discernibility matrix. RSFDGrC 2005, *Lecture Notes in Artificial Intelligence* **3642** (2005), 157–165.

Combining Formal Concept Analysis and Translation to Assign Frames and Thematic Role Sets to French Verbs

Ingrid Falk[1], Claire Gardent[2]

[1] INRIA/Nancy Universités, Nancy (France)
[2] CNRS/LORIA, Nancy (France)

Abstract. We present an application of Formal Concept Analysis in the domain of Natural Language Processing: We give a general overview of the framework, describe its goals, the data it is based on, the way it works and we illustrate the kind of data we expect as a result. More specifically, we examine the ability of the stability, separation and probability indices to select the most relevant concepts with respect to our FCA application. We show that the sum of stability and separation gives results close to those obtained when using the entire lattice.

1 Introduction

Ideally natural language processing (NLP) applications need to analyse texts to answer the question of "Who did What to Whom". For computers to effectively extract this information from texts, it is essential that they be able to detect the events that are being described and the event participants. Because events are mostly lexicalised using verbs, one ingredient that is essential for such systems is detailed knowledge about their syntactic and semantic behaviour. It has been shown (Briscoe and Carroll (1993), Carroll and Fang (2004)) that detailed subcategorisation information (that is, information about the number and the syntactic type of verb complements) is crucial in enhancing their linguistic coverage and theoretical accuracy. However this syntactic information is not sufficient to specify "Who did what to Whom" because it does not allow to identify the thematic roles participating in the event described by the verb. For example in *John threw a ball to Mary* the syntactic analysis of the sentence would not allow to identify John which is the syntactic subject of the sentence as the Agent or Causer of the *throwing* event, Mary, syntactically the prepositional object as the Destination and *ball* (the object) as the item being *thrown*.

To help computer systems in this task of understanding and representing the full meaning of a text, verb classifications have been proposed which group together verbs with similar syntactic and semantic behaviour, ie. which associate groups of verbs with subcategorisation frames showing the syntactic constructions the verbs may appear in and sets of thematic roles which represent the participants in an event described by the verbs in the group.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

For English, there exist several large scale resources providing verb classes (*eg.* Framenet Baker et al. (1998) and VerbNet Schuler (2006), the classification we use in our framework) in a format that is amenable for use by natural language processing systems. For example for the verb *throw* the corresponding VerbNet class shows that the participants in a *throwing* event are an Agent, a Theme (the thing being *thrown*), a Source and a Destination. In addition, the VerbNet class provides the syntactic constructions the verb can occur in (eg. SUBJECT(*John*) V(*throws*) OBJECT(*a ball*) PREPOBJECT(*to Mary*)) and shows how the participant roles can be realised as syntactic arguments: In the example above the Agent (*John*) is realised syntactically as SUBJECT, the Theme (*the ball*) as OBJECT and the Destination (*to Mary*) as prepositional object (PREPOBJECT).

For French however, existing verb classes are either too restricted in scope (Volem Saint-Dizier (1999)) or not sufficiently structured (the LADL tables Gross (1975)) to be directly useful for NLP. Even though recently other large coverage syntactic-semantic resources for French have been made available (Tolone (2011) as well as further processed versions of Dubois and Dubois-Charlier (1997), Hadouche and Lapalme (2010)) the terminology and linguistic formalisms they are based on is often still hardly compatible with the methods and tools currently used in the NLP community.

In this paper we present a method for providing a VerbNet style classification of French verbs which associates verbs with syntactic constructions on the one hand and sets of semantic role sets (the set of semantic roles participating in the event described by the verb) on the other. To obtain this classification, we build and combine two independent classifications. The first is semantic and is obtained from the English VerbNet (VN) by translation, the second is syntactic and is obtained by building an FCA (Formal Concept Analysis) lattice from three, manually validated syntactic lexicons for French. The first associates groups of French verbs with the semantic roles of the English VN class. The second associates groups of French verbs (the concept extent) with syntactic constructions (concept intent). We then merge both classifications by associating with each translated VN class, the FCA concept whose verb set yields the best F-measure with respect to the verb sets contained in each translated VN class. We thus effectively associate the set of semantic roles of the VN class to the group of French verbs and the syntactic information given by the FCA concept.

In the past several linguistic FCA applications have been presented, as Priss (2005) shows in her overview. For example, Sporleder (2002) describes an FCA based approach to build structured class hierarchies starting from unstructured lexicon entries while the features used for building classes in the approach presented in (Cimiano et al., 2003) are collected from a corpus. Our approach (based on earlier work presented in Falk et al. (2010), Falk and Gardent (2010)) is concerned with building a lexical resource based on lexicons and is therefore related to the FCA approach in (Sporleder, 2002). However, the features we use are different. In addition we explore the use of concept selection indices to filter the concept lattices and finally relate the formal concepts we obtain to other classes

obtained by a clustering approach based on different numeric features extracted from lexicons and English-French dictionaries.

In the following we first introduce the terminology and data used in our application domain. Next we describe how we associate groups of French verbs with syntactic information using Formal Concept Analysis (Section 3). As the resulting concept lattice has a very large number of concepts which are mostly not useful verb classes we explore methods to select the concepts most relevant to our application (Section 4). We show in particular that selecting only $\sim 10\%$ of the concepts of the lattice using indices proposed in Klimushkin et al. (2010) gives results close to those obtained when using the entire lattice. We then show how we build the translated VerbNet classes and how they are mapped to the previously pre-selected FCA concepts (Section 5). Finally in Section 6 we present the kind of associations we obtain by our method.

2 Linguistic Concepts and Resources

Our aim is to build a lexicon associating groups of French verbs with:

1) the syntactic constructions the verbs of this group may appear in,
2) the semantic roles participating in an event described by a verb of this group.

Syntactic constructions a verb may occur in are described using *subcategorisation frames* (SCF) and are usually part of a lexical entry describing the verb. A subcategorisation frame (SCF) characterises the number and the type of the syntactic arguments expected by a verb. Each frame describes a set of syntactic arguments and each argument is characterised by a grammatical function (*eg.* SUJ - subject, OBJ - direct object etc.) and a syntactic category (NP indicates a noun phrase, PP a prepositional phrase, etc.). For example *John throws a ball to Mary.* is a possible realisation of the subcategorisation frame SUJ:NP V OBJ:NP POBJ:PP.

The semantic (thematic) roles are the participants in an event described by a particular verb. To date there is no consensus about a set of semantic roles or a set of tests determining them. There may be a general agreement on a set of Semantic Roles (eg. Agent, Patient, Theme, Instrument, Location, etc.) but there is substantial disagreement on when and where they can be assigned (Palmer et al., 2010). Thus each of the well known resources (FrameNet (Baker et al., 1998), PropBank (Palmer et al., 2005), VerbNet (Schuler, 2006), LVF (Dubois and Dubois-Charlier, 1997)) providing semantic role information have their own semantic role inventory. In our work we chose the VerbNet semantic role inventory for several reasons:

1. VN semantic roles provide a compromise between generalisation and specificity in that they are common across all verbs[3] but are still able to capture specificities of particular classes.

[3] in contrast to FrameNet Baker et al. (1998) and PropBankPalmer et al. (2005) roles.

2. VN roles are among those generally agreed upon in the community.
3. None of the other resources provide the link between syntactic arguments and semantic roles across different verbs.
4. Semantic roles are expected to be valid across languages and by using the same role inventory as for English we hope to leverage some of the substantial research done for English and link syntactic information for French with semantic information provided by the English classes. Our method allows us to detect groups of French verbs with the same role set as some English VerbNet class and gives information about how these semantic roles are realised syntactically in French.

Figure 1 shows an excerpt of the *throw-17.1* VerbNet class, with its verbs, thematic roles and subcategorisation frames.

verbs (32): kick, launch, throw, tip, toss, ...
sem. roles: AGENT, THEME, SOURCE, DESTINATION

frames (8):

SCFs	**sem. roles**
Subject V Object	Agent V Theme
John throws a ball	
Subject V Object PrepObject	Agent V Theme Destination
John throws a ball to Mary	
Subject V Object Object	Agent V Destination Theme
John throws Mary a ball	
etc.	

Fig. 1: Simplified VerbNet class *throw-17.1*.

Thus, from this data an English NLP system analysing the sentence *John threw a ball to Mary* could infer the semantic roles involved in the event, namely those given by the VerbNet class. It could also detect the possible semantic roles realised by the syntactic arguments: It would know that the subject is a realisation of the Agent semantic role, the object of the Theme or Destination semantic roles, etc.

3 Associating French Verbs with Subcategorisation Frames

To associate French verbs with syntactic frames, we use the FCA classification approach where the objects are verbs and the attributes are the subcategorisation frames associated with these verbs by the subcategorisation lexicon to be described below.

3.1 Subcategorisation Lexicons

Subcategorisation information is retrieved from three existing lexicons for French: *Dicovalence* van den Eynde and Mertens (2003), *the LADL tables* Gross (1975),

Guillet and Leclère (1992) and finally *TreeLex* Kupść and Abeillé (2008). Each of these was constructed manually or with an important manual validation by linguists. The combined lexicon covers 5918 verbs, 345 SCFs and has a total of 20443 ⟨verb, frame⟩ pairs. Table 1 shows sample entries in this lexicon for the verb *expédier (send)*. Using the Galicia Lattice Builder software[4], we first build

Verb: *expédier*	
SCF	**Source info**
SUJ:NP,DUMMY:REFL	DV:41640,41650
SUJ:NP,OBJ:NP	DV:41640,41650;TL
SUJ:NP,OBJ:NP,AOBJ:PP	TL
SUJ:NP,OBJ:NP,POBJ:PP,POBJ:PP	LA:38L

Table 1: Sample entries in subcategorisation lexicon for verb *expédier (send)*.

a concept lattice based on the formal context $\langle V, F, R\rangle$ such that:

- V is the set of verbs in our subcategorisation lexicon. We ignore verbs with only one SCF as they will result in classes associating verbs with a unique frame.
- F is the set of subcategorisation frames (SCFs) present in the subcategorisation lexicon,
- R is the mapping such that $(v, f) \in R$ iff the subcategorisation lexicon associates the verb v with the SCF f.

The resulting formal context is made of 2091 objects (verbs) and 238 attributes (frames), giving rise to a lattice of 12802 concepts. Clearly however not all these concepts are interesting verb classes. Classes aim to factorise information and express generalisations about verbs. Hence, concepts with few (1 or 2) verbs can hardly be viewed as classes and similarly, concepts with few frames are less interesting.

To select from this lattice those concepts which are most likely to provide the most relevant verb-frame associations, we explore the use of three indices for concept selection: *concept stability*, *separation* and *probability* which have been proposed and analysed in (Klimushkin et al., 2010). In Section 4.2 we investigate which of these indices performs best in the context of our application. We then use the best performing concept filtering method to select the most relevant concepts with respect to our data. For each translated VN class we then identify among the selected FCA concepts the one(s) with best f-measure between precision and recall. For a translated VN class C_{VN} (consisting of French verbs) and the extent (verb set) of an FCA concept C_{FCA} precision, recall and f-measure are computed as follows: $R = \frac{|C_{VN} \cap C_{FCA}|}{|C_{VN}|}, P = \frac{|C_{VN} \cap C_{FCA}|}{|C_{FCA}|}, F = \frac{2RP}{R+P}$
The translated VN class is then associated with the FCA concept(s) with best F-measure. Thus the verbs in the FCA concept are effectively associated with the thematic roles of the translated class and at the same time with the syntactic subcategorisation frames in the intent (attribute set) of the FCA concept.

[4] `http://www.iro.umontreal.ca/~galicia/`

4 Filtering Concept Lattices

The lattices we have to deal with are very large and many of the concepts do not represent valid verb classes. To select those concepts which are most relevant in the context of our application the concept lattice needs to be filtered. Klimushkin et al. (2010) propose three indices for selecting relevant concepts in concept lattices built from noisy data: *concept stability, separation* and *probability*. In this section, we investigate which of these indices works best for our data.

Concept stability is a measure which helps discriminating potentially interesting patterns from irrelevant information in a concept lattice based on possibly noisy data. The stability of a concept $C = (V, F)$ is the proportion of subsets of the extent V which have the same attribute set F as V:

$$\sigma((V,F)) = \frac{|\{A \subseteq V \mid A' = F\}|}{2^{|V|}}.^5 \tag{1}$$

Intuitively, a more stable concept is less dependant on any individual object in its extent and is therefore more resistant to outliers or other noisy data items.

Concept separation indicates the significance of the difference between the objects covered by a given concept from other objects and, simultaneously, between its attributes and other attributes:

$$\mathfrak{s}((V,F)) = \frac{|V|\,|F|}{\sum_{v\in V}|\{v\}'| + \sum_{f\in F}|\{f\}'| - |V|\,|F|}. \tag{2}$$

Intuitively we expect a concept with high separation index to better sort out the verbs it covers from other verbs and simultaneously the frames it covers from other frames. Whereas concept stability is a measure concerned with either objects or attributes, separation gives information about objects and attributes at the same time.

Concept probability. For an attribute $a \in A$, the attribute set, we denote by p_a the probability of an object to have the attribute a. In practise it is the proportion of objects having a: $p_a = \frac{|\{a\}'|}{|O|}$, where O denotes the set of objects.

For $B \subseteq A$, we define p_B as the probability of an arbitrary object having all attributes from B: $p_B = \prod_{a\in B} p_a$. This formulation assumes the mutual independence of attributes. Based on this, and denoting $n = |O|$ we obtain the following formula for the probability of B being closed:

$$p(B = B'') = \sum_{k=0}^{n} p(|B'| = k, B = B'') \tag{3}$$

$$= \sum_{k=0}^{n} \left[\binom{n}{k} p_B^k (1-p_B)^{n-k} \prod_{a\notin B} (1-p_a^k) \right] \tag{4}$$

[5] Here and in the following $'$ represents the operator on the power sets of objects: $' : 2^{\mathcal{O}} \to 2^{\mathcal{A}}, X' = \{a \in \mathcal{A} \mid \forall o \in X.\ (o,a) \in R\}$ and dually on that of attributes.

A small $p(B = B'')$ suggests a small probability of the attribute combination B to be a concept intent by chance only (and $p(B = B'') \approx 1$ that there is a high probability that the combination is a concept intent by chance). However, this reasoning is based on the independence of the attributes, which in our particular case can not be warranted.

4.1 Computing Stability, Separation and Probability Indices.

Stability. Calculating stability is known to be NP-complete (Kuznetsov, 2007), however Jay et al. (2008) show that when the concept lattice is known it can be computed efficiently by a bottom-up traversal algorithm introduced in (Roth et al., 2006). This is the algorithm we used to compute concept stability.
Separation can be computed in $\mathcal{O}(|O|+|A|)$ time, where O and A are the object and attribute sets respectively. Computing separation is the least prohibitive of the three indices.
Probability. Klimushkin et al. (2010) show that computing probability of only one concept involves $\mathcal{O}(|O|^2 \cdot |A|)$ multiplication operations which is computationally very costly. With the computational means at our disposal it was not possible for us to compute the concept probabilities. We therefore computed approximations derived as follows:
First, we consider $\prod_{a \in B}(1 - p_a^k) \approx 1$ for $k > 40$. In view of this, Equation (4) becomes:

$$p(B = B'') = \sum_{k=0}^{40} \left[\binom{n}{k} p_B^k (1 - p_B)^{n-k} \prod_{a \notin B} (1 - p_a^k) \right] \tag{5}$$

$$+ \sum_{k=41}^{n} \left[\binom{n}{k} p_B^k (1 - p_B)^{n-k} \right] \tag{6}$$

As $\sum_{k=0}^{n} \binom{n}{k} p^k (1-p)^{n-k} = 1$, Term (6) can be rewritten as:

$$1 - \sum_{k=0}^{40} \left[\binom{n}{k} p_B^k (1 - p_B)^{n-k} \right] = \tag{7}$$

$$1 - F(40; n, p_B). \tag{8}$$

$F(k; n, p) = \sum_{i=0}^{k} \binom{n}{i} p^i (1-p)^{n-1}$ is the cumulative distribution function of the binomial distribution[6] and can be computed using various statistical software packages. Term (5) can also be computed more easily considering that $\binom{n}{k} p_B^k (1 - p_B)^{n-k}$ are binomial densities the computation of which is also provided by statistics software[7]

[6] Source Wikipedia: `http://en.wikipedia.org/wiki/Binomial_distribution`
[7] We used the R software environment for statistical computing (`http://www.r-project.org/`).

4.2 Evaluating the Concept Selection Indices

In the following we measure the performance of the three concept selection indices with respect to our data. The experimental setting is as follows:

We first select a number of N (1500) concepts with best selection index. The selected concepts are aligned with the classes translated from VerbNet (see Section 5): For each translated class, we select the concept with best precision/recall f-measure. Then we associate to the concept with best f-measure the thematic roles of the translated VN class. Next we compare the obtained ⟨verb, thematic role set⟩ associations with those given by a reference. As for our task recall is more important than precision, we use the $F2$ measure, which gives more weight to recall, for comparison.

As reference we use the data used for training the classifier for learning the translated VN classes (see Section 5): we are checking which index selects the most relevant concepts, that is those best matching the translated classes. The reference consists of the ⟨verb, semantic role set⟩ pairs marked as positive examples in the training set, ie. those for which we considered that the French verbs could have the semantic roles given by the English VN class. Table 2 shows

	cov.	**prec.**	**rec.**	**F2**
stab only	39.88	18.96	32.55	26.27
sep only	34.25	28.37	21.52	23.41
prob only	35.53	26.60	20.73	22.38
w/o filtering	100	12.30	60.96	26.30

Table 2: F2 scores and coverage for stability, separation and the 6th probability 10-quantile.

the F2 scores and coverage when using only one index at a time. For stability and separation we applied the method above on the top ranking 1500 concepts. Regarding probability, at first sight, we should consider best the concepts with lowest probability – because the probability of their intents of being closed by chance only is accordingly low. However, looking at the data we found that these concepts have very few verbs and large intent (frame) sets - which rather suggest improbable or rare verb groups. On the other hand, the interpretation of concept probability suggests that a concept with a probability close to 1 could occur by chance only. For these reasons, to assess probability separately we settled on the 6th 10 quantile. The results confirm the observations of Klimushkin et al. (2010): stability alone gives F2 scores close to an upper bound – the results obtained without filtering, ie. aligning the translated classes with all the concepts of the lattice. The results for separation and probability are several points lower.

As we only select $\sim 10\%$ of the total number of concepts we also have to make sure that the selected concepts cover at least a reasonable amount of verbs. The **cov** column gives the percentage of verbs in the lattice covered by the selected concepts. It shows that using only one index at a time the pre-selected concepts would contain only $35\% - 40\%$ of the verbs in the entire lattice, which is unsatisfactory.

Klimushkin et al. (2010) investigate the performance of the stability, separation and probability indices at finding the original concepts in lattices produced from contexts which were previously altered by introducing two types of noise: *Type I noise* is obtained by altering every cell in the context with some probability, *Type II noise* is obtained by adding a given number or proportion of random objects or attributes. According to this, our contexts are affected by Type I noise rather than Type II. Klimushkin et al. (2010) found that stability was most effective at sorting out Type II noise, but also proved helpful in the case of Type I noise. In contrast, they suggest that separation and probability can not be used on their own but should rather serve as a normalising measure for stability. The most promising combination seemed to be: $\text{stability} + k_{sep} \cdot \text{separation} - k_{prob} \cdot \text{probability}$.

In the following we start from the assumption that the most effective index for selecting relevant concepts is given by a linear combination of stability, separation and probability: $k_{stab} \cdot \text{stability} + k_{sep} \cdot \text{separation} - k_{prob} \cdot \text{probability}$, and empirically determine the coefficients k_{stab}, k_{sep} and k_{prob} such that the selected concepts perform best with respect to our task.

We proceed as follows: We choose k_{stab}, k_{sep} and k_{prob}. We then compute the corresponding linear combination for the concepts and select the 1500 concepts ranking highest. As in the previous experiments, we measure the relevance of the selected concepts by aligning the concepts with the translated VN classes and by comparing the alignments with the same reference as before. We consider the "best" $k_{stab}, k_{sep}, k_{prob}$ combination the one giving highest F2 scores and good coverage.

Table 3a shows the results for a first series of experiments where k_{stab} and k_{sep} were assigned the values 0.5 and 1 and k_{prob} 0.25 and 0.5 (The lines are sorted by decreasing F2 score). They suggest that the stability and separation coefficients had less impact on coverage and F2 score than the probability coefficient. Interestingly the coverage is correlated with the F2 score.

In the second series of experiments, shown in Table 3b, we kept the stability and separation coefficients fixed and varied only the probability coefficient. These results suggest that the probability coefficient may not help at selecting the most relevant concepts in our setting. This may be due first to the fact that our attributes are not independent (we assumed independence of attributes when setting up the formula for computing the probability index) and second to the fact that we had to approximate the probability index and this approximation may not be accurate enough.

In the next series of experiments we investigated the impact of the number of preselected concepts (500). The results showed that with this smaller number of concepts the selected concepts reached a slightly smaller F2 score but a substantially lower coverage. Also, in this configuration the probability index did seem to be helpful. Preselecting 1000 concepts confirmed the previously observed tendencies: The F2 score and coverage were only slightly lower than when preselecting 1500 concepts and again the probability index seemed to have only a small impact on the overall results.

(a) F2 and coverage when $k_{stab}, k_{sep} \in \{0.5, 1\}, k_{prob} \in \{0.25, 0.5\}$.

k_{stab}	k_{sep}	k_{prob}	**cov.**	**prec.**	**rec.**	**F2**
1	1	0.25	98.04	11.87	55.19	24.89
1	0.5	0.25	98.04	11.87	55.19	24.89
1	0.5	0.5	57.69	17.08	30.18	24.04
1	1	0.5	56.15	17.45	29.13	23.82
0.5	0.5	0.25	56.15	17.45	29.13	23.82
0.5	1	0.25	53.81	18.03	27.82	23.36
0.5	0.5	0.5	49.72	18.55	26.25	23.06
0.5	1	0.5	49.90	18.61	25.98	22.95

(b) F2 and coverage when k_{stab} and k_{sep} are kept fixed and k_{prob} varies.

k_{stab}	k_{sep}	k_{prob}	**cov.**	**prec.**	**rec.**	**F2**
1	1	0	98.04	12.05	55.12	25.16
1	1	0.05	98.04	12.05	55.12	25.16
1	1	0.005	98.04	12.05	55.12	25.16
1	1	0.0005	98.04	12.05	55.12	25.16
1	1	0.1	98.00	11.91	55.38	25.00
1	1	0.2	98.08	11.88	55.12	24.91
1	1	0.25	98.04	11.87	55.12	24.89
1	1	0.3	98.00	11.79	55.38	24.80
1	1	0.4	59.95	16.27	31.23	23.91
1	1	0.5	56.16	17.45	29.13	23.82
w/o filtering			100	12.30	60.96	26.30

Table 3: F2 scores and coverage for various $k_{stab}, k_{sep}, k_{prob}$ combinations.

From these experiments we conclude the following: First they suggest that the best linear combination is the sum of the stability and separation indices as the F2 measure and the coverage for this combination are similar to those of an upper bound, ie. the alignment obtained without filtering. They show that selecting only $\sim 10\%$ of the original lattice gives a verb, frame, semantic role set alignment which is close to the alignment obtained when using the entire lattice and that the pre-selected concepts also have a similar coverage.

Second, it does not seem evident that probability has a positive effect on the selected concepts. However, it does improve f-measure when the number of selected concepts is lower (500 or 1000 vs. 1500 in our experiments). Hence, for our application we concluded that it is a better strategy to select a larger number of concepts (1500) and not take probability into account. This is even more so as the probability index in our case should be taken with caution because first we had to use an approximation to compute it which may be too rough, and second the computation of probability is based on the independence of attributes which is not warranted in our case.

5 Associating French Verbs with Thematic Role Sets.

We associate French verbs with thematic role sets by translating the English VerbNet classes to French using 3 English-French dictionaries. In the following we first briefly describe the relevant resources, ie. VerbNet and the dictionaries before giving the translation methodology. As for this paper only the translated classes, but not the method to produce them is relevant[8] we only very briefly sketch the methodology.

[8] Of course better translated classes will result in a better performance of our method, but it is not straight forward to evaluate the quality of the translated classes.

VerbNet (Schuler (2006)) is the largest electronic verb classification for English. It was created manually and classifies 3626 verbs using 411 classes. Each VN class includes among other things a set of verbs, a set of subcategorisation frames and a set of thematic roles. Figure 2 shows an excerpt of the *amuse-31.1* class, with its verbs, thematic roles and subcategorisation frames.

verbs (242): abash, affect, afflict, amuse, annoy, ...
thematic roles: EXPERIENCER, CAUSE
frames (6):

NP V NP	EXPERIENCER V CAUSE
NP V ADV-Middle	EXPERIENCER V Adv
NP V NP-PRO-ARB	CAUSE V
...	

Fig. 2: Simplified VerbNet class *amuse-31.1*.

English-French dictionaries. We use the following resources to translate the verbs in the English VN classes to French: Sci-Fran-Euradic, a French-English bilingual dictionary, built and improved by linguists , Google dictionary[9] and Dicovalence van den Eynde and Mertens (2003)[10]. The merged dictionary contains 51242 French-English verb pairs.

In the following we describe our method for translating the English VerbNet classes to French.

The translation of VerbNet classes is bound to be very noisy because verbs are polysemous and the dictionaries typically give translations for several readings of the verb: Thus the dictionary may give several translations v_{fr} which do not correspond to the meaning given by the $\langle v_{en}, class\rangle$ pair or this meaning may even not be covered at all by the dictionary. To get more accurate translated VN classes we use a machine learning method, namely Support Vector Machines (SVM)[11]. We follow a straight forward SVM application scenario: we build all the French verb, VN class pairs $\langle v_{fr}, C_{VN}\rangle$ where v_{fr} is a translation of an English verb in C_{VN}. The classifier has to give a probability estimate about whether this association is correct or not.

For training the classifier we use the 160 verbs appearing in the gold standard proposed by Sun et al. (2010)[12]. We build the pairs $\langle v_{fr}, C_{VN}\rangle$ where v_{fr} is a verb in the gold standard which is a translation of a verb in C_{VN}. For each of these pairs we assessed whether or not there was a meaning of v_{fr} where the semantic roles involved in the event described by the verb were those given by C_{VN}. The features associated to the ⟨verb, class⟩ pairs are numeric and are extracted from the dictionaries and VerbNet.

[9] `http://www.google.com/dictionary`. We obtained 13824 French-English verb pairs.
[10] The number of French-English verb pairs we obtained is 11351
[11] We used *libsvm*, the software package and methodology presented on `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`, Chang and Lin (2011).
[12] In fact this is the only existing gold standard for French VerbNet style classes and we also use it for the overall evaluation of our system (not presented in this paper).

The trained classifier is then used to produce probability estimates for all verb, class instances. We select the 6000 pairs with highest probability estimates[13] and finally obtain the translated classes by assigning each verb in a selected pair to the corresponding class.

To give an idea of the quality of the obtained classes: The accuracy of the classifier on the held out test set was 90%, compared to a maximum accuracy of 93.84% for five fold cross-validation on the development set. The frequency distribution of the translated classes obtained this way is much closer to the distribution of verbs in VerbNet classes as when using an approach based only on translation frequencies, thus providing more accurate verb groups to guide the FCA concept - thematic roles associations.

6 The French Verb ↔ Thematic Role Sets ↔ Syntactic Frame Associations

As a detailed and thorough evaluation of the verb, thematic role sets and syntactic frames associations would be out of the scope of this paper we only give here an intuition of the type of information provided by our method. Following the preliminary investigations in the previous sections we associated French verbs with subcategorisation frames and thematic role sets according to the scheme listed below:

- We group the VerbNet thematic roles and assign to one class all the VN verbs whose class have the same role set. We then translate the obtained classes using the methods described in Section 5.
- We use FCA to group French verbs and syntactic frames associated to these verbs by the lexicons described in Section 3. The concept lattices we create are based on the formal contexts consisting of French verbs as objects and SCFs as attributes.
- We then select the 1500 concepts where the sum of the stability and separation indices is highest because in Section 4 we found this combination of concept selection indices to work best for our application.
- For each translated VN class we identify among the 1500 filtered FCA concepts the one(s) with best f-measure between precision and recall.

The translated VerbNet class is then associated with this FCA concept(s). Thus the verbs in the FCA concept are effectively associated with the thematic role set of the translated class and at the same time with the syntactic frames in the intent (attribute set) of the FCA concept. Figure 3 shows the associations between concepts, thematic role sets and frames generated by our method for some VN classes[14]. The figure shows the concepts associated to these thematic role sets and for each of these concepts: their attribute set (syntactic frames),

[13] In VerbNet there are 5726 verb, class pairs

[14] These are the classes occuring in the gold standard proposed by Sun et al. (2010), mentioned in Section 5.

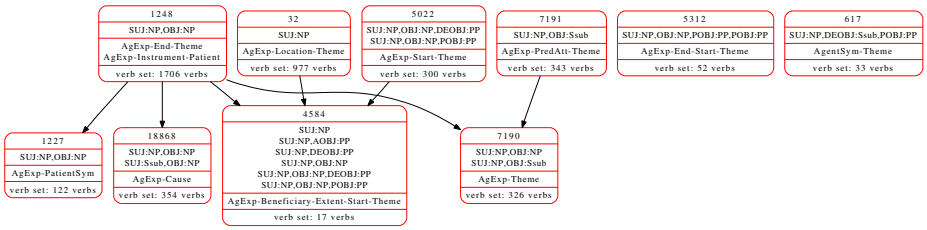


Fig. 3: French verb ↔ synt. frames ↔ thematic role set associations.

the associated thematic role set(s), the number of verbs in the concept and the hierarchical relations between the concepts as given by the concept lattice.

Thus for example the following 11 verbs (occuring in the gold standard) *bouger, déplacer, emporter, passer, promener, envoyer, expédier, jeter, porter, transmettre, transporter* are in concept 5312 and thereby may be used in the construction SUJ:NP,OBJ:NP,POBJ:PP,POBJ:PP[15] (according to our lexical resources). When they occur in this construction they are associated with the thematic role set *AgExp, End, Start, Theme*, i.e. the semantic roles involved are an AGENT or EXPERIENCER, a START point, an END point and a THEME. The listed verbs are all verbs of movement where an agent may move a theme from a start point to an end point – therefore in this case the associations with the syntactic frame and thematic role set seem to be correct. An NLP system which encounters the verb *déplacer* for example, used in the construction SUJ:NP,OBJ:NP,POBJ:PP,POBJ:PP could infer that possible thematic roles involved in the described event are an AGENT (or EXPERIENCER), a THEME, an END point and a START point. However, it still would not know which thematic role is realised by which syntactic argument.

There are also some problems with these associations. As can be seen in Figure 3, there is one case where the classification maps the same concept to two distinct VerbNet classes (*AgExp-End-Theme* and *AgExp-Instrument-Patient*). In addition, verbs in sub-concepts inherit the class label of the super-concept. Although there are verbs which belong to several VN classes, in many cases this multiple mapping was not warranted. Improving the precision of these mappings requires further investigations.

7 Conclusion

We introduced a new approach to verb clustering which involves the combined use of the English VerbNet, a bilingual English-French lexicon and a merged subcategorisation lexicon for French. Using these resources, we built two classifications, one derived from the English VN by translation and the other, from the subcategorisation lexicons via the construction of a formal concept lattice. We then use the translated VN to associate FCA concepts with VN classes

[15] a transitive construction with two additional prepositional objects

and thereby associate verbs with both syntactic frames and a thematic role set. We explored the performance of the concept selection indices introduced by Klimushkin et al. (2010) which are *stability*, *separation* and *probability* at selecting most relevant concepts with respect to our data and found that the sum of stability and separation gave best results in the setting of our application. These results were similar to those obtained without filtering, showing that this combination of the indices did indeed allow to select the most relevant concepts with respect to our data. Finally we showed the French verb, syntactic constructions and semantic role sets associations we obtained and briefly illustrated their potential use. Thus Formal Concept Analysis in combination with the concept selection indices, translation and set mapping methods proved an adequate method in this knowledge acquisition process.

Bibliography

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The berkeley FrameNet project. In *Proceedings of the 17th International Conference on Computational Linguistics*, volume 1, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Briscoe, T. and Carroll, J. (1993). Generalized probabilistic lr parsing of natural language (corpora) with unification-based grammars. *Comput. Linguist.*, 19(1):25–59.

Carroll, J. and Fang, A. C. (2004). The automatic acquisition of verb subcategorisations and their impact on the performance of an hpsg parser. In *IJCNLP*, pages 646–654.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Cimiano, P., S.Staab, and Tane, J. (2003). Automatic Acquisition of Taxonomies from Text: FCA meets NLP. In *Proceedings of the PKDD/ECML'03 International Workshop on Adaptive Text Extraction and Mining (ATEM)*, pages 10–17.

Dubois, J. and Dubois-Charlier, F. (1997). *Les verbes français*. Larousse.

Falk, I. and Gardent, C. (2010). Bootstrapping a Classification of French Verbs Using Formal Concept Analysis. In *Interdisciplinary Workshop on Verbs Interdisciplinary Workshop on Verbs*, page 6, Pisa Italy.

Falk, I., Gardent, C., and Lorenzo, A. (2010). Using Formal Concept Analysis to Acquire Knowledge about Verbs. In *Concept Lattices and their applications*, page 12, Sevilla, Spain.

Gross, M. (1975). *Méthodes en syntaxe*. Hermann, Paris.

Guillet, A. and Leclère, C. (1992). *La structure des phrases simples en français. 2 : Constructions transitives locatives*. Droz, Geneva.

Hadouche, F. and Lapalme, G. (2010). Une version électronique du LVF comparée avec d'autres ressources lexicales. *Langages*, pages 193–220. Mise en page différente que celle parue dans la revue.

Jay, N., Kohler, F., and Napoli, A. (2008). Analysis of social communities with iceberg and stability-based concept lattices. In *ICFCA'08: Proceedings of the 6th international conference on Formal concept analysis*, pages 258–272, Berlin, Heidelberg. Springer-Verlag.

Klimushkin, M., Obiedkov, S., and Roth, C. (2010). Approaches to the selection of relevant concepts in the case of noisy data. In Kwuida, L. and Sertkaya, B., editors, *Formal Concept Analysis*, volume 5986 of *Lecture Notes in Computer Science*, chapter 18, pages 255–266. Springer Berlin / Heidelberg, Berlin, Heidelberg.

Kupść, A. and Abeillé, A. (2008). Growing treelex. In Gelbkuh, A., editor, *Computational Linguistics and Intelligent Text Processing*, volume 4919 of *Lecture Notes in Computer Science*, pages 28–39. Springer Berlin / Heidelberg.

Kuznetsov, S. O. (2007). On stability of a formal concept. *Annals of Mathematics and Artificial Intelligence*, 49(1-4):101–115.

Palmer, M., Gildea, D., and Xue, N. (2010). *Semantic Role Labeling*. Synthesis lectures on human language technologies. Morgan & Claypool Publishers.

Palmer, M., Kingsbury, P., and Gildea, D. (2005). The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Priss, U. (2005). Linguistic Applications of Formal Concept Analysis. In Ganter, B., Stumme, G., and Wille, R., editors, *Formal Concept Analysis*, volume 3626 of *Lecture Notes in Computer Science*, pages 149–160–160. Springer Berlin / Heidelberg.

Roth, C., Obiedkov, S. A., and Kourie, D. G. (2006). Towards concise representation for taxonomies of epistemic communities. In *CLA*, pages 240–255.

Saint-Dizier, P. (1999). Alternation and verb semantic classes for french: Analysis and class formation. In *Predicative forms in natural language and in lexical knowledge bases*. Kluwer Academic Publishers.

Schuler, K. K. (2006). *VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon*. PhD thesis, University of Pennsylvania.

Sporleder, C. (2002). A Galois Lattice based Approach to Lexical Inheritance Hierarchy Learning. In *15th European Conference on Artificial Intelligence (ECAI'02): Workshop on Machine Learning and Natural Language Processing for Ontology Engineering, Lyon, France*.

Sun, L., Korhonen, A., Poibeau, T., and Messiant, C. (2010). Investigating the cross-linguistic potential of VerbNet-style classification. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 1056–1064, Stroudsburg, PA, USA. Association for Computational Linguistics.

Tolone, E. (2011). *Analyse syntaxique à l'aide des tables du Lexique-Grammaire du français*. PhD thesis, LIGM, Université Paris-Est, France, Laboratoire d'Informatique Gaspard-Monge, Université Paris-Est Marne-la-Vallée, France. (326 pp.).

van den Eynde, K. and Mertens, P. (2003). La valence : l'approche pronominale et son application au lexique verbal. *Journal of French Language Studies*, 13:63–104.

Generation algorithm of a concept lattice with limited object access

Ch. Demko♦★, K. Bertet♦

♦ L3I - Université de La Rochelle - av Michel Crépeau - 17042 La Rochelle
cdemko,kbertet@univ-lr.fr
★ Joomla! Production Leadership Team
christophe.demko@joomla.org

Abstract. Classical algorithms for generating the concept lattice $(C, \leq)$ of a binary table (O, I, R) have a complexity in $O(|C| * |I|^2 * |O|)$. Although the number of concepts is exponential in the size of the table in the worst case, the generation of a concept is output polynomial. In practice, the number of concepts is often polynomial in the size of the table. However, the cost of generating a concept remains high when the table is composed of a large number of objects.
We propose in this paper an algorithm for generating the lattice with limited object access, which can improve the computation time. Experiments were conducted with Joomla!, a content management system based on relational algebra, and located on a MySQL database.

keywords: concept lattice ; databases ; algorithm

1 Introduction

Galois lattices (or *concept lattices*) were first introduced in a formal way in the graph and ordered structures theory [1–3]. Later, they were developed in the field of Formal Concept Analysis (FCA) [4] for data analysis and classification. The concept lattice structure, based on the notion of *concept*, enables data description while preserving its diversity. It is used to analyse data when organised by a binary relation between objects and attributes.

Galois lattice is a graph providing a representation of all the possible correspondences between a set of *objects* (or examples) O and a set of *attributes* (or features) I. The technological improvements of the last decades enable use of these structures for data mining problems though they are exponential in space/time (worst case). It has to be noted that in practice, in most cases, the size of the lattice remains reasonable.

In addition, some applications offer to only generate some concepts from the huge amount of available data. Bordat's algorithm [5] is the more appropriate since it generates the cover relation between concepts, and thus allows an on-demand generation of concepts. Moreover, huge amount of data are often described by a huge amount of objects. It is the case in databases where sophisticated key-indexation techniques are used to improve object access.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

In this paper, we propose the Limited Object Access algorithm (LOA algorithm), an extension of Bordat's immediate successors generation with a limited access to objects. This algorithm, compounded with an on-demand strategy, and with sophisticated key-indexation techniques to improve objects's access, aims to improve time computation for a large amount of objects. However, worst case theoretical complexity remains the same as Bordat's algorithm. Experiments were conducted with Joomla!, a content management system based on relational algebra, and located on a MySQL database.

This paper is organized as follows. In section 2, we describe the Galois lattice structure and the Bordat's generation algorithm. In section 3, we describe our limited object access algorithm, illustrated by an example and some experiments.

2 Description and generation of a concept lattice

2.1 Description of a concept lattice

The *concept lattice* is a particular graph defined and generated from a relation R between objects O and attributes I. This graph is composed of a set of *concepts* ordered by a relation verifying the properties of a *lattice*, i.e. an order relation $\leq$ (transitive, reflexive and antisymmetric relation) such that, for each pair of concepts in the graph, there exists both a lower bound and an upper bound. Therefore, a lattice contains a minimum (resp. maximum) element according to the relation $\leq$ called the *bottom* (resp. *top*) of the lattice. The *Hasse diagram* of a graph [1] is the cover relation of $\leq$ denoted as $\prec$, i.e. the suppression on the graph of both transitivity and reflexivity edges.

We associate to a set of objects $A \subseteq O$ the set $f(A)$ of attributes in relation R with the objects of A:

$$f(A) = \{y \in I \mid xRy \ \forall \ x \in A\}$$

Dually, to a set of attributes $B \subseteq I$, we define the set $g(B)$ of objects in relation with the attributes of B:

$$g(B) = \{x \in O \mid xRy \ \forall \ y \in B\}$$

These two functions f and g defined between objects and attributes form a *Galois correspondence*. The relation between the set of objects and the set of attributes is described by a *formal context*. A formal context C is a triplet $C = (O, I, R)$ (or $C = (O, I, (f, g))$) represented by a table.

A *formal concept* represents maximal objects-attributes correspondences (following relation R) by a pair (A, B) with $A \subseteq O$ and $B \subseteq I$, which verifies $f(A) = B$ and $g(B) = A$. The whole set of formal concepts thus corresponds to all the possible maximal correspondences between a set of objects O and a set of attributes I.

Two formal concepts (A_1, B_1) and (A_2, B_2) are in relation in the lattice when they verify the following inclusion property:

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

The whole set of formal concepts fitted out by the order relation $\leq$ is called *concept lattice* or *Galois lattice* because it verifies the lattice properties: the relation $\leq$ is clearly an order relation, and for each pair of concepts (A_1, B_1) and (A_2, B_2), there exists the greatest lower bound (resp. the least upper bound) called *meet* (resp. *join*) denoted $(A_1, B_1) \wedge (A_2, B_2)$ (resp. $(A_1, B_1) \vee (A_2, B_2)$) defined by:

$$(A_1, B_1) \wedge (A_2, B_2) = (g(B_1 \cap B_2), (B_1 \cap B_2)) \quad (1)$$

$$(A_1, B_1) \vee (A_2, B_2) = ((A_1 \cap A_2), f(A_1 \cap A_2)) \quad (2)$$

The concepts $\perp = (O, f(O))$ and $\top = (g(I), I)$ respectively correspond to the bottom and the top of the concept lattice.

In *formal concept analysis* (FCA) concept lattices are used to analyse data when organised by a binary relation between objects and attributes. See the book of Ganter and Wille [4] for a more complete description of formal concept analysis.

In the following, we abuse notation and use $X + x$ (respectively, $X \setminus x$) for $X \cup \{x\}$ (respectively, $X \backslash \{x\}$).

2.2 Generation algorithms of a concept lattice

Numerous generation algorithms for concept lattices have been proposed in literature [6,7,5,8]. Although all these algorithms generate the same lattice, they propose different strategies. Some of these algorithms are incremental [6,9]. Ganter's NextClosure [7] is the reference algorithm that determines the concepts in lectical order (next, the concepts may be ordered by $\leq$ to form the concept lattice) while Bordat's algorithm [5] is the first algorithm that computes directly the Hasse diagram of the lattice. Recent work [10] proposed a generic algorithm unifying the existing algorithms in a unique framework, which makes easier the comparison of these algorithms. A formal and experimental comparative study of the different algorithms has been published [11].

All of these proposed algorithms have a polynomial complexity with respect to the number of concepts (at best quadratic in [8]). The complexity is therefore determined by the size of the lattice, this size being bounded by $2^{|O+I|}$ in the worst case and by $|O + I|$ in the best case. Studies on average complexity are difficult to carry out because the size of the concept lattice depends both on the dimensionality of the data to classify and on their organization and diversity. However, in practice the size of the Galois lattice generally remains reasonable.

Some applications offer to only generate some concepts from the huge amount of available data. Bordat's algorithm [5] is the more appropriate since it generates the cover relation between concepts, and thus allows an on-demand generation of concepts usually used in concrete applications. Bordat's algorithm is issued from a corollary of Bordat's theorem:

Theorem 1 (Bordat [5]). *Let (A, B) and (A', B') be two concepts of a context (O, I, R). Then $(A, B) \prec (A', B')$ if and only if A' is inclusion-maximal in the*

following set system $\mathcal{F}_A$ *defined on* O[1] *:*

$$\mathcal{F}_A = \{g(x + B) \ : \ x \in I \backslash B\} \tag{3}$$

Corollary 2 (Bordat [5]). *Let* (A, B) *be a concept. There is a one-to-one mapping between the immediate successors of* (A, B) *in the Hasse diagram of the lattice and the inclusion-maximal subsets of* $\mathcal{F}_A$*.*

Bordat's algorithm recursively computes all the concepts of $\mathbb{C}$ by computing immediate successors for each concept (A, B), starting from the bottom concept $\perp = (f(G), G)$, until all concepts are generated. Immediate successors are generated using Corollary 2 in $O(|I|^2 * |O|)$: the set system has first to be generated in a linear time ; then inclusion-maximal subsets of $\mathcal{F}_B$, can easily be computed in $O(|I|^2 * |O|)$.

3 Limited Object Access Algorithm (LOA)

3.1 Description of the LOA Algorithm

Large data are often described by a huge amount of objects, as in databases for example where the number of recordings (i.e. objects) can be huge, indexed using sophisticated key-indexation techniques. In this section, we describe our Limited Object Access algorithm (LOA algorithm), an extension of Bordat's immediate successors generation with a limited object access. This algorithm, compounded with an on-demand strategy aims to improve time computation for large amount of objects.

Our algorithm considers the restriction of a concept lattice to the attributes. A nice result establishes that any concept lattice $(\mathbb{C}, \leq_C)$ is isomorphic to the lattice $(\mathbb{C}_I, \subseteq)$ defined on the set I of attributes, with $\mathbb{C}_I$ the restriction of $\mathbb{C}$ to the attributes in each concept. The lattice $(\mathbb{C}_I, \subseteq)$ is also known as the closed sets lattice on the attributes I of a context (O, I, R), where the set system $\mathbb{C}_I$ is composed of all closed set - i.e. fixed points - for the closure operator $\varphi = g \circ f$. See the survey of Caspard and Monjardet [12] for more details about closed set lattices.

Using the closed sets lattice $(\mathbb{C}_I, \subseteq)$ instead of the whole concept lattice $(\mathbb{C}, \leq_C)$ gives raise to a storage improvement, for example in case of large amount of objects.

A closed sets lattice can be generated using an algorithm similar to Bordat's algorithm, and therefore enabling an on-demand generation in order to reduce the whole amount of closed sets. This algorithm (see Alg. 1) recursively computes immediate successors (see Alg. 2) of a closed set B, starting from the bottom closed set $\perp = \varphi(\emptyset)$, until I is generated.

The `Immediates_Successors_LOA` algorithm we propose (see Alg. 3) reinforces the object access limitation by considering the cardinality of the subset $g(X + B)$ instead of the subset itself to compute the inclusion-maximal subsets of $\mathcal{F}_A$ using the following property:

[1] In [5], the equivalent formulation $g(x) \cap A$ is used instead of $g(x + B)$

Proposition 3. *Consider a concept* (A, B)*, and two subsets* X *and* Y *of attributes in* $B \backslash I$*. Then*

$$g(X + B) \subseteq g(Y + B) \iff |g(X + B)| = |g(X + Y + B)| \tag{4}$$

This proposition is a direct consequence of the two following remarks:

1. The equivalence between inclusion and intersection set operations ($C \subseteq D \iff C = C \cap B$) allows to deduce the equivalence between $g(X + B) \subseteq g(Y + B)$ and $g(X + B) = g(X + B) \cap g(Y + B)$:
2. Then, by definition of g, we have $g(X + B) \cap g(Y + B) = g(X + Y + B)$.

More precisely, the `Immediates_Successors_LOA` algorithm (see Alg. 3) first initialize the set *Succ* of immediate successors of a closed set B with the emptyset. The set *Succ* is then updated by considering each attribute x of $I \backslash B$ and another already inserted potential successor $X \subseteq I \backslash B$ by considering the following four cases, where $c_B(Y)$ denotes the cardinality of $g(B + Y)$ for a Y of attributes:

Merge x with X: When $g(x + B) = g(X + B)$, then x and X belongs to the same closed set, and thus have to be merged in a same potential successor of B. By Proposition 3, this case is tested by $c_B(X + x) = c_B(X)$ and $c_B(X) = c_B(x)$.

Eliminate X: When $g(X + B) \subset g(x + B)$, then the closed set containing X isn't inclusion-maximal in $\mathcal{F}_A$, and thus hasn't to be considered as a potential successor of B. By Proposition 3, this case is tested by $c_B(X + x) = c_B(X)$ and $c_B(X) < c_B(x)$.

Eliminate x: When $g(x + B) \subset g(X + B)$, then the closed set containing x isn't inclusion-maximal if $\mathcal{F}_A$, and thus hasn't to be considered as a potential successor of B. By Proposition 3, this case is tested by $c_B(X + x) = c_B(X)$ and $c_B(x) < c_B(X)$.

Insert x: When x is neither eliminated or merged with X, then it is added as a potential successor of B ; another attribute is then considered.

3.2 Example

To illustrate our algorithm, we use the following context where numbers from 1 to 9 are described by some properties: the number is a prime number, an odd or even number, a square, a composite number or a factorial number.

	(p)rime	o(dd)	(e)ven	(s)quare	(c)omposite	(f)actorial
nb 1		×		×		×
nb 2	×		×			×
nb 3	×	×				
nb 4			×	×	×	
nb 5	×	×				
nb 6			×		×	×
nb 7	×	×				
nb 8			×		×	
nb 9		×		×	×	

Name: `Closed_Set_Lattice`
Data: A context $K = (O, I, R)$
Result: The Hasse diagram $(\mathbb{C}_I, \prec)$ of the lattice $(\mathbb{C}_I, \subseteq)$
begin
  $\mathbb{C}_I = \{f(O)\}$;
  foreach $B \in \mathbb{C}_I$ *not marked* **do**
    $Succ_B$=`Immediates_successors` (K, B);
    foreach $X \in Succ_B$ **do**
      $B' = B + X$;
      if $B' \notin \mathbb{C}_I$ **then** add B' to $\mathbb{C}_I$;
      add a cover relation $B \prec B'$
    end
    mark B
  end
  return $(\mathbb{C}_I, \prec)$
end

Algorithm 1: Generation of the Hasse diagram of the closed set lattice $(\mathbb{C}_I, \subseteq)$

Name: `Immediates_Successors`
Data: A context K ; 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
  initialize the set system $\mathcal{F}_A$ with $\emptyset$;
  foreach $x \in I \backslash B$ **do**
    add $g(x + B)$ to $\mathcal{F}_A$
  end
  $Succ$=maximal inclusion subsets of $\mathcal{F}_A$;
  return $Succ$
end

Algorithm 2: Generation of the immediate successors of a closed set in the Hasse diagram of the lattice $(\mathbb{C}_I, \subseteq)$

```
Name: Immediates_Successors_LOA
Data: A context K ; A closed set B of the closed set lattice (C_I, ⊆) of K
Result: The immediate successors of B in the lattice
begin
    initialize the Succ_B family to an empty set;
    foreach x ∈ I \ B do
        add = true;
        foreach X ∈ Succ_B do
            \\ 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
            \\ 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
            \\ 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
        \\ Insert x as a new potential successor ;
        if add then  add {x} to Succ_B
    end
    return Succ_B;
end
```

Algorithm 3: Generation of the immediate successors of a closed set in the Hasse diagram of the lattice $(\mathbb{C}_I, \subseteq)$

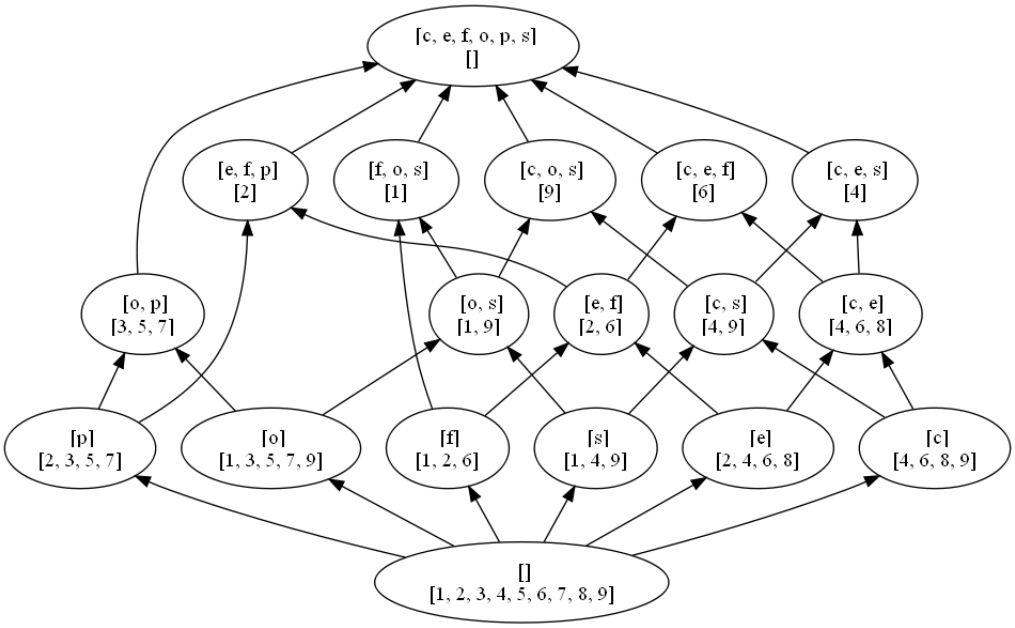


Fig. 1. Concept lattice

Figure 1 gives the concept lattice of this context. When the algorithm computes the successors of the closed sets e (resp. p), it proceeds as described in TABLE 1 (resp. TABLE 2). The different steps of these two examples show the different actions taken by the algorithm.

$Succ_F$	x	$c_B(x)$	X	$c_B(X)$	$c_B(x+X)$	Case	Action
$\emptyset$	p	1					Insert $[p]$
$\{[p]\}$	o	0	$[p]$	1	0	$c_B(x+X) = c_B(x) < c_B(X)$	Eliminate $[o]$
$\{[p]\}$	s	1	$[p]$	1	0	$c_B(x+X) < c_B(X) = c_B(x)$	
$\{[p]\}$	c	3	$[p]$	1	0	$c_B(x+X) < c_B(X) < c_B(x)$	Insert $[c]$
$\{[p],[c]\}$	f	2	$[p]$	1	1	$c_B(x+X) = c_B(X) < c_B(x)$	Eliminate $[p]$
$\{[c]\}$	f	2	$[c]$	3	1	$c_B(x+X) < c_B(x) < c_B(X)$	Insert $[f]$
$\{[c],[f]\}$							

Table 1. Immediate successors of $[e]$

3.3 Complexity

The complexity of computing the immediate successors of a closed set B using the `Immediates_Successors_LOA` algorithm is:

$$\frac{(|I| - |B|)(|I| - |B|)}{2} * O(c_B(X))$$

$Succ_F$	x	$c_B(x)$	X	$c_B(X)$	$c_B(x+X)$	Case	Action
$\emptyset$	o	3					Insert $[o]$
$\{[o]\}$	e	1	$[o]$	3	0	$c_B(x+X) < c_B(x) < c_B(X)$	Insert $[e]$
$\{[o],[e]\}$	s	0	$[o]$	3	0	$c_B(x+X) = c_B(x) < c_B(X)$	Eliminate $[s]$
$\{[o],[e]\}$	c	0	$[o]$	3	0	$c_B(x+X) = c_B(x) < c_B(X)$	Eliminate $[c]$
$\{[o],[e]\}$	f	1	$[o]$	3	0	$c_B(x+X) < c_B(x) < c_B(X)$	
$\{[o],[e]\}$	f	1	$[e]$	1	1	$c_B(x+X) = c_B(x) = c_B(X)$	Merge $[e]$, $[f]$
$\{[o],[ef]\}$							

Table 2. Immediate successors of $[p]$

which leads to

$$O((|I| - |B|)^2 * O(c_B(X)))$$

using the big O notation.

This has to be compared with $O(|I|^2 * |O|)$ of the `Immediates_Successors` algorithm. In addition the cost $O(C_B(x))$ of computing the cardinality of objects satisfying the required properties can be based on multiple keys and robust algorithms used in databases that do not need to load all data for computing a cardinality.

3.4 Experimentations

In the experiment, we use a dataset composed of:

54 attributes: the 6 attributes of the example, and attributes corresponding to the property to be multiple of $\{3 - 50\}$.
100000 objects: the integers between 1 and 100000

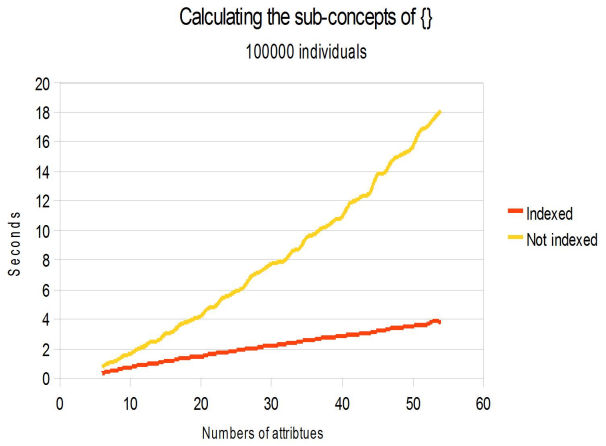
The dataset is stored in a database MySQL 5.5.14. We have implemented our `Immediates_Successors_LOA` algorithm using PhP 5.3.6. The counting of objects satisfying a set of properties is realised by the SQL request comparing indexes with a constant:

```
select count (*) from att1=1 and att2=1
```

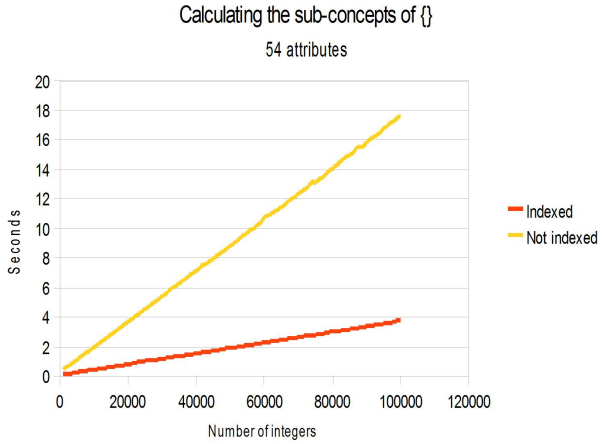
We compare the processing time of our `Immediates_Successors_LOA` algorithm in the two following cases:

Indexed: Each attribute is defined to be an index. Objects are indexed by their attributes, and MySQL can quickly retrieve them in the dataset using a B-tree indexation with a logarithmic complexity [13]: $O(c_B(X)) = O(log|I|)$.
Not indexed: Objects are not indexed and a scan of all the lines is necessary to retrieve objects. The complexity is then similar to those of the `Immediates_Successors` algorithm: $O(c_B(X)) = O(|I| - |B|)$.

We compare the processing time of computing the immediate successors of the botom element $\emptyset$ in this two cases (indexed and not indexed):



(a) 100000 first integers as objects ; a number of attributes between 6 to 54.



(b) 54 attributes ; integers between 1000 and 100000

Fig. 2. Calculating the immediate successors of $\emptyset$

Fig.2(a): for the 100000 first integers as objects, and a number of attributes between 6 to 54.

Fig.2(b): for the 54 attributes, and integer between 1000 and 100000.

In the results, the time processing is really improved with an indexed dataset, and seems to be near to linear in $O(|I| + |O|)$. Immediate successors of the $\emptyset$ for 100000 objects and 54 attributes are computed in 3 seconds with the indexed algorithm, and in 18 seconds with the not indexed one.

Moreover, the `explain` of the count operation shows that an `index-merge` operation is realized on indexes corresponding to an `intersect` computation:

```
mysql> explain select count(*) from numbers where p=1 and o=1;
+----+-------------+------+---------+------+----------------------+
| id | select_type | key  | key_len | rows | Extra                |
+----+-------------+------+---------+------+----------------------+
|  1 | index_merge | p,o  | 1,1     |    2 | Using intersect(p,o); |
|    |             |      |         |      | Using where;         |
|    |             |      |         |      | Using index          |
+----+-------------+------+---------+------+----------------------+
1 row in set (0.00 sec)
```

Therefore, optimizing the intersection operation, with an adaptated sort on the lines for example, would be a possible optimization of our algorithm.

4 Conclusion

In this paper, we described a new algorithm for computing the immediate successors of a concept using the counting of objects satisfying a set of properties. By separating the counting from the rest of the algorithm, new systems for exploring concept lattices can now rely on optimization algorithms used in relational databases. If the tests we will realize on PostgreSQL and MySQL databases are successfull in terms of manipulating a huge amounts of data, we plan to propose a library for extending content management system such as Joomla!.

References

1. Birkhoff, G.: Lattice theory. 3d edn. American Mathematical Society (1967)
2. Barbut, M., Monjardet, B.: Ordres et classifications : Algèbre et combinatoire. Hachette, Paris (1970) 2 tomes.
3. Davey, B., Priestley, H.: Introduction to lattices and orders. 2nd edn. Cambridge University Press (1991)
4. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical foundations. Springer Verlag, Berlin (1999)
5. Bordat, J.: Calcul pratique du treillis de Galois d'une correspondance. Math. Sci. Hum. **96** (1986) 31–47
6. Norris, E.: An algorithm for computing the maximal rectangles in a binary relation. Revue Roumaine de Mathématiques Pures et Appliquées **23** (1978)
7. Ganter, B.: Two basic algorithms in concept analysis. Technische Hochschule Darmstadt (Preprint 831) (1984)
8. Nourine, L., Raynaud, O.: A fast algorithm for building lattices. Information Processing Letters **71** (1999) 199–204
9. Gödin, R., Missaoui, R., Alaoui, H.: Incremental concept formation algorithms based on Galois (concept) lattices. Computational Intelligence **11** (1995) 246–267

10. Gely, A.: A generic algorithm for generating closed sets of binary relation. Third International Conference on Formal Concept Analysis (ICFCA 2005) (2005) 223–234
11. Kuznetsov, S., Obiedkov, S.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental and Theorical Artificial Intelligence **14** (2002) 189–216
12. Caspard, N., Monjardet, B.: The lattice of closure systems, closure operators and implicational systems on a finite set: a survey. Discrete Applied Mathematics **127** (2003) 241–269
13. Bayer, R. et McCreight, E.M.: Organization and maintenance of large ordered indexes. Acta Informatica **1** (1972) 173–189

Homogeneity and Stability in Conceptual Analysis

Paula Brito[1] and Géraldine Polaillon[2]

[1] Faculdade de Economia & LIAAD/INESC-Porto L.A., Universidade do Porto
Rua Dr. Roberto Frias, 4200-464 Porto, Portugal mpbrito@fep.up.pt
[2] SUPELEC Science des Systèmes (E3S) - Département Informatique
Plateau de Moulon, 3 rue Joliot Curie, 91192 Gif-sur-Yvette cedex, France
geraldine.polaillon@supelec.fr

Abstract. This work comes within the field of data analysis using Galois lattices. We consider ordinal, numerical single or interval data as well as data that consist on frequency/probability distributions on a finite set of categories. Data are represented and dealt with on a common framework, by defining a generalization operator that determines intents by intervals. In the case of distribution data, the obtained concepts are more homogeneous and more easily interpretable than those obtained by using the maximum and minimum operators previously proposed. The number of obtained concepts being often rather large, and to limit the influence of atypical elements, we propose to identify stable concepts using interval distances in a cross validation-like approach.

1 Introduction

This work concerns multivariate data analysis using Galois concept lattices. Let $E = \{\omega_1, \ldots, \omega_n\}$ be the set of elements to be analyzed, described by p variables $Y_1, \ldots, Y_p$. In this paper we consider the specific case where the variables Y_j are numerical (real or interval-valued), ordinal and modal. Modal variables allow associating with each element of E a probability/frequency distribution on an underlying finite set of categories (see [9]).

The use of Galois lattices in Data Analysis was first introduced by Barbut and Monjardet, in the seventies of last century [2] and then further developed and largely spread out by the work of R. Wille and B. Ganter (see, e.g., [6]). Let $(A, \leq_1)$ and $(B, \leq_2)$ be two ordered sets. A Galois connection is a pair (f, g), where f is a mapping $f : A \to B$, g is a mapping $g : B \to A$, such that f and g are antitone, and both $h = gof$ and $h' = fog$ are extensive; h and h' are then closure operators. The mapping f defines the intent of a set $S \subseteq E$, and the mapping g that allows obtaining the extent in E associated with a set of attributes $T \subseteq O$, where O is the set of the considered (binary) attributes. The couple (f, g) then constitutes a Galois connection between $(P(E), \subseteq)$ and $(P(O), \subseteq)$. A concept is defined as a couple (S, T) where $S \subseteq E, T \subseteq O, S = g(T)$ and $T = f(S)$, i.e., we have $h(S) = S$; S is the *extent* of the concept and T its *intent*. This approach has been applied to non-binary variables, but in this case data are generally submitted to a previous "binarization", by performing a binary coding of the

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

data array; for numerical or ordinal variables Y, attributes of the form "$Y \leq x$," for each observed value x, are considered.

In [3] this approach has been extended by defining directly the intent of a set of elements; which has allowed obtaining, for each variable type (classical or otherwise) appropriate couples of mappings (f, g) forming a Galois connection. This has the advantage of allowing analyzing the data directly as it is presented, without imposing any sort of binary pre-coding, which may, and generally does, drastically increase the size of the data array to be analyzed. Galois lattices where intents are obtained by union and by intersection are obtained. This approach has been further extended to modal variables (see [4]). The case of ordinal variables has been dealt with in [11], using an approach similar to that of [4] for modal variables.

Ganter and Kuznetsov [5] proposed a general construction, called pattern structures, which allows for arbitrary descriptions with a semilattice operation on them; since union and intersection of intervals define semilattices, they make respective pattern structures. An application on gene expression data is presented in [7].

Here, we consider a common framework for numerical (real or interval-valued), ordinal and modal variables, by defining a generalization operator that determines the intent in the form of vectors of intervals. For ordinal and modal (i.e., distribution-valued) variables the obtained concepts are more homogeneous and therefore easier to interpret than those obtained by applying the minimum and maximum operators, as previously proposed. In the next sections, we detail how generalization of a set of elements is performed for each variable type.

The number of obtained concepts being often rather large, we propose to identify stable concepts (see also [8] and [12]), using distances designed for interval data. The criteria is that the intent of a concept should not be too different from those obtained by sequentially removing one element of the extent at a time - which would reveal that this particular element is provoking a drastic change in the concepts' intent. Should it occur, the concept would be considered to be non-stable.

In the case of multi-valued data, other approaches of lattice reduction, directly applied to the concept lattice, have been proposed in [1] and [10]. These two approaches rely on the same idea of merging together similar attribute values (in respect to a given threshold), and thereby reducing the number of concepts.

The remainder of the paper is organized as follows. Section 2 describes the generalization procedure for real and interval-valued variables, which is extended in Section 3 to modal variables. In Section 4 a common generalization approach by vectors of intervals is presented. In Section 5 the problem of concept stability is considered, and a method using interval distances is proposed, which allows addressing the question of lattice reduction. Section 6 concludes the paper, opening paths for future research.

2 Real and interval-valued variables

Let $E = \{\omega_1, ..., \omega_n\}$ be the set of n elements or objects to be analyzed, and $Y_1, \ldots, Y_p$ real or interval-valued variables such that $Y_j(\omega_i) = [l_{ij}, u_{ij}]$. We shall consider real-valued variables as a special case of interval-valued ones; it is therefore equivalent to write $Y_j(\omega_i) = x$ or $Y_j(\omega_i) = [x, x]$.

Let $A = \{\omega_1, \ldots, \omega_h\} \subseteq E$. Generalization by union is defined (see [3]) by the mapping $f : P(E) \to I^p$ where I is the set of intervals of $\mathbb{R}$ endowed with the inclusion order, such that $f(A) = (I_1, \ldots, I_p)$, with $I_j = [Min\{l_{ij}\}, Max\{u_{ij}\}]$, $\omega_i \in A$, $j = 1, \ldots, p$, i.e., for each $j = 1, \ldots, p, I_j$ is the minimum interval (for the inclusion order) that covers all values taken by the elements of A for variable Y_j. Let $g : I^p \to P(E)$ be the mapping defined as $g((I_1, \ldots, I_p)) =$
$= \{\omega_i \in E : Y_j(\omega_i) \subseteq I_j, j = 1, \ldots, p\}$, i.e., the set of elements of E taking values within I_j, for $j = 1, \ldots, p$. The couple (f, g) is a Galois connection.

Likewise, we may generalise by intersection defining f and g as follows: $f^* : P(E) \to I^p$, $f(A) = (I_1, \ldots, I_p)$, with $I_j = [Max\{l_{ij}\}, Min\{u_{ij}\}]$ if $Max\{l_{ij}\} \leq Min\{u_{ij}\}, \omega_i \in A$, $I_j = \oslash$ otherwise (i.e., the largest interval contained in all intervals taken by the elements of A for variable Y_j, which may be empty), for $j = 1, \ldots, p$, and $g^* : I^p \to P(E)$ with $g^*((I_1, \ldots, I_p)) =$ $\{\omega_i \in E : Y_j(\omega_i) \supseteq I_j, j = 1, \ldots, p\}$ (the set of elements of E taking interval-values that contain I_j,) for $j = 1, \ldots, p$. The couple (f^*, g^*) forms also a Galois connection.

Example 1:
Consider three persons, Ann, Bob and Charles characterized by two variables, age and amount of time (in minutes) necessary to go to work (which varies from day to day, and is therefore represented by an interval-valued variable), as presented in Table 1.

	Age	Time
Ann	25	[15, 20]
Bob	32	[25, 30]
Charles	40	[10, 20]

Table 1. Age and amount of time (in minutes) necessary to go to work for three persons.

Let $A = \{$Bob,Charles$\}$. Generalization by the union leads to
$f(A) = ([32, 40], [10, 30])$, describing people who are between 32 and 40 years old and take 10 to 30 minutes to go to work; in this dataset people meeting this description are given by $g(f(A)) = g(([32, 40], [10, 30]))$, i.e., {Bob, Charles} = A. Here, ({Bob, Charles}, $([32, 40], [10, 30])$) is a concept.

3 Modal variables

Two Galois connections may also be defined for the case of modal variables (see [4]). Let $Y_1, \ldots, Y_p$ be p modal variables, $O_j = \{m_{j1}, \ldots, m_{jk_j}\}$ the set of k_j possible categories of variable Y_j, M_j the set of distributions defined on O_j, for $j = 1, \ldots, p$, and $M = M_1 \times \ldots \times M_p$. For variable Y_j and element $\omega_i \in E$, $Y_j(\omega_i) = \left\{m_{j1}(p_{j1}^{\omega_i}), \ldots, m_{jk_j}(p_{jk_j}^{\omega_i})\right\}$, where $p_{jk_\ell}^{\omega_i}$ is the probability/frequency associated with category $m_{j\ell}$ $(\ell = 1, \ldots, k_j)$ of variable Y_j, and element ω_i. Let $A = \{\omega_1, \ldots, \omega_h\} \subseteq E$.

To generalise by the maximum we take, for each category $m_{j\ell}$, the maximum of its probabilities/frequencies in A. Let $f : P(E) \rightarrow M$, such that $f(A) = (d_1, \ldots, d_p)$, with $d_j = \{m_{j1}(t_{j1}), \ldots, m_{jk_j}(t_{jk_j})\}$, where $t_{j\ell} = Max\{p_{j\ell}^{\omega_i}, \omega_i \in A\}, \ell = 1, \ldots, k_j$. The intent of a set $A \subseteq E$ is then to be interpreted as "objects with *at most* $t_{j\ell}$ cases presenting category $m_{j\ell}, \ell = 1, \ldots, k_j, j = 1, \ldots, p$". The couple (f, g) with $g : M \rightarrow P(E)$ defined as, for $d_j = \{m_{j1}(p_{j1}), \ldots, m_{jk_j}(p_{jk_j})\}$, $g((d_1, \ldots, d_p)) = \left\{\omega_i \in E : p_{j\ell}^{\omega_i} \leq p_{j\ell}, \ell = 1, \ldots, k_j, j = 1, \ldots, p\right\}$, forms a Galois connection.

Similarly, we may generalise by the minimum taking for each category the minimum of its probabilities/frequencies. Let $f^* : P(E) \rightarrow M$, $f^*(A) = (d_1, \ldots, d_p)$, with $d_j = \{m_{j1}(v_{j1}), \ldots, m_{jk_j}(v_{jk_j})\}$, where $v_{j\ell} = Min\{p_{j\ell}^{\omega_i}, \omega_i \in A\}, \ell = 1, \ldots, k_j$. The intent of a set $A \subseteq E$ is now interpreted as "objects with *at least* $v_{j\ell}$ cases presenting category $m_{j\ell}, \ell = 1, \ldots, k_j, j = 1, \ldots, p$".
The couple (f^*, g^*) with $g^* : M \rightarrow P(E)$ such that, for $d_j = \{m_{j1}(p_{j1}), \ldots, m_{jk_j}(p_{jk_j})\}$, $g^*((d_1, \ldots, d_p)) = \left\{\omega_i \in E : p_{j\ell}^{\omega_i} \geq p_{j\ell}, \ell = 1, \ldots, k_j, j = 1, \ldots, p\right\}$ forms likewise a Galois connection.

Example 2:
Consider four groups of students for each of which a categorical mark is given, according to the following scale: a: mark < 10, b: mark between 10 and 15, c: mark > 15 as summarized in Table 2.

	Mark
Group 1	$< 10(0.2), [10 - 15]\,(0.6), > 15(0.2)$
Group 2	$< 10(0.3), [10 - 15]\,(0.3), > 15(0.4)$
Group 3	$< 10(0.1), [10 - 15]\,(0.6), > 15(0.3)$
Group 4	$< 10(0.3), [10 - 15]\,(0.6), > 15(0.1)$
Group 5	$< 10(0.5), [10 - 15]\,(0.3), > 15(0.2)$

Table 2. Frequency distributions of the students marks, in 3 categories, for 5 groups.

The intent, obtained by the maximum operator, of the set formed by groups 1 and 2, is $\{a(0.3), b(0.6), c(0.4)\}$ and is interpreted as "students' groups with *at*

most 30% of marks a, *at most* 60% of marks b and *at most* 40% of marks c". The corresponding extent comprehends groups 1, 2, 3 and 4. If, alternatively, we determine the intent of the same set by the minimum operator, we obtain $\{a(0.2), b(0.3), c(0.2)\}$, to be read as "students' groups with *at least* 20% of marks a, *at least* 30% of marks b and *at least* 20% of marks c", whose extent is formed by groups 1, 2 and 5.

4 A common approach: generalization by intervals

We now present a unique framework allowing to perform generalization for numerical (real or interval-valued) variables, ordinal variables and modal variables, based on generalization by intervals.

For numerical (real or interval-valued) data, we are in the above mentioned case of generalization by taking the union.

For modal variables, it amounts to consider, for each category, an interval corresponding to the range of its probability/frequency. In fact, it has often been observed that generalization either by the maximum or by the minimum, as defined in Section 3, may quickly lead to over-generalization. As a consequence, $f(A)$, $A \subseteq E$, is not very informative.

Let $M_j^I = \{m_{j\ell}(I_{j\ell}), \ell = 1, \ldots, k_j\}, m_{j\ell} \in O_j, I_{j\ell} \subseteq [0,1]$ and $M^I = M_1^I \times \ldots \times M_p^I$. Generalization is now defined as

$$f^I : P(E) \to M^I$$
$$f^I(A) = (d_1, \ldots, d_p)$$
$$\text{with } d_j = \{m_{j1}(I_{j1}), \ldots, m_{jk_j}(I_{jk_j})\},$$

where $I_{j\ell} = \left[Min\{p_{j\ell}^{\omega_i}\}, Max\{p_{j\ell}^{\omega_i}\}\right], \omega_i \in A, \ell = 1, \ldots, k_j,\ j = 1, \ldots, p$ and

$$g^I : M^I \to E$$
$$g((d_1, \ldots, d_p)) = \left\{\omega_i \in E : p_{j\ell}^{\omega_i} \in I_{j\ell}, \ell = 1, \ldots, k_j, j = 1, \ldots, p\right\}$$

The so-defined couple of mappings (f^I, g^I) forms a new Galois connection.

On the data of Example 2, generalization by intervals of groups 1 and 2 provides the intent $\{a\,[0.2, 0.3]\,, b\,[0.3, 0.6]\,, c\,[0.2, 0.4]\}$, to be read as "students' groups having between 20% and 30% cases of mark a, between 30% and 60% cases of mark b and between 20% and 40% cases of mark c" and whose extent now only contains groups 1 and 2.

The case of ordinal variables has been addressed in [11], performing generalization either using the maximum or the minimum. To allow for more flexibility, the author proposes to choose the operator individually for each variable. Nevertheless, one of these generalization operators must be chosen in each case, and over-generalization is not prevented. Our proposal for this type of variables, is to generalise a set $A \subseteq E$ considering, no longer a minimum or a maximum, but rather an interval of ordinal values.

Example 3:
Consider the classifications given by four cinema critics while evaluating three movies, Movie 1, Movie 2 and Movie 3 as given in Table 3.

	Movie 1	Movie 2	Movie 3
Critic 1	5	5	4
Critic 2	5	4	4
Critic 3	1	2	2
Critic 4	2	1	1

Table 3. Classifications given by four critics to three movies.

The intent obtained by using the maximum operator of the group formed by critics 1 and 2 is $(5, 5, 4)$, to be interpreted as "critics giving *at most* mark 5 to Movie 1, *at most* mark 5 to Movie 2 and *at most* mark 4 to Movie 3" - which is obviously too general and would cover almost everyone; in this dataset the corresponding extent contains critics 1, 2, 3 and 4. Therefore, the class formed by critics 1 and 2, who present a similar behavior, does not correspond to a concept. The intent obtained by using the minimum operator of the group formed by critics 3 and 4 is $(1, 1, 1)$, to be read "critics giving *at least* mark 1 to Movie 1, *at least* mark 1 to Movie 2 and *at least* mark 1 to Movie 3" - which would cover every critic; its extent in this dataset consists again of critics 1, 2, 3 and 4. Here again, the class formed by critics 3 and 4, who give quite similar marks, does not correspond to a concept. If we now perform generalization by interval-vectors of the group formed by critics 1 and 2, we obtain the intent $([5, 5], [4, 5], [4, 4])$; likewise for the group formed by critics 3 and 4, we have $([1, 2], [1, 2], [1, 2])$; in the first case we are clearly referring to critics giving high marks while in the second case we describe critics giving low marks to all movies. The corresponding extents no longer contain other critics, presenting a rather different profile from those considered each time. Furthermore, both ({Critic 1, Critic 2}, $([5, 5], [4, 5], [4, 4])$ and ({Critic 3, Critic 4}, $([1, 2], [1, 2], [1, 2])$ are concepts. When determining concepts, according to the minimum or the maximum operators, e.g. in a clustering context, there is therefore a risk of forming heterogeneous clusters, since over-generalization may lead to a too large extent. By taking interval-vectors of observed values, the over-generalization problem is avoided.
To conclude this section, we now present a more general example, with variables of the different considered types.

Example 4:
Consider the data in Table 4, where 4 persons are described by their age, a real-valued variable, time (in minutes) they take to go to work, an interval-valued variable, the means of transportation used, a modal variable, and their classifications given to three newspapers, A, B and C (ordinal variable).

	Age	Time	Transport	A	B	C
Albert	25	[15, 20]	car (0.2) bus (0.8))	4	2	5
Bellinda	40	[25, 30]	car (0.7), bus (0.2), train (0.1))	2	4	3
Christine	32	[10, 15]	car (0.2), bus (0.7), train (0.1))	5	1	4
David	58	[30, 45]	car (0.9), bus (0.1))	2	4	1

Table 4. Age, time taken to go to work (in minutes), means of transportation used and classifications given to newspapers A, B and C for four persons.

The intent of $A = \{\text{Albert, Christine}\}$ is
$V = ([25, 32]\ ,\ [10, 20]\ ,\ ([0.2, 0.2], [0.7, 0.8], [0.0, 0.1])\ ,\ [4, 5], [1, 2], [4, 5])$ and (A, V) is a concept.

5 Stability

Concepts are theoretically very interesting, and do provide rich information on the values shared by subsets of elements of the set under study. However, the number of concepts of a data array is often rather large, even for relatively low cardinals of the sets of elements and variables. This fact makes the analysis and interpretation of results a bit delicate. It is often to be noticed that when analyzing the concepts generated by numerical or modal variables, groups of concepts appear which are quite similar. This may be due to noise or minor differences, generally not pertinent. The idea is therefore to extract only those concepts which are representative of these groups of similar concepts, so as to obtain a more concise representation with significantly homogeneous concepts.

Several solutions may be pointed out for this objective. We will focus on the notion of stability, as introduced in [8] and [12], which evaluates the amount of information of the intent that depends on specific objects of the concept's extent. Formally, the stability of a concept is defined as the probability of keeping its intent unchanged while deleting arbitrarily chosen objects of its extent.

When analyzing data described by numerical (real or interval-valued), ordinal or modal variables, and generalizing using interval-vectors (as described in the previous sections), we shall apply a similar approach to each formed concept, but introducing a distance measure. The objective being to retain the homogeneous concepts, it is wished to avoid that a single element of the concepts' extent produces an important increase in the intent's intervals' ranges.

To identify the stable concepts, a threshold α depending on the maximum distance is defined (so as no to be dependent from the variables' scales). A concept is said to be "stable" if the distance between the intent obtained by removing one element of the extent at a time, and its original intent, is not above the given threshold. This is in fact a cross-validation-like approach, in that one element of the extent is removed at a time, and the resulting intent is compared with the original one.

When data have an interval form, interval distances should be used. Different measures are available in the literature; we will focus on three interval distance measures: the *Hausdorff distance*, the *interval Euclidean distance* and the *interval City-Block distance*.

Let $I_i = [l_i, u_i]$ and $I_h = [l_h, u_h]$ be two intervals we wish to compare. The *Hausdorff distance* d_H, the *interval Euclidean distance* d_2 and the *interval City-Block distance* d_1 between I_i and I_h are respectively

$$d_H(I_i, I_h) = Max\left\{\left\{|l_i - l_h|, |u_i - u_h|\right\}\right.$$
$$d_2(I_i, I_h) = \sqrt{(l_i - l_h)^2 + (u_i - u_h)^2}$$
$$d_1(I_i, I_h) = |l_i - l_h| + |u_i - u_h|\,.$$

The *Hausdorff distance* between two sets is the maximum distance of a set to the nearest point in the other set, i.e., two sets are close in terms of the Hausdorff distance if every point of either set is close to some point of the other set. *Interval Euclidean* and *City-Block* distances are just the counterparts of the corresponding distances for real values; if we embed the interval set in $\mathbb{R}^2$, where one dimension is used for the lower and the other for the upper bound of the intervals, then these distances are just the *Euclidean* and *City-Block* distances between the corresponding points in the two-dimensional space.

Let $C = (A, D)$ be a concept, where $A = \{\omega_1, \ldots, \omega_h\} \subseteq E$ is its extent and $D = (I_1, \ldots, I_p)$ is its intent, $D = f(A)$. The considered criterion is then the distance Δ between D et D^{-i} where D^{-i} is the intent of A without ω_i, $D^{-i} = f(A \setminus \{\omega_i\}), i = 1, \ldots, h$, defined by: $\Delta = Max\{\delta(D, D^{-i}), \omega_i \in A\}$, δ measuring the dissimilarity between interval-vectors.

Let d be the distance (according to the chosen measure) between the intervals corresponding to variable Y_j in a concept's intent. Two options may then be foreseen, whether it is wished to consider the maximal or the average distance on the intervals defining the intents:

1. $\delta_{Max}(D, D^{-i}) = Max\{d(I_j, I_j^{-i})\}$, j indexing the variable set $Y_j, j = 1, \ldots, p$ in the case of numerical and ordinal variables, and the global category set $O = O_1 \cup \ldots \cup O_p$ in the case of p modal variables;
2. $\delta_{Mean}(D, D^{-i}) = Mean\{d(I_j, I_j^{-i})\}$, j as in 1.

A concept $C = (A, D)$ is then considered to be stable if $\Delta \leq \alpha$. This approach allows keeping only the stable, and therefore more representative, concepts, avoiding the effect of outlier observations.

6 Illustrative application

Consider again classifications given by cinema critics evaluating three movies, Movie 1, Movie 2 and Movie 3 where $Y_j(Critic_i)$ is the mark given by Critic i to Movie j, $i = 1, \ldots, 5; j = 1, 2, 3$, as given in Table 5.

Tables 6 and 7 list the concepts obtained when the Minimum and the Maximum generalization operators are used, respectively.

	Movie 1	Movie 2	Movie 3
Critic 1	3	2	3
Critic 2	1	1	2
Critic 3	5	5	1
Critic 4	4	3	2
Critic 5	2	4	5

Table 5. Classifications given by five critics to three movies.

	Intent		
Extent	Movie 1	Movie 2	Movie 3
$\{1\}$	≥ 3	≥ 2	≥ 3
$\{3\}$	≥ 5	≥ 5	≥ 1
$\{4\}$	≥ 4	≥ 3	≥ 2
$\{5\}$	≥ 2	≥ 4	≥ 5
$\{1,4\}$	≥ 3	≥ 2	≥ 2
$\{1,5\}$	≥ 2	≥ 2	≥ 3
$\{3,4\}$	≥ 4	≥ 3	≥ 1
$\{3,5\}$	≥ 2	≥ 4	≥ 1
$\{1,3,4\}$	≥ 3	≥ 2	≥ 1
$\{1,4,5\}$	≥ 2	≥ 2	≥ 2
$\{3,4,5\}$	≥ 2	≥ 3	≥ 1
$\{1,2,4,5\}$	≥ 1	≥ 1	≥ 2
$\{1,3,4,5\}$	≥ 2	≥ 2	≥ 1
$\{1,2,3,4,5\}$	≥ 1	≥ 1	≥ 1

Table 6. Concepts of the Minimum lattice corresponding to the data in Table 5.

	Intent		
Extent	Movie 1	Movie 2	Movie 3
$\{2\}$	≤ 1	≤ 1	≤ 2
$\{3\}$	≤ 5	≤ 5	≤ 1
$\{1,2\}$	≤ 3	≤ 2	≤ 3
$\{2,4\}$	≤ 4	≤ 3	≤ 2
$\{2,5\}$	≤ 2	≤ 4	≤ 5
$\{1,2,4\}$	≤ 4	≤ 3	≤ 3
$\{1,2,5\}$	≤ 3	≤ 4	≤ 5
$\{2,3,4\}$	≤ 5	≤ 5	≤ 2
$\{1,2,3,4\}$	≤ 5	≤ 5	≤ 3
$\{1,2,4,5\}$	≤ 4	≤ 4	≤ 5
$\{1,2,3,4,5\}$	≤ 5	≤ 5	≤ 5

Table 7. Concepts of the Maximum lattice corresponding to the data in Table 5.

The concepts (except for the empty extent one) obtained from this data table, using generalization by intervals, i.e., for $A \subseteq E, f(A) = (I_1, I_2, I_3)$, with $I_j = [Min\{Y_j(Critic_i)\}, Max\{Y_j(Critic_i)\}]$, $Critic_i \in A, j = 1, 2, 3$, are listed in Table 8.

	Intent		
Extent	Movie 1	Movie 2	Movie 3
$\{1\}$	$[3, 3]$	$[2, 2]$	$[3, 3]$
$\{2\}$	$[1, 1]$	$[1, 1]$	$[2, 2]$
$\{3\}$	$[5, 5]$	$[5, 5]$	$[1, 1]$
$\{4\}$	$[4, 4]$	$[3, 3]$	$[2, 2]$
$\{5\}$	$[2, 2]$	$[4, 4]$	$[5, 5]$
$\{1, 2\}$	$[1, 3]$	$[1, 2]$	$[2, 3]$
$\{1, 4\}$	$[3, 4]$	$[2, 3]$	$[2, 3]$
$\{1, 5\}$	$[2, 3]$	$[2, 4]$	$[3, 5]$
$\{2, 4\}$	$[1, 4]$	$[1, 3]$	$[2, 2]$
$\{2, 5\}$	$[1, 2]$	$[1, 4]$	$[2, 5]$
$\{3, 4\}$	$[4, 5]$	$[3, 5]$	$[1, 2]$
$\{3, 5\}$	$[2, 5]$	$[4, 5]$	$[1, 5]$
$\{4, 5\}$	$[2, 4]$	$[3, 4]$	$[2, 5]$
$\{1, 2, 4\}$	$[1, 4]$	$[1, 3]$	$[2, 3]$
$\{1, 2, 5\}$	$[1, 3]$	$[1, 3]$	$[2, 5]$
$\{1, 3, 4\}$	$[3, 5]$	$[2, 5]$	$[1, 3]$
$\{1, 4, 5\}$	$[2, 4]$	$[2, 4]$	$[2, 5]$
$\{2, 3, 4\}$	$[1, 5]$	$[1, 5]$	$[1, 2]$
$\{3, 4, 5\}$	$[2, 5]$	$[3, 5]$	$[1, 5]$
$\{1, 2, 3, 4\}$	$[1, 5]$	$[1, 5]$	$[1, 3]$
$\{1, 2, 4, 5\}$	$[1, 4]$	$[1, 4]$	$[2, 5]$
$\{1, 3, 4, 5\}$	$[2, 5]$	$[2, 5]$	$[1, 5]$
$\{1, 2, 3, 4, 5\}$	$[1, 5]$	$[1, 5]$	$[1, 5]$

Table 8. Concepts of the interval lattice for the data in Table 5.

We notice that all the concepts obtained using the Minimum or the Maximum operator are concepts for the interval generalization, although with a different meaning, given the different intent mapping. As discussed before, even in this small example it may be observed that concepts obtained using the Minimum or the Maximum operator often present a rather general intent, thus leading to over-generalization in the concept formation. Consider, for instance, the concept ($\{1\}$, (Movie 1 ≥ 3 , Movie 2 ≥ 2 , Movie 3 ≥ 3)) in Table 6, it indicates that Critic 1 gives high marks to each movie, which is not really the case, whereas the concept ($\{1\}$, (Movie 1 $\in [3, 3]$, Movie 2 $\in [2, 2]$, Movie 3 $\in [3, 3]$)) in Table 8 gives a much more accurate description of the concepts's extent. Also, concept ($\{3\}$, (Movie 1 ≤ 5 , Movie 2 ≤ 5 , Movie 3 ≤ 1)) in Table 7 describes Critic 3

as giving any marks to Movies 1 and 2, and low marks to Movie 3; using interval generalization we learn that the marks given by Critic 3 to Movies 1 and 2 are the highest and non other. Consider now concept ($\{3,4\}$, (Movie $1 \geq 4$, Movie $2 \geq 3$, Movie $3 \geq 1$)) in Table 6: the intent reports any mark for Movie 3 (in particular, high marks are possible); if we use interval generalization instead we obtain the concept ($\{3,4\}$, (Movie $1 \in [4,5]$, Movie $2 \in [3,5]$, Movie $3 \in [1,2]$ which more accurately describes the observed situation.

We now compare the concepts retained as stable with each of the three distances, using both δ_{Max} and δ_{Mean}, and a threshold value of 1 and 2. The identified stable concepts in each case, represented by the corresponding extent, are listed in Table 9.

Distance	Criterion	Threshold	Stable concepts (extent)
d_H	Max	1	$\{1\},\{2\},\{3\},\{4\},\{5\},\{1,4\}$
		2	$\{1\},\{2\},\{3\},\{4\},\{5\},\{1,2\},\{1,4\},\{1,5\},\{3,4\},$ $\{1,2,4\},\{1,2,5\},\{1,3,4\},\{1,4,5\},$ $\{1,2,3,4\},\{1,2,4,5\},\{1,3,4,5\},\{1,2,3,4,5\}$
	Mean	1	$\{1\},\{2\},\{3\},\{4\},\{5\},\{1,4\},$ $\{1,2,4\},\{1,2,5\},\{1,2,4,5\},\{1,3,4,5\},\{1,2,3,4,5\}$
		2	$\{1\},\{2\},\{3\},\{4\},\{5\},$ $\{1,2\},\{1,4\},\{1,5\},\{2,4\},\{3,4\},\{4,5\}$ $\{1,2,4\},\{1,2,5\},\{1,3,4\},\{1,4,5\},\{2,3,4\},\{3,4,5\}$ $\{1,2,3,4\},\{1,2,4,5\},\{1,3,4,5\},\{1,2,3,4,5\}$
d_2	Max	1	$\{1\},\{2\},\{3\},\{4\},\{5\},\{1,4\}$
		2	$\{1\},\{2\},\{3\},\{4\},\{5\},\{1,2\},\{1,4\},\{1,5\},\{3,4\},$ $\{1,2,4\},\{1,2,5\},\{1,3,4\},\{1,4,5\},$ $\{1,2,3,4\},\{1,2,4,5\},\{1,3,4,5\},\{1,2,3,4,5\}$
	Mean	1	$\{1\},\{2\},\{3\},\{4\},\{5\},\{1,4\},$ $\{1,2,4\},\{1,2,4,5\},\{1,3,4,5\},\{1,2,3,4,5\}$
		2	$\{1\},\{2\},\{3\},\{4\},\{5\},$ $\{1,2\},\{1,4\},\{1,5\},\{2,4\},\{3,4\},\{4,5\}$ $\{1,2,4\},\{1,2,5\},\{1,3,4\},\{1,4,5\},\{2,3,4\},\{3,4,5\},$ $\{1,2,3,4\},\{1,2,4,5\},\{1,3,4,5\},\{1,2,3,4,5\}$
d_1	Max	1	$\{1\},\{2\},\{3\},\{4\},\{5\},\{1,4\}$
		2	$\{1\},\{2\},\{3\},\{4\},\{5\},\{1,2\},\{1,4\},\{1,5\},\{3,4\},$ $\{1,2,4\},\{1,2,5\},\{1,3,4\},\{1,4,5\},$ $\{1,2,3,4\},\{1,2,4,5\},\{1,3,4,5\},\{1,2,3,4,5\}$
	Mean	1	$\{1\},\{2\},\{3\},\{4\},\{5\},\{1,4\},$ $\{1,2,4\},\{1,2,4,5\},\{1,3,4,5\},\{1,2,3,4,5\}$
		2	$\{1\},\{2\},\{3\},\{4\},\{5\},$ $\{1,2\},\{1,4\},\{1,5\},\{2,4\},\{3,4\},\{4,5\},$ $\{1,2,4\},\{1,2,5\},\{1,3,4\},\{1,4,5\},\{2,3,4\},\{3,4,5\},$ $\{1,2,3,4\},\{1,2,4,5\},\{1,3,4,5\},\{1,2,3,4,5\}$

Table 9. Stable concepts for different distances, criteria and threshold values.

As it may be seen from Table 9, for all distances and both criteria, a demanding threshold identifies a small number of stable concepts, therefore leading to an important reduction in the number of retained concepts; if we use a more liberal threshold, a larger number of concepts are retained as stable, as was to be expected. The maximum criterion is naturally more strict than the mean, which retains more concepts as stable, for all distances and both threshold values. Finally, in this example, no important difference appears between the results obtained for the different distance measures.

7 Conclusion

A common generalization procedure, for numerical, ordinal and modal variables, which uses a representation based on interval-vectors is presented. This allows defining more homogeneous concepts, than generalization operators that use the maximum and/or the minimum. The proposed approach for ordinal variables allows addressing recommendation systems, analyzing preference data tables. It would also be interesting to explore how the proposed generalization operator behaves in a supervised learning context.

The number of obtained concepts being often rather large, a method for identifying stable concepts is proposed, using a cross-validation-like approach. This allows avoiding the effect of atypical elements in the concepts' formation. Naturally, the value of the used threshold has an important influence in the rate of concept reduction. The next step will be to explore this methodology for larger data tables, so as to have a more accurate evaluation of its efficiency in concept reduction. Another issue interesting to investigate is the comparaison of the list of concepts with those obtained with a subset of the given variables. This then leads to the problem of variable selection in the context of Galois lattices construction and analysis. As concerns applications, we are particularly interested in analyzing real preference data, for application in recommendation systems.

References

[1] Z. Assaghir, M. Kaytoue, N. Messai and A. Napoli (2009). On the mining of numerical data with Formal Concept Analysis and similarity. In *Proc. Société Francophone de Classification*, pp. 121-124.

[2] Barbut, M. and B. Monjardet (1970). *Ordre et Classification, Algèbre et Combinatoire, Tomes I et II*. Paris: Hachette.

[3] Brito, P. (1994). Order structure of symbolic assertion objects. *IEEE Transactions on Knowledge and Data Engineering 6*(5), 830–835.

[4] Brito, P. and G. Polaillon (2005). Structuring probabilistic data by Galois lattices. *Math. & Sci. Hum. / Mathematics and Social Sciences 169*(1), 77–104.

[5] Ganter, B. and S.O. Kuznetsov (2001). Pattern structures and their projections. In: G. Stumme and H. Delugach (Eds.), *Proc. 9th Int. Conf. on Conceptual Structures, ICCS'01*, Lecture Notes in Artificial Intelligence, vol. 2120, pp. 129-142.

[6] Ganter, B. and R. Wille (1999). *Formal Concept Analysis, Mathematical Foundations*. Berlin: Springer.

[7] Kaytoue, M., S.O. Kuznetsov, A. Napoli and S. Duplessis (2011). Mining gene expression data with pattern structures in formal concept analysis. *Information Sciences*, Volume 181, Issue 10, 1989–2001.

[8] Kuznetsov, S. (2007). On stability of a formal concept. *Annals of Mathematics and Artificial Intelligence 49*(1-4), 101–115.

[9] Noirhomme-Fraiture, M. and P. Brito (2011). Far beyond the classical data models: Symbolic Data Analysis. *Statistical Analysis and Data Mining 4*(2), 157–170.

[10] Pernelle, N., M.-C. Rousset, and V. Ventos (2001). Automatic construction and refinement of a class hierarchy over multi-valued data. In L. De Raedt and A. Siebes (Eds.), *Principles of Data Mining and Knowledge Discovery*, Lecture Notes in Computer Science, pp. 386–398.

[11] Pfaltz, J. (2007). Representing numeric values in concept lattices. In J. Diatta, P. Eklund and M. Liquiere (Eds.), *Proc. Fifth International Conference on Concept Lattices and Their Applications*, pp. 260–269.

[12] Roth, C., S. Obiedkov and D. Kourie (2008). On succint representation of knowledge community taxonomies with Formal Concept Analysis. *International Journal of Foundations of Computer Science 19*(2), 383–404.

A lattice-based query system for assessing the quality of hydro-ecosystems

Agnès Braud[1], Cristina Nica[2], Corinne Grac[3], and Florence Le Ber[⋆3,4]

[1] LSIIT, CNRS-UdS, Strasbourg, France
[2] University Dunărea de Jos, Galaţi, Romania
[3] LHYGES, CNRS-ENGEES-UdS, Strasbourg, France
[4] LORIA – INRIA NGE, Nancy, France

Abstract. Concept lattices are useful tools for organising and querying data. In this paper we present an application of lattices for analysing and classifying stream sites described by physical, physico-chemical and biological parameters. Lattices are first used for building a hierarchy of site profiles which are annotated by hydro-ecologists. This hierarchy can then be queried to classify and assess new sites. The whole approach relies on an information system storing data about Alsatian stream sites and their parameters. A specific interface has been designed to manipulate the lattices and an incremental algorithm has been implemented to perform the query operations.

Keywords: incremental lattice, lattice-based query system, classification, information system, biological quality of water-bodies

1 Introduction

Concept -or Galois- lattices are useful tools for organising, mining, and querying qualitative data in various application domains [14, 10, 24]. However when developing a domain specific lattice-based tool -to be used by domain analysts, a main problem is to define the proper approach and tool that fit the requirements of the experts and other users involved in the project. This paper presents an application of Galois lattices to the hydro-ecological domain, focussing on how to assess and monitor the ecological state of streams or water areas. These questions are currently major problems in Europe, as underlined by the recent European Water Framework Directive (2000). Assessing the ecological quality of streams requires to take into account various data such as physico-chemical measures on sites, but also taxonomic statements or qualitative information on species. Furthermore tools are needed to summarise all these data and to provide a global and reliable information on the ecological state of streams and water areas. Following this aim we have developed an information system to collect data on Alsatian streams (North-East of France) [17] and implemented a lattice-based query system to help hydro-ecologists to compare and assess the ecological state

⋆ Corresponding author, florence.leber@engees.unistra.fr.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

of streams. Concepts lattices are used: (1) to organise data, i.e. stream or water area sites with similar parameters are clustered within concepts; (2) to embed expert knowledge, i.e. concepts are annotated with an expert qualification or comment; (3) to perform queries, i.e. the annotated concepts are used to help assessing new sites of streams or water areas.

The paper is organised as follows. First (Section 2) we present the application domain. Section 3 is devoted to the principles of lattice-based querying. Sections 4 and 5 describe the principles and the implementation of our proposition. Section 6 compares our approach to other lattice-based tools and the last section is a conclusion.

2 Assessing the quality of hydro-ecosystems

The European Water Framework Directive (2000) requires the development of new tools for monitoring and assessing the quality of water-bodies (i.e. rivers, lake, gravel pits,...). Such an assessment is built on various information: information about the species living in the streams and physical, chemical and biological data collected on the sites. From these information are built several numerical indices that are synthetic indicators for assessing the physico-chemical or biological quality of an hydro-ecosystem.

More precisely, in France, five biological indices have been normalised to assess the quality of running water. They are based on three faunistic groups: the invertebrate index [1], the oligochaete (small worms living in sediments) index [3], the fish index [5], and on two floristic groups: the diatom (microscopic algae) index [2], and the macrophyte (macroscopic plants living in water) index [4]. Illustrations of the taxa used for these indices are given in Figure 1.

(a) Invertebrate (b) Oligochaete (c) Fish (d) Diatom (e) Macrophyte

Fig. 1: Taxa examples for the five biological indices

According to AFNOR (French organism of normalisation) [1, 3, 5, 2, 4] each of them gives a different estimation of the water ecosystem quality. The macrophyte index estimates the trophic level of water, the diatom index gives the global water quality, the oligochaete index gives an evaluation of the sediment quality, and the fish index allows to classify the chemical and physical water quality quite like the invertebrate index. Therefore, their answers on a same site, with a same undergone pressure, at the same time can be really different but the simultaneous

application of these five indices is not common and work comparing their answers are not frequent [20].

Furthermore, indices based on physical (e.g. width and slope of the stream bed) and physico-chemical (e.g. pH, temperature, nitrates, organic matters, pesticides) data give an other estimation of the ecosystem quality.

Thus, it is necessary to combine the various indices to assess the quality of a whole water ecosystem. Such an approach, called the *ecological ambiance system*, has been proposed in [20, 21] based on the five French biological indices. Our objective is to develop this concept and to propose a concretely applicable tool. We therefore rely on a large database collecting data on Alsatian streams and water areas [18]. The database contains 38 tables and it suits the SANDRE[1] French national format for aquatic data. It is implemented within the MySQL Database Management System.

The data are either issued from samples, synthetic data or general information issued from the literature. They are qualitative and quantitative, and suit the current standards about protocol sampling and indices computation based on thresholds [1, 3, 5, 2, 4, 22, 23]. Data issued from samples correspond to raw data. Synthetic data are produced from these samples, in particular taxonomic lists are used to compute biological indices. Data issued from the literature are used for the analysis and synthesis of the preceding data (for example they provide the thresholds for the classification of physical, physico-chemical and biological results into classes ranging from 1 (very good quality) to 5 (very bad quality)). We have gathered information on 700 sites in the Alsace Plain, the oldest one being collected 20 years ago. Details on this database and how it is used are given in [17].

3 Using lattices for querying databases

Galois lattices are useful tools for organising data and building knowledge bases [7, 14, 24]. Furthermore, they are very interesting for information retrieval since they allow both direct retrieval and browsing [16]. Primarily, concept lattices have been used for information retrieval within texts [25, 11]. More recently lattice-based approaches have been used to build query or information retrieval systems on various data: e.g. information retrieval within photos or personal data [13], geographical data [8], or museum collections [26]. The underlying hypothesis is that a concept extent represents the result of a query which is defined by the conjunction of its intent. The query can be easily refined or enlarged following the edges starting from the concept into the lattice hierarchy.

Practically, the query (a A set of attributes) can be performed as follows: the lattice is looked for a matching concept that is a concept which intent equals the A set -if it exists- or the most general concept which intent is larger than A. This concept can also be characterised as the infimum (greatest lower bound) of all the concepts containing at least one of the attributes of A. This can be done

[1] `http://sandre.eaufrance.fr`

with various algorithms and the queried lattice does not have to be modified. Furthermore, a local view can be displayed to the user.

However, when the query represents a new object that is to be incorporated within the lattice, an incremental algorithm has to be used [15, 10]. This is the case in our application, since the user has got data about real stream sites which she/he wants to confront to the sites represented in the existing lattice. Furthermore, she/he can add the new sites to the lattice and thus modify its structure. We have implemented therefore two incremental algorithms proposed in [10], and roughly described in section 5.1. These algorithms have been chosen because they allow to build the Hasse diagram of the lattice, contrarily to most of incremental algorithms (see [19] for a comparison on these algorithms). Furthermore, we did not look for performance, since in this first step of our work only small data sets (40 sites) have been considered.

4 Using lattices for assessing hydro-ecosystems

Lattices have been used in two ways: firstly to cluster stream sites into concepts that are used by hydro-ecologists to define *profiles* of these sites; secondly, the lattices are annotated with the profiles and used into a query-system to help the assessment of new sites. The proposed tool includes the two stages (see Section 5.2).

4.1 A lattice-based clustering of Alsatian stream sites

Stream sites are described by different numerical attributes, biological indices on the one hand, physico-chemical data on the other hand. Those attributes are converted into ordinal scales leading to quality classes. The whole context contains about 40 stream sites, described with 5 biological indices, 10 physico-chemical indices and 5 physical indices. In the following, we focus on the biological indices. Table 1 gives the values of these five indices restricted to seven sites. Each site is denoted by a code: for example, the BW2 site (Brunnwasser downstream) has a good quality (class 2) for the IBGN (invertebrate), IBD (diatom) and IPR (fish) indices, a bad quality (class 4) for the IBMR (macrophyte) index and an average quality (class 3) for the IOBS (oligochaete) index. The multi-valued context represented in table 1, denoted $C7$ in the following, can be converted into a binary one by using a linear scale [14].

The general idea is to gather similar sites and to allocate them a profile describing their ecological state, combining the quality estimations of all compartments, with respect to the different classes of indices. This work is based on the approach described in [20]. The process is as follows:

- Step 1: Lattice construction on the data. To facilitate the expert analysis, the context size is reduced by focussing on a small number of indices or by identifying sub-lattices with respect to classes of indices. For example,

Site code	IBGN	IBMR	IOBS	IBD	IPR
BW2	2	4	3	2	2
IL1	3	3	3	2	3
MO1	1	4	3	3	4
MS2	2	4	5	2	2
RT2	2	5	4	2	2
ST1	1	3	4	3	2
ZN4	1	4	4	3	2

Table 1: Quality classes of the five biological indices for 7 stream sites

Figure 2 presents the lattice obtained from the context *C7* (the lattice was built with ConExp[2]) .

- Step 2: Analysis by the experts of the lattice hierarchy and its implication rules in order to select relevant concepts (or site profiles). In this step, the expert may identify profiles which are not present in the lattice and create virtual sites to be represented in the lattice.
- Step 3: Qualification of the concepts by the experts. For example, the concept ({IBGN 2, IBD 2, IPR 2, IBMR 4, IOBS 3},{BW2}) (down on the lattice, Figure 2) is interpreted as follows: *Brunnwasser downstream: low sediment degradation, high eutrophication, good general potential of resilience and possible resilience for sediments, various habitats.*

Once a suitable annotated lattice has been built following this process, it can be used to determine the profile of a new site based on its values for the corresponding indices. This is explained in the next section.

4.2 Assessing a stream site from the lattice

According to the *ecological ambiance system* described in [20], several lattices have been built for clustering sites with similar average values (or *alteration* degrees[3]) on the five biological indices. The underlying hypothesis is that global state of an hydro-ecosystem can be assessed on the basis of the five biological indices and synthesised by the alteration degree. Sites with similar alteration degrees can be compared even if they represent various profiles. The intervals of similarity have been defined by the hydro-ecologists [18]. For example, the lattice in Figure 2 was obtained from a set of sites with an alteration degree belonging to [2.5 ; 3] (see *C7* context in table 1). The classes of indices in the lattice vary between 1 and 5. Each site is represented alone in an atom of the lattice, which is coherent with the choices done in the project, trying to represent all the variety of streams or water areas in the Alsace plain.

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

[3] The alteration degree is computed as the average value of the five biological indices, e.g. the alteration degree of BW2 equals 13/5. Currently the physico-chemical parameters are not taken into account.

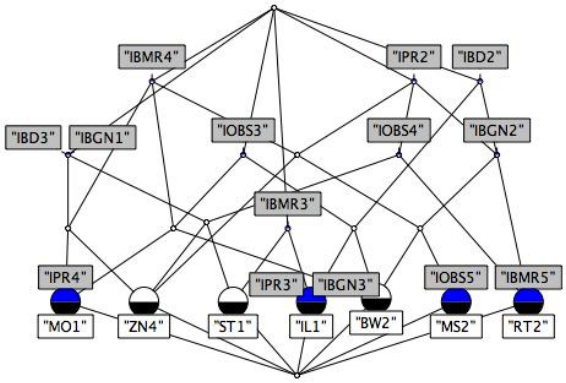


Fig. 2: The lattice based on the context of table 1 (linear scale)

Let us now suppose that we have got a partial information on a new stream site, denoted Q, defined by the following values: IBGN 2 IBMR 4 IOBS 3 IPR 2 (IBD missing). Its alteration degree is $2.75 \in [2.5\ ;\ 3]$, Q can thus be compared to the stream sites represented in the *C7* lattice. This is done by classifying Q within this lattice, as shown in Figure 3.

Looking at the lattice in Figure 3, one can see that the Q site-query has four common values with only the BW2 site (Brunnwasser downstream). The expert qualification of BW2 (except for the IBD index) can thus be used to assess the Q site. The Q site could thus be assessed as follows: *the habitat quality and the water physico-chemical quality are good, expect for nutriments (nitrate and phosphor mineral forms) which quality is medium; the sediment quality is medium, the resilience potential of the general ecosystem is good, while the resilience potential of sediments is deteriorated.*

5 Implementation

5.1 Algorithms

As explained before, the built lattices have to be queried for assessing new sites. Furthermore, they could have to be updated, by adding a new site, or by modifying an existing site. The new/updated object is described by attributes which can exist in the context of the lattice or not. In this paper we only consider the case where the attributes already exist. Two algorithms described by Carpineto and Romano [10] have been implemented, the first one allows to add a new object in a lattice, while the second one allows to delete an object from a lattice.

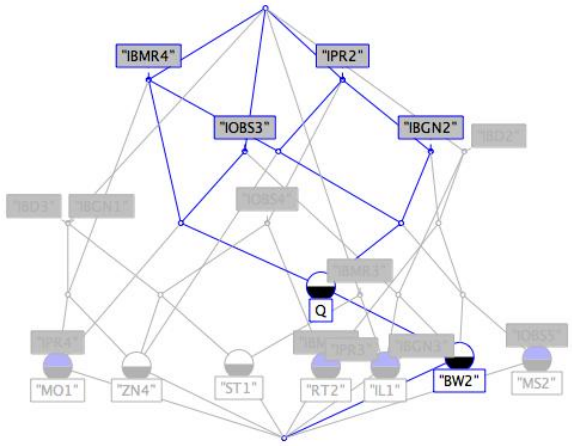


Fig. 3: The *C7* lattice with the Q site-query inserted

The first algorithm allows to add a new object into an existing Galois lattice, which can be interpreted as classifying a new object. It takes as input a Galois lattice and the new object with its attributes. The output is the updated Galois lattice of the new context. The mechanism of the algorithm is as follows. The set of the concepts is divided into subsets according to their intent cardinality, and then analysed in ascending order. For each concept of a subset, if the intent is included in or equal to the set of the new object attributes then the current concept extent is augmented by the new object; otherwise a new concept is created, after verifying that such a concept is not in the initial set of concepts or among the new added ones. The intent of this new concept is determined by the intersection of the current concept intent and the new object attributes; its extent is defined by the current concept extent augmented with the new object. After the addition of a new concept a new link between this concept and the current concept is created. The links with neighbouring concepts are also updated.

The second algorithm allows to delete an object from a lattice. It takes as input a Galois lattice and the object to be removed. The output is the updated Galois lattice of the new context. The mechanism of the algorithm is as follows. For each concept, if the object to be deleted is included or equal to the current concept extent, then it is removed from this extent. If the modified concept has then the same extent as one of its children, it is deleted. When a concept is removed the links among the concepts are updated.

The modification of an existing object in a Galois lattice is performed in two steps: (1) deleting this object using the second algorithm; (2) adding the updated object using the first algorithm. The whole process could be improved with a third algorithm for adding attributes into the lattice context, allowing to enrich the initial lattice with new information.

5.2 User interface and manipulation

The user interface allows to use a lattice either stored in the database or stored in a `XML` file with the structure used in the software Galicia[4]. Three main functional views are provided to the user. The first one allows to qualify concepts, i.e. to describe the profile of a set of sites. The second one allows to define a query, i.e. a new site to be assessed according to an existing lattice. The third view allows to explore the result of the query, i.e. to compare the characteristics of the new site to those of the already assessed sites. Currently texts appearing on the interface views are written in French since the target users are French. Other languages could be used in the future.

The functional view for qualifying concepts is presented on Figure 4. Once a lattice is chosen, it is possible to select a given concept in a list and to see its description (intent, extent, and comment). The lists of the parents and children of that concept are also shown, and by a click on one of them, we see its related information. These information may help the experts in qualifying the concept. The comment is then stored in the database.

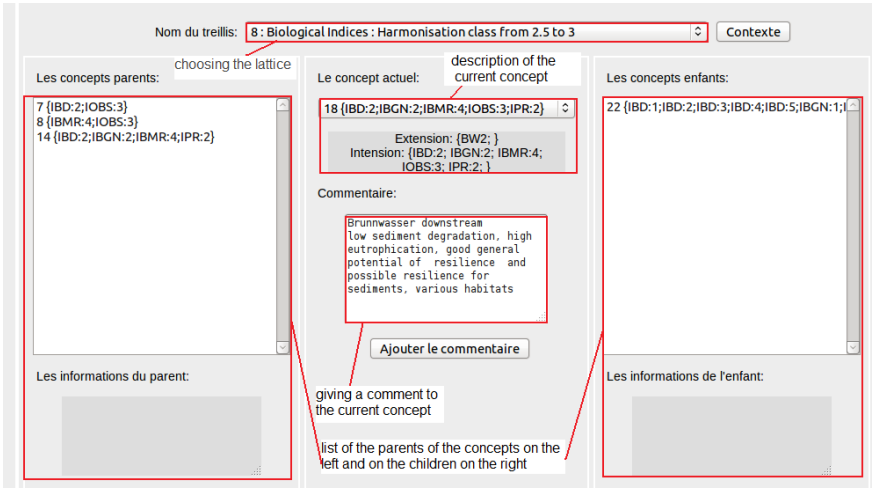


Fig. 4: Qualifying the concepts of the site lattice

[4] `http://www.iro.umontreal.ca/~galicia/`

The functionality for classifying a new site based on its values (for one or several indices) is presented on Figure 5. One has first to select a lattice and to give a name for the new site, and then to provide a description of this new site by choosing indices and their values. Once this is done, it is possible to classify the site, that is to integrate it in the lattice, either temporarily or to save it in the lattice. The button "Classer" allows this classification. To interpret the result, the button "Visualiser le résultat" can be used to see the new lattice with the modifications shown in a specific colour. The button "Explorer le treillis" also helps in the interpretation by giving access to a third view (Figure 6) where it is possible to navigate within the concepts and see the description of the parents and children of the current concept.

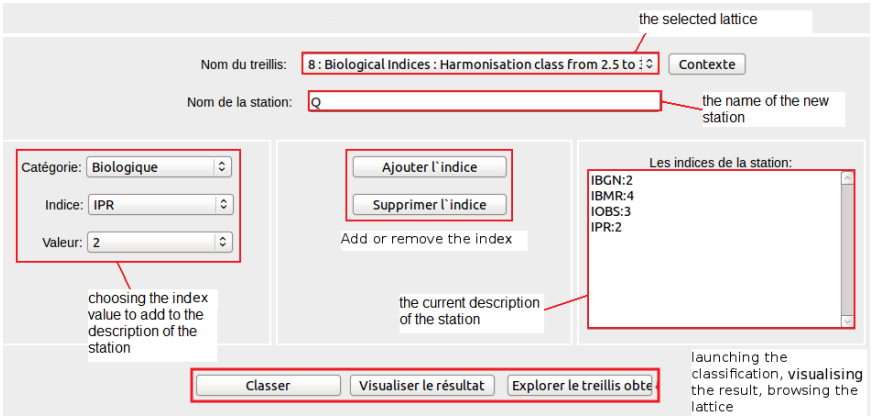


Fig. 5: Definition of the Q site-query

More precisely, the third view allows to explore only the modified or new concepts of the lattice, i.e. the concepts where the site-query is represented. These concepts can be commented and the modified lattice can be stored in the database. Eventually, the commented lattices can be exported in various formats to be further analysed.

6 Discussion

We decided to implement a specific tool for several reasons:

1. the tool has to be interconnected with a database and to offer a user-friendly interface for hydro-ecologists, allowing them to annotate the concepts;
2. the purpose of the tool is not navigating throughout the whole database;
3. this is a two-stage tool: the first stage organises a specific information within a lattice; the second stage allows the user to explore and possibly modify this lattice.

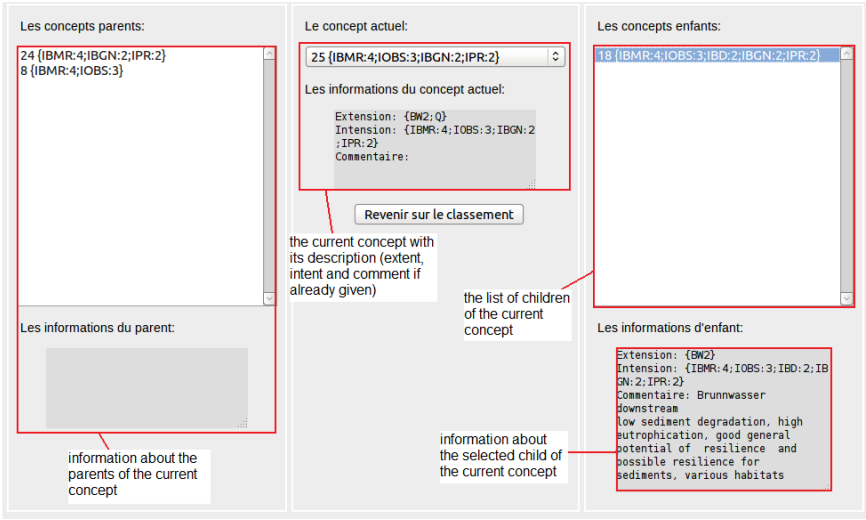


Fig. 6: Analysing the classification result of the Q query

Regarding the first point, lattice-builder tools like Galicia, ConExp, or the Toscana suite[5] cannot be used, since they do not fit the requirements of hydro-ecologists. Actually, as said before, we have used Galicia to build the lattices which are then recorded in the database to be annotated and explored by hydro-ecologists. Besides, the lattices built through our tool can be exported into a Galicia format.

Regarding the second point, our approach differs from those used in search or browsing tools like Camelis [13], Abilis [6], D-SIFT [12] or in the Virtual Museum of the Pacific [26]. Indeed we did not try to implement a lattice-based approach to explore the whole database, but only specific information from this database. This information was chosen by hydro-ecologists as a synthetic view of the database. Furthermore, the lattice is used as a basis to record expert knowledge (the annotations) that can be involved in further investigations.

Regarding the last point, our tool can be compared to Ulysses [9] which is a visual interface allowing to access a lattice structure organising information from a database. Ulysses allows the user to search the retrieval space both by browsing or querying, whereas our tool only allows querying. Nevertheless, the originality of our tool is the user possibility of modifying and annotating the lattice concepts.

Finally, the underlying aim of our approach is to build an ontology, gathering the knowledge of various experts on hydro-ecosystems. Each expert indeed focuses on a specific compartment of the hydro-ecosystems (e.g. fishes, macro-

[5] `http://toscanaj.sourceforge.net/`

phytes, diatoms...) and a generic tool is needed to combine their expertises and produce a global assessment of the ecological state of a stream site.

7 Conclusion

This paper presents a lattice-based query system for helping the assessment of hydro-ecosystems. The approach relies on a database storing various information on stream sites of the Alsace plain. These data are summarised within qualitative indices, biological indices or physico-chemical and physical indices. Based on these indices and their own expertise, hydro-ecologists can perform a global evaluation of the functioning of a stream ecosystem. Furthermore, they want to define quality profiles of streams or water areas that could be used to assess new sites. Eventually a tool is needed to help the whole process.

Our work aims at building such a tool. Concept lattices appeared as a good approach since they allow both to build hierarchical clustering of sites, to navigate through the clusters, and to perform queries for helping the assessment of a new site. The clustering aspects already proved to be interesting, and the user interface allowing to comment and query the lattices is currently being experimented by hydro-ecologists. In the future, several lattices have to be built including various sets of indices (physico-chemical and physical indices). Furthermore, the whole approach will be tested with stream or water area data from other regions in France.

Regarding the implementation aspects, the system should be improved in two ways: allowing the integration of new attributes in an existing lattice and allowing the navigation through bigger lattices. Finally improvements can be done to provide self-building comments on the site-queries, based on the comments of the neighbouring concepts.

Acknowledgements

The INDICE project (2006-11) was supported by the Agence de l'Eau Rhin-Meuse. We also acknowledge the scientific and technical help of the Cemagref Centre in Lyon, the Gabriel Lippmann Public Research Centre in Luxembourg and the regional delegation of ONEMA (Office National de l'Eau et des Milieux Aquatiques). Cristina Nica's stay in France was supported by the Erasmus European program. We acknowledge the anonymous reviewers who helped us to improve our paper.

References

1. AFNOR: Qualité de l'eau : détermination de l'Indice Biologique Global Normalisé (IBGN). NF T90-350 (1992), révision 2004
2. AFNOR: Qualité de l'eau : détermination de l'Indice Biologique Diatomées (IBD. NF T90-354 (2000), révision 2007

3. AFNOR: Qualité de l'eau : détermination de l'Indice Oligochètes de Bioindication des Sédiments (IOBS). NF T90-390 (2002)
4. AFNOR: Qualité de l'eau : détermination de l'Indice Biologique Macrophytique en Rivière (IBMR). NF T90-395 (2003)
5. AFNOR: Qualité de l'eau : détermination de l'Indice poissons rivière (IPR). NF T90-344 (2004)
6. Allard, P., Ferré, S., Ridoux, O.: Discovering Functional Dependencies and Association Rules by Navigating in a Lattice of OLAP Views. In: Kryszkiewicz, M., Obiedkov, S. (eds.) Proceedings of CLA 2010, Sevilla, Spain. pp. 199–210 (2010)
7. Barbut, M., Monjardet, B.: Ordre et classification – Algèbre et combinatoire. Hachette (1970)
8. Bedel, O., Ferré, S., Ridoux, O., Quesseveur, E.: GEOLIS: a logical information system for geographical data. Revue Internationale de Géomatique 17, 371–390 (2007)
9. Carpineto, C., Romano, G.: ULYSSES: A Lattice-based Multiple Interaction Strategy Retrieval Interface. In: Blumenthal, B., Gornostaev, J., Unger, C. (eds.) Human-Computer Interaction, 5th International Conference, EWHCI'95, Moscow, Russia. LNCS, vol. 1015, pp. 91–104. Springer-Verlag (1995)
10. Carpineto, C., Romano, G.: Concept Data Analysis. Theory and Applications. John Wiley & Sons Ltd (2004), 201 pages
11. Carpineto, C., Romano, G.: Using concept lattices for text retrieval and mining. In: Ganter, B., Stumme, G., Wille, R. (eds.) Formal Concept Analysis, LNCS, vol. 3626, pp. 3–45. Springer Berlin / Heidelberg (2005)
12. Ducrou, J., Wormuth, B., Eklund, P.: D-SIFT: A Dynamic Simple Intuitive FCA Tool. In: Dau, F., Mugnier, M.L., Stumme, G. (eds.) Conceptual Structures: Common Semantics for Sharing Knowledge – Proceedings of ICCS 2005. vol. LNAI 3596, pp. 295–306. Springer-Verlag (2005)
13. Ferré, S.: Camelis: a logical information system to organise and browse a collection of documents. International Journal of General Systems 38(4), 379–403 (2009)
14. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer Verlag (1999)
15. Godin, R., Missaoui, R., Alaoui, H.: Incremental concept formation algorithm based on Galois (concept) lattices. Computational Intelligence 11(2), 246–267 (1995)
16. Godin, R., Missaoui, R., April, A.: Experimental comparison of navigation in a Galois lattice with conventional information retrieval method. International Journal of Man-Machine Studies 38, 747–767 (1993)
17. Grac, C., Braud, A., Le Ber, F., Trémolières, M.: Un système d'information pour le suivi et l'évaluation de la qualité des cours d'eau – Application à l'hydro-région de la plaine d'Alsace. RSTI - Ingénierie des Systèmes d'Information 16, 9–30 (2011)
18. 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)
19. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. J. Exp. Theor. Artif. Inelligence 14(2-3), 189–216 (2002)
20. Lafont, M.: A conceptual approach to the biomonitoring of freshwater: the ecological ambience system. Journal of Limnology 6, 17–24 (2001)
21. Lafont, M., Jézéquel, C., Vivier, A., Breil, P., Schmitt, L., Bernoud, S.: Refinement of biomonitoring of urban water courses by combining descriptive and ecohydrological approaches. Ecohydrol. Hydrobiol. 10, 3–11 (2010)

22. MEDD: Système d'évaluation de la qualité de l'eau des cours d'eau (SEQ-Eau), version 2. Ministère de l'Ecologie et du Développement Durable et Agences de l'Eau (2003), Étude inter-agences de l'eau, no 52
23. MEDD: Circulaire dce 2007/22 du 11 avril 2007 relative au protocole de prélèvement et de traitement des échantillons des invertébrés pour la mise en œuvre du programme de surveillance sur cours d'eau. Ministère de l'Ecologie et du Développement Durable (2007)
24. Napoli, A.: A smooth introduction to symbolic methods in knowledge discovery. In: Cohen, H., Lefebvre, C. (eds.) Categorization in Cognitive Science. Elsevier (2006)
25. Priss, U.: Lattice-based information retrieval. Knowledge Organization 27(3), 132142 (2000)
26. Wray, T., Eklund, P.: Exploring the Information Space of Cultural Collections Using Formal Concept Analysis. In: Valtchev, P., Jäschke, R. (eds.) Proceedings of 9th International Conference on Formal Concept Analysis, ICFCA 2011, Nicosia, Cyprus. LNAI, vol. 6628, pp. 251–266. Springer-Verlag (2011)

The Word Problem in Semiconcept Algebras

Philippe Balbiani

CNRS — Université de Toulouse
Institut de recherche en informatique de Toulouse
118 ROUTE DE NARBONNE, 31062 TOULOUSE CEDEX 9, France
Philippe.Balbiani@irit.fr

Abstract. The aim of this article is to prove that the word problem in semiconcept algebras is **PSPACE**-complete.

Keywords: Formal concept analysis, semiconcept algebras, word problem, decidability/complexity.

1 Introduction

In formal concept analysis [2, 3], the properties of formal contexts are reflected by the properties of the concept lattices they give rise to [10, 12]. Extending concept lattices to protoconcept algebras and semiconcept algebras, Herrmann *et al.* [5] and Wille [11] introduced negations in conceptual structures based on formal contexts such as double Boolean algebras and pure double Boolean algebras. These algebras have attracted interest for their theoretical merits — basic representations have been obtained — and for their practical relevance — applications in the field of knowledge representation and reasoning have been developed [5–7, 9, 11].

The basic representations of protoconcept algebras and semiconcept algebras evoked above have been obtained by means of equational axioms. Hence, the problem naturally arises of whether there is an algorithm which given terms s, t, decides whether they represent the same element in all models of these equational axioms. Such a problem is called the word problem (WP) in protoconcept algebras or in semiconcept algebras. In Mathematics and Computer Science, word problems are of the utmost importance.

Within the context of protoconcept algebras, Vormbrock [8] demonstrates that given terms s, t, if $s = t$ is not valid in all protoconcept algebras then there exists a finite protoconcept algebra in which $s = t$ is not valid. Nevertheless, the upper bound on the size of the finite protoconcept algebra given in [8, Page 258] is not elementary. Therefore, it does not allow us to conclude — as wrongly stated in [8, Page 240] — that the WP in protoconcept algebras is **NP**-complete.

Switching over to semiconcept algebras, the aim of this article is to prove that the WP in semiconcept algebras is **PSPACE**-complete.

Sections 2 and 3 show some of the basic properties of formal contexts and semiconcept algebras that have been discussed in [5–7, 9, 11]. In Section 4, we present

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

the WP in semiconcept algebras. Section 5 introduces a basic 2-sorted modal logic that will be used in Sections 6 and 7 to prove that the WP in semiconcept algebras is **PSPACE**-complete. The proofs of Lemmas 10, 11, 12 and 13 can be found in the annex.

2 From Formal Contexts to Semiconcept Algebras

In formal concept analysis, the properties of semiconcepts are reflected by the properties of the algebras they give rise to.

2.1 Formal Contexts

Formal contexts are structures of the form $\mathbb{K} = (G, M, \vartriangle)$ where G is a nonempty set (with typical member denoted g), M is a nonempty set (with typical member denoted m) and $\vartriangle$ is a binary relation between G and M. The elements of G are called "objects", the elements of M are called "attributes" and the intended meaning of $g \vartriangle m$ is "object g possesses attribute m".

$\vartriangle$	a_1	a_2
o_1	$\times$	$\times$
o_2	$\times$	

Tab. 1.

Example 1. In *Tab.* 1 is an example of a formal context $\mathbb{K}^{2,2}$ with 2 objects — o_1 and o_2 — and 2 attributes — a_1 and a_2.

For all $X \subseteq G$ and for all $Y \subseteq M$, let

$$X^{\triangleright} = \{m \in M\text{: for all } g \in G, \text{ if } g \in X \text{ then } g \vartriangle m\}$$
$$Y^{\triangleleft} = \{g \in G\text{: for all } m \in M, \text{ if } m \in Y \text{ then } g \vartriangle m\}$$

That is to say, $X^{\triangleright}$ is the set of all attributes possessed by all objects in X and $Y^{\triangleleft}$ is the set of all objects possessing all attributes in Y.

Example 2. In the formal context $\mathbb{K}^{2,2}$ of *Tab.* 1, $\{o_1\}^{\triangleright} = \{a_1, a_2\}$ and $\{a_2\}^{\triangleleft} = \{o_1\}$.

To carry out our plan, we need to learn a little more about the pair $({}^{\triangleright}, {}^{\triangleleft})$ of maps ${}^{\triangleright}\colon 2^G \mapsto 2^M$ and ${}^{\triangleleft}\colon 2^M \mapsto 2^G$. Obviously, for all $X \subseteq G$ and for all $Y \subseteq M$,

- $X \subseteq Y^{\triangleleft}$ iff $X^{\triangleright} \supseteq Y$.

Hence, the pair $({}^{\triangleright}, {}^{\triangleleft})$ of maps ${}^{\triangleright}\colon 2^G \mapsto 2^M$ and ${}^{\triangleleft}\colon 2^M \mapsto 2^G$ is a Galois connection between $(2^G, \subseteq)$ and $(2^M, \supseteq)$. Thus, for all $X, X_1, X_2 \subseteq G$ and for all $Y, Y_1, Y_2 \subseteq M$,

- if $X_1 \subseteq X_2$ then $X_1^{\triangleright} \supseteq X_2^{\triangleright}$,
- if $Y_1 \supseteq Y_2$ then $Y_1^{\triangleleft} \subseteq Y_2^{\triangleleft}$,
- $X \subseteq X^{\triangleright\triangleleft}$ and $X^{\triangleright} = X^{\triangleright\triangleleft\triangleright}$,
- $Y^{\triangleleft\triangleright} \supseteq Y$ and $Y^{\triangleleft} = Y^{\triangleleft\triangleright\triangleleft}$.

2.2 Semiconcept Algebras

Let $\mathbb{K} = (G, M, \Delta)$ be a formal context. Given $X \subseteq G$, the pair $(X, X^{\triangleright})$ is called "left semiconcept of $\mathbb{K}$". Remark that $(\emptyset, M)$ is a left semiconcept of $\mathbb{K}$. Let

$$\underline{\mathcal{H}_l}(\mathbb{K}) = (\mathcal{H}_l(\mathbb{K}), \bot_l, \top_l, \neg_l, \vee_l, \wedge_l)$$

be the algebraic structure of type $(0, 0, 1, 2, 2)$ where $\mathcal{H}_l(\mathbb{K})$ is the set of all left semiconcepts of $\mathbb{K}$, $\bot_l = (\emptyset, M)$, $\top_l = (G, G^{\triangleright})$, $\neg_l(X, X^{\triangleright}) = (G \setminus X, (G \setminus X)^{\triangleright})$, $(X_1, X_1^{\triangleright}) \vee_l (X_2, X_2^{\triangleright}) = (X_1 \cup X_2, (X_1 \cup X_2)^{\triangleright})$ and $(X_1, X_1^{\triangleright}) \wedge_l (X_2, X_2^{\triangleright}) = (X_1 \cap X_2, (X_1 \cap X_2)^{\triangleright})$. Remark that if G is finite then $\mathcal{H}_l(\mathbb{K})$ is finite too and moreover, $\mid \mathcal{H}_l(\mathbb{K}) \mid = 2^{|G|}$. It is a simple exercise to check that the above operations $\bot_l$, $\top_l$, $\neg_l \cdot$, $\cdot \vee_l \cdot$ and $\cdot \wedge_l \cdot$ on $\mathcal{H}_l(\mathbb{K})$ are isomorphic to the Boolean operations $\emptyset$, G, $G \setminus \cdot$, $\cdot \cup \cdot$ and $\cdot \cap \cdot$ on 2^G. Hence, $\underline{\mathcal{H}_l}(\mathbb{K})$ satisfies the conditions of nondegenerate Boolean algebras. Given $Y \subseteq M$, the pair $(Y^{\triangleleft}, Y)$ is called "right semiconcept of $\mathbb{K}$". Remark that $(G, \emptyset)$ is a right semiconcept of $\mathbb{K}$. Let

$$\underline{\mathcal{H}_r}(\mathbb{K}) = (\mathcal{H}_r(\mathbb{K}), \bot_r, \top_r, \neg_r, \vee_r, \wedge_r)$$

be the algebraic structure of type $(0, 0, 1, 2, 2)$ where $\mathcal{H}_r(\mathbb{K})$ is the set of all right semiconcepts of $\mathbb{K}$, $\bot_r = (M^{\triangleleft}, M)$, $\top_r = (G, \emptyset)$, $\neg_r(Y^{\triangleleft}, Y) = ((M \setminus Y)^{\triangleleft}, M \setminus Y)$, $(Y_1^{\triangleleft}, Y_1) \vee_r (Y_2^{\triangleleft}, Y_2) = ((Y_1 \cap Y_2)^{\triangleleft}, Y_1 \cap Y_2)$ and $(Y_1^{\triangleleft}, Y_1) \wedge_r (Y_2^{\triangleleft}, Y_2) = ((Y_1 \cup Y_2)^{\triangleleft}, Y_1 \cup Y_2)$. Remark that if M is finite then $\mathcal{H}_r(\mathbb{K})$ is finite too and moreover, $\mid \mathcal{H}_r(\mathbb{K}) \mid = 2^{|M|}$. It is a simple exercise to check that the above operations $\bot_r$, $\top_r$, $\neg_r \cdot$, $\cdot \vee_r \cdot$ and $\cdot \wedge_r \cdot$ on $\mathcal{H}_r(\mathbb{K})$ are anti-isomorphic to the Boolean operations $\emptyset$, M, $M \setminus \cdot$, $\cdot \cup \cdot$ and $\cdot \cap \cdot$ on 2^M. Hence, $\underline{\mathcal{H}_r}(\mathbb{K})$ satisfies the conditions of nondegenerate Boolean algebras. Now, for the concept underlying most of our work in this article. Given $X \subseteq G$ and $Y \subseteq M$, the pair (X, Y) is called "semiconcept of $\mathbb{K}$" iff $Y = X^{\triangleright}$ or $X = Y^{\triangleleft}$. Remark that $(\emptyset, M)$ and $(G, \emptyset)$ are semiconcepts of $\mathbb{K}$.

Example 3. In the formal context $\mathbb{K}^{2,2}$ of *Tab.* 1, the semiconcepts are $(\emptyset, \{a_1, a_2\})$, $(\{o_1\}, \{a_1, a_2\})$, $(\{o_2\}, \{a_1\})$, $(\{o_1\}, \{a_2\})$, $(\{o_1, o_2\}, \{a_1\})$ and $(\{o_1, o_2\}, \emptyset)$.

Let

$$\underline{\mathcal{H}}(\mathbb{K}) = (\mathcal{H}(\mathbb{K}), \bot_l, \bot_r, \top_l, \top_r, \neg_l, \neg_r, \vee_l, \vee_r, \wedge_l, \wedge_r)$$

be the algebraic structure of type $(0, 0, 0, 0, 1, 1, 2, 2, 2, 2)$ where $\mathcal{H}(\mathbb{K})$ is the set of all semiconcepts of $\mathbb{K}$, $\bot_l = (\emptyset, M)$, $\bot_r = (M^{\triangleleft}, M)$, $\top_l = (G, G^{\triangleright})$, $\top_r = (G, \emptyset)$, $\neg_l(X, Y) = (G \setminus X, (G \setminus X)^{\triangleright})$, $\neg_r(X, Y) = ((M \setminus Y)^{\triangleleft}, M \setminus Y)$, $(X_1, Y_1) \vee_l (X_2, Y_2) = (X_1 \cup X_2, (X_1 \cup X_2)^{\triangleright})$, $(X_1, Y_1) \vee_r (X_2, Y_2) = ((Y_1 \cap Y_2)^{\triangleleft}, Y_1 \cap Y_2)$, $(X_1, Y_1) \wedge_l (X_2, Y_2) = (X_1 \cap X_2, (X_1 \cap X_2)^{\triangleright})$ and $(X_1, Y_1) \wedge_r (X_2, Y_2) = ((Y_1 \cup Y_2)^{\triangleleft}, Y_1 \cup Y_2)$.

Example 4. In the formal context $\mathbb{K}^{2,2}$ of *Tab.* 1, $\bot_l = (\emptyset, \{a_1, a_2\})$, $\top_l = (\{o_1, o_2\}, \{a_1\})$, $\bot_r = (\{o_1\}, \{a_1, a_2\})$ and $\top_r = (\{o_1, o_2\}, \emptyset)$.

Remark that if G, M are finite then $\mathcal{H}(\mathbb{K})$ is finite too and moreover, $\mid \mathcal{H}(\mathbb{K}) \mid \leq 2^{|G|} + 2^{|M|}$. Obviously, the operations $\bot_l$, $\top_l$, $\neg_l \cdot$, $\cdot \vee_l \cdot$ and $\cdot \wedge_l \cdot$, when restricted to the set of all left semiconcepts of $\mathbb{K}$, are isomorphic to the Boolean operations

$\emptyset$, G, $G \setminus \cdot$, $\cdot \cup \cdot$ and $\cdot \cap \cdot$ on 2^G whereas the operations $\bot_r$, $\top_r$, $\neg_r \cdot$, $\cdot \vee_r \cdot$ and $\cdot \wedge_r \cdot$, when restricted to the set of all right semiconcepts of $\mathbb{IK}$, are anti-isomorphic to the Boolean operations $\emptyset$, M, $M \setminus \cdot$, $\cdot \cup \cdot$ and $\cdot \cap \cdot$ on 2^M. In other respects, it is a simple matter to check that $\underline{\mathcal{H}}(\mathbb{IK})$ satisfies the following conditions for every $x, y, z \in \mathcal{H}(\mathbb{IK})$:

- $x \wedge_l (y \wedge_l z) = (x \wedge_l y) \wedge_l z$ and $x \vee_r (y \vee_r z) = (x \vee_r y) \vee_r z$,
- $x \wedge_l y = y \wedge_l x$ and $x \vee_r y = y \vee_r x$,
- $\neg_l (x \wedge_l x) = \neg_l x$ and $\neg_r (x \vee_r x) = \neg_r x$,
- $x \wedge_l (y \wedge_l y) = x \wedge_l y$ and $x \vee_r (y \vee_r y) = x \vee_r y$,
- $x \wedge_l (y \vee_l z) = (x \wedge_l y) \vee_l (x \wedge_l z)$ and $x \vee_r (y \wedge_r z) = (x \vee_r y) \wedge_r (x \vee_r z)$,
- $x \wedge_l (x \vee_l y) = x \wedge_l x$ and $x \vee_r (x \wedge_r y) = x \vee_r x$,
- $x \wedge_l (x \vee_r y) = x \wedge_l x$ and $x \vee_r (x \wedge_l y) = x \vee_r x$,
- $\neg_l (\neg_l x \wedge_l \neg_l y) = x \vee_l y$ and $\neg_r (\neg_r x \vee_r \neg_r y) = x \wedge_r y$,
- $\neg_l \bot_l = \top_l$ and $\neg_r \top_r = \bot_r$,
- $\neg_l \top_r = \bot_l$ and $\neg_r \bot_l = \top_r$,
- $\top_r \wedge_l \top_r = \top_l$ and $\bot_l \vee_r \bot_l = \bot_r$,
- $x \wedge_l \neg_l x = \bot_l$ and $x \vee_r \neg_r x = \top_r$,
- $\neg_l \neg_l (x \wedge_l y) = x \wedge_l y$ and $\neg_r \neg_r (x \vee_r y) = x \vee_r y$,
- $(x \wedge_l x) \vee_r (x \wedge_l x) = (x \vee_r x) \wedge_l (x \vee_r x)$,
- $x \wedge_l x = x$ or $x \vee_r x = x$.

Let us remark that the first 13 above conditions come in pairs of mirror images obtained by interchanging $\bot_l$ with $\top_r$, $\top_l$ with $\bot_r$, $\neg_l$ with $\neg_r$, $\vee_l$ with $\wedge_r$ and $\wedge_l$ with $\vee_r$ whereas the last 2 above conditions are equivalent to their own mirror images. This leads us to the principle of duality stating that from any condition provable from the 15 above conditions, another such condition results immediately by interchanging $\bot_l$ with $\top_r$, $\top_l$ with $\bot_r$, $\neg_l$ with $\neg_r$, $\vee_l$ with $\wedge_r$ and $\wedge_l$ with $\vee_r$. The set $\mathcal{H}(\mathbb{IK})$ can be ordered by the binary relation $\sqsubseteq$ defined by

$$(X_1, Y_1) \sqsubseteq (X_2, Y_2) \text{ iff } X_1 \subseteq X_2 \text{ and } Y_1 \supseteq Y_2$$

for every $(X_1, Y_1), (X_2, Y_2) \in \mathcal{H}(\mathbb{IK})$. Obviously, for all $(X_1, Y_1), (X_2, Y_2) \in \mathcal{H}(\mathbb{IK})$,

- $(X_1, Y_1) \sqsubseteq (X_2, Y_2)$ iff $(X_1, Y_1) \wedge_l (X_2, Y_2) = (X_1, Y_1) \wedge_l (X_1, Y_1)$ and $(X_1, Y_1) \vee_r (X_2, Y_2) = (X_2, Y_2) \vee_r (X_2, Y_2)$,
- if $(X_1, Y_1) \in \mathcal{H}_l(\mathbb{IK})$ then $(X_1, Y_1) \sqsubseteq (X_2, Y_2)$ iff $(X_1, Y_1) \wedge_l (X_2, Y_2) = (X_1, Y_1)$,
- if $(X_2, Y_2) \in \mathcal{H}_r(\mathbb{IK})$ then $(X_1, Y_1) \sqsubseteq (X_2, Y_2)$ iff $(X_1, Y_1) \vee_r (X_2, Y_2) = (X_2, Y_2)$.

Moreover, the binary relation $\sqsubseteq$ is reflexive, antisymmetric and transitive on $\mathcal{H}(\mathbb{IK})$. In order to give an abstract characterization of the operations $\bot_l$, $\bot_r$, $\top_l$, $\top_r$, $\neg_l$, $\neg_r$, $\vee_l$, $\vee_r$, $\wedge_l$ and $\wedge_r$, we shall say that an algebraic structure $\mathcal{D} = (D, \bot_l, \bot_r, \top_l, \top_r, \neg_l, \neg_r, \vee_l, \vee_r, \wedge_l, \wedge_r)$ of type $(0, 0, 0, 0, 1, 1, 2, 2, 2, 2)$ is a pure double Boolean algebra iff the operations $\bot_l$, $\bot_r$, $\top_l$, $\top_r$, $\neg_l$, $\neg_r$, $\vee_l$, $\vee_r$, $\wedge_l$ and $\wedge_r$ satisfy the 15 above conditions.

3 From Semiconcept Algebras to Formal Contexts

The aim of this section is to give an abstract characterization of the operations $\bot_l$, $\bot_r$, $\top_l$, $\top_r$, $\neg_l$, $\neg_r$, $\vee_l$, $\vee_r$, $\wedge_l$ and $\wedge_r$.

3.1 Filters and Ideals

Let $\mathcal{D} = (D, \bot_l, \bot_r, \top_l, \top_r, \neg_l, \neg_r, \vee_l, \vee_r, \wedge_l, \wedge_r)$ be a pure double Boolean algebra. We define

$$D_l = \{x \wedge_l x \colon x \in D\}$$
$$D_r = \{x \vee_r x \colon x \in D\}$$

Intuitively, elements of D_l can be considered as sets of objects and elements of D_r can be considered as sets of attributes.

Example 5. In the semiconcept algebra associated to the formal context $\mathbb{K}^{2,2}$ of *Tab.* 1, $D^{2,2} = \{(\emptyset, \{a_1, a_2\}), (\{o_1\}, \{a_1, a_2\}), (\{o_2\}, \{a_1\}), (\{o_1\}, \{a_2\}), (\{o_1, o_2\}, \{a_1\}), (\{o_1, o_2\}, \emptyset)\}$, $D_l^{2,2} = \{(\emptyset, \{a_1, a_2\}), (\{o_1\}, \{a_1, a_2\}), (\{o_2\}, \{a_1\}), (\{o_1, o_2\}, \{a_1\})\}$ and $D_r^{2,2} = \{(\{o_1\}, \{a_1, a_2\}), (\{o_1, o_2\}, \{a_1\}), (\{o_1\}, \{a_2\}), (\{o_1, o_2\}, \emptyset)\}$.

Obviously, the operations $\bot_l$, $\top_l$, $\neg_l$, $\vee_l$ and $\wedge_l$ are stable on D_l and the operations $\bot_r$, $\top_r$, $\neg_r$, $\vee_r$ and $\wedge_r$ are stable on D_r. Hence, the algebraic structures $\mathcal{D}_l = (D_l, \bot_l, \top_l, \neg_l, \vee_l, \wedge_l)$ and $\mathcal{D}_r = (D_r, \bot_r, \top_r, \neg_r, \vee_r, \wedge_r)$ are algebraic structures of type $(0, 0, 1, 2, 2)$. More precisely, they are Boolean algebras. Moreover, the set D can be ordered by the binary relation $\leq$ defined by

$$x \leq y \text{ iff } x \wedge_l y = x \wedge_l x \text{ and } x \vee_r y = y \vee_r y$$

for every $x, y \in D$. Obviously, for all $x, y \in D$,

- if $x \in D_l$ then $x \leq y$ iff $x \wedge_l y = x$,
- if $y \in D_r$ then $x \leq y$ iff $x \vee_r y = y$.

Moreover, the binary relation $\leq$ is reflexive, antisymmetric and transitive on D.

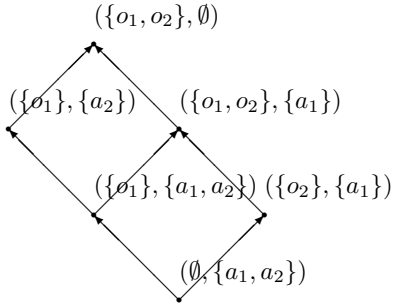


Fig. 1.

Example 6. In *Fig.* 1 is represented the binary relation $\leq^{2,2}$ ordering the set $D^{2,2}$ of the semiconcept algebra associated to the formal context $\mathbb{K}^{2,2}$ of *Tab.* 1.

A nonempty subset F of D is called a filter iff for all $x, y \in D$,

- $x, y \in F$ implies $x \wedge_l y \in F$,
- $x \in F$ and $x \leq y$ imply $y \in F$.

A nonempty subset I of D is called an ideal iff for all $x, y \in D$,

- $x, y \in I$ implies $x \vee_r y \in I$,
- $x \in I$ and $y \leq x$ imply $y \in I$.

The following lemma explains how filters and ideals can be transformed into filters and ideals of the Boolean algebras $\mathcal{D}_l$ and $\mathcal{D}_r$.

Lemma 1. *Let F, I be nonempty subsets of D. If F is a filter then $F \cap D_l$ is a filter of the Boolean algebra $\mathcal{D}_l$ and $F \cap D_r$ is a filter of the Boolean algebra $\mathcal{D}_r$ and if I is an ideal then $I \cap D_l$ is an ideal of the Boolean algebra $\mathcal{D}_l$ and $I \cap D_r$ is an ideal of the Boolean algebra $\mathcal{D}_r$.*

Let F be a nonempty subset of D_l and I be a nonempty subset of D_r. We define

$$[F) = \{x \in D\text{: there exists } y \in F \text{ such that } y \leq x\}$$
$$(I] = \{x \in D\text{: there exists } y \in I \text{ such that } x \leq y\}$$

The following lemma explains how filters of the Boolean algebra $\mathcal{D}_l$ and ideals of the Boolean algebra $\mathcal{D}_r$ can be transformed into filters and ideals.

Lemma 2. *Let F be a nonempty subset of D_l, I be a nonempty subset of D_r. If F is a filter of the Boolean algebra $\mathcal{D}_l$ then $[F)$ is a filter and $[F) \cap D_l = F$ and if I is an ideal of the Boolean algebra $\mathcal{D}_r$ then $(I]$ is an ideal and $(I] \cap D_r = I$.*

As a result,

Lemma 3. *There exists filters F such that $F \cap D_l$ is a prime filter of the Boolean algebra $\mathcal{D}_l$ and there exists ideals I such that $I \cap D_r$ is a prime ideal of the Boolean algebra $\mathcal{D}_r$.*

We shall say that $\mathcal{D}$ is concrete iff there exists a formal context $\mathbb{K}$ and a function h assigning to each element of D an element of $\mathcal{H}(\mathbb{K})$ such that h is injective and h is a homomorphism from $\mathcal{D}$ to $\underline{\mathcal{H}}(\mathbb{K})$.

3.2 Representation

Now, the main question is to prove that every pure double Boolean algebra is concrete. Let $\mathcal{D} = (D, \bot_l, \bot_r, \top_l, \top_r, \neg_l, \neg_r, \vee_l, \vee_r, \wedge_l, \wedge_r)$ be a pure double Boolean algebra and consider the formal context

$$\mathbb{K}(D) = (\mathcal{F}_p(D), \mathcal{I}_p(D), \Delta)$$

where $\mathcal{F}_p(D)$ is the set of all filters F for which $F \cap D_l$ is a prime filter of the Boolean algebra $\mathcal{D}_l$, $\mathcal{I}_p(D)$ is the set of all ideals I for which $I \cap D_r$ is a prime ideal of the Boolean algebra $\mathcal{D}_r$ and $F \mathrel{\Delta} I$ iff $F \cap I$ is nonempty. Let

$$\underline{\mathcal{H}}(\mathbb{IK}(D)) = (\mathcal{H}(\mathbb{IK}(D)), \bot'_l, \bot'_r, \top'_l, \top'_r, \neg'_l, \neg'_r, \vee'_l, \vee'_r, \wedge'_l, \wedge'_r)$$

For all elements x of D, let

$$\mathcal{F}_x = \{F \in \mathcal{F}_p(D) \colon x \in F\}$$
$$\mathcal{I}_x = \{I \in \mathcal{I}_p(D) \colon x \in I\}$$

Here, the first results are

Lemma 4. *Let $x \in D$. $\mathcal{F}_{x \wedge_l x} = \mathcal{F}_x$ and $\mathcal{I}_{x \vee_r x} = \mathcal{I}_x$.*

Lemma 5. *Let $x \in D$. If $x \in D_l$ then $\mathcal{F}_x^{\triangleright} = \mathcal{I}_x$ and if $x \in D_r$ then $\mathcal{I}_x^{\triangleleft} = \mathcal{F}_x$.*

Lemma 6. *Let $x \in D$. $\mathcal{F}^{\triangleright}_{\neg_l \neg_l x} = \mathcal{I}_{\neg_l \neg_l x}$ and $\mathcal{I}^{\triangleleft}_{\neg_r \neg_r x} = \mathcal{F}_{\neg_r \neg_r x}$.*

The next lemmas point the way to the strategy followed in our approach to the proof that every pure double Boolean algebra is concrete.

Lemma 7. *Let $x \in D$. The pair $(\mathcal{F}_x, \mathcal{I}_x)$ is a semiconcept of $\mathbb{IK}(D)$.*

Lemma 8. *Let $x, y \in D$. If $x \neq y$ then $(\mathcal{F}_x, \mathcal{I}_x) \neq (\mathcal{F}_y, \mathcal{I}_y)$.*

For all $x \in D$, let

$$h(x) = (\mathcal{F}_x, \mathcal{I}_x)$$

The next lemma is central for proving that the function h is a homomorphism from $\mathcal{D}$ to $\underline{\mathcal{H}}(\mathbb{IK})$.

Lemma 9. *Let $x, y \in D$.*

- $\mathcal{F}_{\bot_l} = \emptyset$ *and* $\mathcal{I}_{\bot_l} = \mathcal{I}_p(D)$,
- $\mathcal{F}_{\bot_r} = \mathcal{I}_p(D)^{\triangleleft}$ *and* $\mathcal{I}_{\bot_r} = \mathcal{I}_p(D)$,
- $\mathcal{F}_{\top_l} = \mathcal{F}_p(D)$ *and* $\mathcal{I}_{\top_l} = \mathcal{F}_p(D)^{\triangleright}$,
- $\mathcal{F}_{\top_r} = \mathcal{F}_p(D)$ *and* $\mathcal{I}_{\top_r} = \emptyset$,
- $\mathcal{F}_{\neg_l x} = \mathcal{F}_p(D) \setminus \mathcal{F}_x$ *and* $\mathcal{I}_{\neg_l x} = (\mathcal{F}_p(D) \setminus \mathcal{F}_x)^{\triangleright}$,
- $\mathcal{F}_{\neg_r x} = (\mathcal{I}_p(D) \setminus \mathcal{I}_x)^{\triangleleft}$ *and* $\mathcal{I}_{\neg_r x} = \mathcal{I}_p(D) \setminus \mathcal{I}_x$,
- $\mathcal{F}_{x \vee_l y} = \mathcal{F}_x \cup \mathcal{F}_y$ *and* $\mathcal{I}_{x \vee_l y} = (\mathcal{F}_x \cup \mathcal{F}_y)^{\triangleright}$,
- $\mathcal{F}_{x \vee_r y} = (\mathcal{I}_x \cap \mathcal{I}_y)^{\triangleleft}$ *and* $\mathcal{I}_{x \vee_r y} = \mathcal{I}_x \cap \mathcal{I}_y$,
- $\mathcal{F}_{x \wedge_l y} = \mathcal{F}_x \cap \mathcal{F}_y$ *and* $\mathcal{I}_{x \wedge_l y} = (\mathcal{F}_x \cap \mathcal{F}_y)^{\triangleright}$,
- $\mathcal{F}_{x \wedge_r y} = (\mathcal{I}_x \cup \mathcal{I}_y)^{\triangleleft}$ *and* $\mathcal{I}_{x \wedge_r y} = \mathcal{I}_x \cup \mathcal{I}_y$.

As a result,

Theorem 1. *The function h is a homomorphism from $\mathcal{D}$ to $\underline{\mathcal{H}}(\mathbb{IK})$.*

In other words: every pure double Boolean algebra is concrete.

4 The Word Problem in Pure Double Boolean Algebras

Let us introduce the word problem in pure double Boolean algebras.

4.1 Syntax

Let Var denote a countable set of individual variables (with typical instances denoted x, y, etc). The set $t(Var)$ of all terms (with typical instances denoted s, t, etc) is given by the rule

$$s ::= x \mid 0_l \mid 0_r \mid 1_l \mid 1_r \mid -_l s \mid -_r s \mid (s \sqcup_l t) \mid (s \sqcup_r t) \mid (s \sqcap_l t) \mid (s \sqcap_r t)$$

Let us adopt the standard rules for omission of the parentheses.

Example 7. For instance, $x \sqcap_l (x \sqcup_r y)$ is a term.

4.2 Semantics

Let $\mathcal{D} = (D, \bot_l, \bot_r, \top_l, \top_r, \neg_l, \neg_r, \vee_l, \vee_r, \wedge_l, \wedge_r)$ be a pure double Boolean algebra. A valuation based on $\mathcal{D}$ is a function m assigning to each individual variable x an element $m(x)$ of D.

Example 8. The function $m^{2,2}$ defined below is a valuation based on the pure double Boolean algebra $\mathcal{D}^{2,2}$ defined in Example 5: $m^{2,2}(x) = (\{o_2\}, \{a_1\})$, $m^{2,2}(y) = (\{o_1\}, \{a_2\})$ and for all individual variables z, if $z \neq x, y$ then $m^{2,2}(z) = (\{o_1, o_2\}, \{a_1\})$.

m induces a function $(\cdot)^m$ assigning to each term s an element $(s)^m$ of D such that $(x)^m = m(x)$, $(0_l)^m = \bot_l$, $(0_r)^m = \bot_r$, $(1_l)^m = \top_l$, $(1_r)^m = \top_r$, $(-_l s)^m = \neg_l (s)^m$, $(-_r s)^m = \neg_r (s)^m$, $(s \sqcup_l t)^m = (s)^m \vee_l (t)^m$, $(s \sqcup_r t)^m = (s)^m \vee_r (t)^m$, $(s \sqcap_l t)^m = (s)^m \wedge_l (t)^m$ and $(s \sqcap_r t)^m = (s)^m \wedge_r (t)^m$.

Example 9. Concerning the valuation $m^{2,2}$ defined in Example 8, we have $(x \sqcup_r y)^{m^{2,2}} = (\{o_1, o_2\}, \emptyset)$ and $(-_l x)^{m^{2,2}} = (\{o_1\}, \{a_1, a_2\})$.

4.3 The Word Problem

Now, for the WP in pure double Boolean algebras:

input: terms s, t,
output: determine whether there exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s)^m \neq (t)^m$.

A general strategy for proving a decision problem to be **PSPACE**-complete is first, to reduce to it a decision problem easily proved to be **PSPACE**-hard and second, to reduce it to a decision problem easily proved to be in **PSPACE**. **PSPACE** is the key complexity class of the satisfiability problem of numerous modal logics [1, Chapter 6]. Therefore, we introduce in Section 5 a **PSPACE**-complete modal logic and we show in Sections 6 and 7 how to reduce one into the other its satisfiability problem and the WP in pure double Boolean algebras.

5 A Basic 2-Sorted Modal Logic

In Section 3, we gave the proof that every pure double Boolean algebra can be homomorphically embedded into the pure double Boolean algebra over some formal context. Formal contexts are 2-sorted structures. Hence, the modal logic that will be used in Sections 6 and 7 for proving the WP in pure double Boolean algebras to be **PSPACE**-complete is a 2-sorted one.

5.1 Syntax

The language of K_2 is based on a countable set $OVar$ of object variables (with typical instances denoted P, Q, etc) and a countable set $AVar$ of attribute variables (with typical instances denoted p, q, etc). Without loss of generality, let us assume that $OVar$ and $AVar$ are disjoint. The set of all object formulas (with typical instances denoted A, B, etc) and the set of all attribute formulas (with typical instances denoted a, b, etc) are given by the rules

$$A ::= P \mid \bot \mid \neg A \mid (A \vee B) \mid \Box a$$
$$a ::= p \mid \bot \mid \neg a \mid (a \vee b) \mid \Box A$$

The other Boolean constructs are defined as usual. Let us adopt the standard rules for omission of the parentheses. A formula (with typical instances denoted α, β, etc) is either an object formula or an attribute formula. The notion of "being a subformula of" is standard, the expression $\alpha \ll \beta$ denoting the fact that α is a subformula of β. A substitution is a pair (Θ, θ) where Θ is a function assigning to each object variable P an object formula $\Theta(P)$ and θ is a function assigning to each attribute variable p an attribute formula $\theta(p)$. (Θ, θ) induces a homomorphism $(\cdot)^{(\Theta,\theta)}$ assigning to each formula α a formula $(\alpha)^{(\Theta,\theta)}$ such that $(P)^{(\Theta,\theta)} = \Theta(P)$ and $(p)^{(\Theta,\theta)} = \theta(p)$. Remark that for all object formulas A and for all attribute formulas a,

- $(A)^{(\Theta,\theta)}$ is an object formula,
- $(a)^{(\Theta,\theta)}$ is an attribute formula.

Let $\boldsymbol{OVar} = P_1, P_2, \ldots$ be an enumeration of $OVar$ and $\boldsymbol{AVar} = p_1, p_2, \ldots$ be an enumeration of $AVar$. We shall say that a substitution (Θ, θ) is normal with respect to $\boldsymbol{OVar}$ and $\boldsymbol{AVar}$ iff for all positive integers i,

- $\Theta(P_i) = P_i$ and $\theta(p_i) = \Box P_i$ or $\Theta(P_i) = \Box p_i$ and $\theta(p_i) = p_i$.

Given a formula α, $Var(\alpha)$ will denote the set of all variables occurring in α. A formula α is said to be nice iff

- $Var(\alpha) \subseteq OVar$ or $Var(\alpha) \subseteq AVar$.

5.2 Semantics

Let $\mathbb{IK} = (G, M, \Delta)$ be a formal context. A $\mathbb{IK}$-valuation is a pair (V, v) of functions where V assigns to each object variable P a subset $V(P)$ of G and v assigns to each attribute variable p a subset $v(p)$ of M. (V, v) induces a function $(\cdot)^{(V,v)}$ assigning to each formula α a subset $(\alpha)^{(V,v)}$ of $G \cup M$ such that $(P)^{(V,v)} = V(P)$, $(\bot)^{(V,v)} = \emptyset$, $(\neg A)^{(V,v)} = G \setminus (A)^{(V,v)}$, $(A \vee B)^{(V,v)} = (A)^{(V,v)} \cup (B)^{(V,v)}$, $(\Box a)^{(V,v)} = \{g \in G$: for all $m \in M$, if $m \in (a)^{(V,v)}$ then $g \mathrel{\Delta} m\}$, $(p)^{(V,v)} = v(p)$, $(\bot)^{(V,v)} = \emptyset$, $(\neg a)^{(V,v)} = M \setminus (a)^{(V,v)}$, $(a \vee b)^{(V,v)} = (a)^{(V,v)} \cup (b)^{(V,v)}$ and $(\Box A)^{(V,v)} = \{m \in M$: for all $g \in G$, if $g \in (A)^{(V,v)}$ then $g \mathrel{\Delta} m\}$. Remark that for all object formulas A and for all attribute formulas a,

- $(A)^{(V,v)}$ is a subset of G such that $(A)^{(V,v)^{\triangleright}} = (\Box A)^{(V,v)}$,
- $(a)^{(V,v)}$ is a subset of M such that $(a)^{(V,v)^{\triangleleft}} = (\Box a)^{(V,v)}$.

A formula α is said to be satisfiable iff

- there exists a formal context $\mathbb{IK} = (G, M, \Delta)$ and a $\mathbb{IK}$-valuation (V, v) such that $(\alpha)^{(V,v)}$ is nonempty.

5.3 Decision

Now, for the nice satisfiability problem for K_2:

input: a nice formula α,
output: determine whether α is satisfiable.

The next lemmas are central for proving that the problem of deciding equations in pure double Boolean algebras is **PSPACE**-complete.

Theorem 2. *The nice satisfiability problem for K_2 is* **PSPACE***-hard.*

Proof. A reduction similar to the reduction from the QBF-validity problem to the satisfiability problem for K considered in [1, Theorem 6.50] can be easily obtained.

Now, for the satisfiability problem for K_2:

input: a formula α,
output: determine whether α is satisfiable.

Theorem 3. *The satisfiability problem for K_2 is in* **PSPACE**.

Proof. An algorithm similar to the $Witness$ algorithm considered in [1, Theorem 6.47] can be easily obtained.

From Theorems 2 and 3, it follows immediately that the nice satisfiability problem for K_2 and the satisfiability problem for K_2 are both **PSPACE**-complete.

6 From K_2 to Pure Double Boolean Algebras

First, we consider the lower bound of the complexity of the problem of deciding the WP in pure double Boolean algebras. Given a nice formula α, we wish to construct a pair $(s_1(\alpha), s_2(\alpha))$ of terms such that α is satisfiable iff there exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s_1(\alpha))^m \neq (s_2(\alpha))^m$. Let $\boldsymbol{OVar} = P_1, P_2, \ldots$ be an enumeration of $OVar$, $\boldsymbol{AVar} = p_1, p_2, \ldots$ be an enumeration of $AVar$ and $\boldsymbol{Var} = x_1, y_1, x_2, y_2, \ldots$ be an enumeration of Var. The function $T(\cdot)$ assigning to each nice object formula A a term $T(A)$ and the function $t(\cdot)$ assigning to each nice attribute formula a a term $t(a)$ are such that $T(P_i) = x_i$, $T(\bot) = 0_l$, $T(\neg A) = -_l T(A)$, $T(A \vee B) = T(A) \sqcup_l T(B)$, $T(\Box a) = -_l -_l -_r -_r t(a)$, $t(p_i) = y_i$, $t(\bot) = 1_r$, $t(\neg a) = -_r t(a)$, $t(a \vee b) = t(a) \sqcap_r t(b)$ and $t(\Box A) = -_r -_r -_l -_l T(A)$. Let $(s_1(\cdot), s_2(\cdot))$ be the function assigning to each nice formula α a pair $(s_1(\alpha), s_2(\alpha))$ of terms such that if α is a nice object formula then $s_1(\alpha) = T(\alpha)$ and $s_2(\alpha) = 0_l$ and if α is a nice attribute formula then $s_1(\alpha) = t(\alpha)$ and $s_2(\alpha) = 1_r$. Obviously, $(s_1(\alpha), s_2(\alpha))$ can be computed in space $\log \mid \alpha \mid$. Moreover,

Proposition 1. *If α is nice then α is satisfiable iff there exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s_1(\alpha))^m \neq (s_2(\alpha))^m$.*

Proof. Since α is nice, $Var(\alpha) \subseteq OVar$ or $Var(\alpha) \subseteq AVar$. Without loss of generality, let us assume that $Var(\alpha) \subseteq OVar$. Hence, there exists a positive integer n such that $Var(\alpha) \subseteq \{P_1, \ldots, P_n\}$.
($\Rightarrow$) Suppose α is satisfiable, we demonstrate there exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s_1(\alpha))^m \neq (s_2(\alpha))^m$. Since α is satisfiable, there exists a formal context $\mathbb{K} = (G, M, \Delta)$ and a valuation (V, v) based on $\mathbb{K}$ such that $(\alpha)^{(V,v)}$ is nonempty. Let $\underline{\mathcal{H}}(\mathbb{K}) = (\mathcal{H}(\mathbb{K}), \bot_l, \bot_r, \top_l, \top_r, \neg_l, \neg_r, \vee_l, \vee_r, \wedge_l, \wedge_r)$ and m be a valuation based on $\underline{\mathcal{H}}(\mathbb{K})$ such that for all positive integers i, if $i \leq n$ then $m(x_i) = (V(P_i), V(P_i)^{\triangleright})$. We show first that

Lemma 10. *Let A be a nice object formula and a be a nice attribute formula. If $A \ll \alpha$ then $(T(A))^m = ((A)^{(V,v)}, (A)^{(V,v)^{\triangleright}})$ and if $a \ll \alpha$ then $(t(a))^m = ((a)^{(V,v)^{\triangleleft}}, (a)^{(V,v)})$.*

Continuing the proof of Proposition 1, since $(\alpha)^{(V,v)}$ is nonempty, by Lemma 10, if α is a nice object formula then $(T(\alpha))^m \neq (0_l)^m$ and if α is a nice attribute formula then $(t(\alpha))^m \neq (1_r)^m$. Hence, $(s_1(\alpha))^m \neq (s_2(\alpha))^m$. Thus, there exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s_1(\alpha))^m \neq (s_2(\alpha))^m$.
($\Leftarrow$) Suppose there exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s_1(\alpha))^m \neq (s_2(\alpha))^m$, we demonstrate α is satisfiable. Let $\mathbb{K}(D) = (\mathcal{F}_p(D), \mathcal{I}_p(D), \Delta)$ and (V, v) be a valuation based on $\mathbb{K}(D)$ such that for all positive integers i, if $i \leq n$ then $V(P_i) = \mathcal{F}_{m(x_i)}$. Interestingly,

Lemma 11. *Let A be a nice object formula and a be a nice attribute formula. If $A \ll \alpha$ then $(A)^{(V,v)} = \mathcal{F}_{(T(A))^m}$ and if $a \ll \alpha$ then $(a)^{(V,v)} = \mathcal{I}_{(t(a))^m}$.*

Continuing the proof of Proposition 1, since $(s_1(\alpha))^m \neq (s_2(\alpha))^m$, if α is a nice object formula then $(T(\alpha))^m \neq (0_l)^m$ and if α is a nice attribute formula then $(t(\alpha))^m \neq (1_r)^m$. Hence, by Lemma 11, $(\alpha)^{(V,v)}$ is nonempty. Thus, α is satisfiable. This ends the proof of Proposition 1.

Hence, $(s_1(\cdot), s_2(\cdot))$ is a reduction from the nice satisfiability problem for K_2 to the WP in pure double Boolean algebras. Thus, by Theorem 2,

Corollary 1. *The WP in pure double Boolean algebras is* **PSPACE***-hard.*

7 From Pure Double Boolean Algebras to K_2

Second, we consider the upper bound of the complexity of the WP in pure double Boolean algebras. Given a pair (s,t) of terms, we wish to construct an object formula $O(s,t)$ and an attribute formula $A(s,t)$ such that there exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s)^m \neq (t)^m$ iff some instance of $O(s,t)$ is satisfiable or some instance of $A(s,t)$ is satisfiable. Let $\boldsymbol{Var} = x_1, x_2, \ldots$ be an enumeration of Var, $\boldsymbol{OVar} = P_1, P_2, \ldots$ be an enumeration of $OVar$ and $\boldsymbol{AVar} = p_1, p_2, \ldots$ be an enumeration of $AVar$. The function $F(\cdot)$ assigning to each term s an object formula $F(s)$ and the function $f(\cdot)$ assigning to each term s an attribute formula $f(s)$ are such that $F(x_i) = P_i$, $f(x_i) = p_i$, $F(0_l) = \bot$, $f(0_l) = \Box\bot$, $F(0_r) = \Box\top$, $f(0_r) = \top$, $F(1_l) = \top$, $f(1_l) = \Box\top$, $F(1_r) = \Box\bot$, $f(1_r) = \bot$, $F(-_l s) = \neg F(s)$, $f(-_l s) = \Box\neg F(s)$, $F(-_r s) = \Box\neg f(s)$, $f(-_r s) = \neg f(s)$, $F(s \sqcup_l t) = F(s) \vee F(t)$, $f(s \sqcup_l t) = \Box(F(s) \vee F(t))$, $F(s \sqcup_r t) = \Box(f(s) \wedge f(t))$, $f(s \sqcup_r t) = f(s) \wedge f(t)$, $F(s \sqcap_l t) = F(s) \wedge F(t)$, $f(s \sqcap_l t) = \Box(F(s) \wedge F(t))$, $F(s \sqcap_r t) = \Box(f(s) \vee f(t))$ and $f(s \sqcap_r t) = f(s) \vee f(t)$. Let $O(\cdot,\cdot)$ be the function assigning to each pair (s,t) of terms the object formula $O(s,t)$ such that $O(s,t) = \neg(F(s) \leftrightarrow F(t))$. Let $A(\cdot,\cdot)$ be the function assigning to each pair (s,t) of terms the attribute formula $A(s,t)$ such that $A(s,t) = \neg(f(s) \leftrightarrow f(t))$. Obviously, $O(s,t)$ and $A(s,t)$ can be computed in space $\log \mid (s,t) \mid$. Moreover,

Proposition 2. *There exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s)^m \neq (t)^m$ iff there exists a substitution (Θ, θ) such that (Θ, θ) is normal with respect to $\boldsymbol{OVar}$ and $\boldsymbol{AVar}$ and $O(s,t)^{(\Theta,\theta)}$ is satisfiable or $A(s,t)^{(\Theta,\theta)}$ is satisfiable.*

Proof. Let n be a positive integer such that $Var(s) \cup Var(t) \subseteq \{x_1, \ldots, x_n\}$.
($\Rightarrow$) Suppose there exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s)^m \neq (t)^m$, we demonstrate there exists a substitution (Θ, θ) such that (Θ, θ) is normal with respect to $\boldsymbol{OVar}$ and $\boldsymbol{AVar}$ and $O(s,t)^{(\Theta,\theta)}$ is satisfiable or $A(s,t)^{(\Theta,\theta)}$ is satisfiable. Let (Θ, θ) be a normal substitution with respect to $\boldsymbol{OVar}$ and $\boldsymbol{AVar}$ such that for all positive integers i, if $i \leq n$ then if $m(x_i)$ is in D_l then $\Theta(P_i) = P_i$ and $\theta(p_i) = \Box P_i$ and if $m(x_i)$ is in D_r then $\Theta(P_i) = \Box p_i$ and $\theta(p_i) = p_i$. Let $\mathbb{IK}(D) = (\mathcal{F}_p(D), \mathcal{I}_p(D), \Delta)$ and (V, v) be a valuation based on $\mathbb{IK}(D)$ such that for all positive integers i, if $i \leq n$ then $V(P_i) = \mathcal{F}_{m(x_i)}$ and $v(p_i) = \mathcal{I}_{m(x_i)}$. Remark that for all positive integers i,

if $i \leq n$ then if $m(x_i)$ is in D_l then $(P_i)^{(\Theta,\theta)^{(V,v)}} = (P_i)^{(V,v)} = V(P_i) = \mathcal{F}_{m(x_i)}$ and if $m(x_i)$ is in D_r then $(P_i)^{(\Theta,\theta)^{(V,v)}} = (\Box p_i)^{(V,v)} = (p_i)^{(V,v)^{\triangleleft}} = v(p_i)^{\triangleleft} = \mathcal{I}^{\triangleleft}_{m(x_i)} = \mathcal{F}_{m(x_i)}$. Similarly, for all positive integers i, if $i \leq n$ then if $m(x_i)$ is in D_l then $(p_i)^{(\Theta,\theta)^{(V,v)}} = (\Box P_i)^{(V,v)} = (P_i)^{(V,v)^{\triangleright}} = V(P_i)^{\triangleright} = \mathcal{F}^{\triangleright}_{m(x_i)} = \mathcal{I}_{m(x_i)}$ and if $m(x_i)$ is in D_r then $(p_i)^{(\Theta,\theta)^{(V,v)}} = (p_i)^{(V,v)} = v(p_i) = \mathcal{I}_{m(x_i)}$ We first observe

Lemma 12. *Let u be a term. If $u \ll s$ or $u \ll t$ then $(F(u))^{(\Theta,\theta)^{(V,v)}} = \mathcal{F}_{(u)^m}$ and $(f(u))^{(\Theta,\theta)^{(V,v)}} = \mathcal{I}_{(u)^m}$.*

Continuing the proof of Proposition 2, since $(s)^m \neq (t)^m$, $\mathcal{F}_{(s)^m} \neq \mathcal{F}_{(t)^m}$ or $\mathcal{I}_{(s)^m} \neq \mathcal{I}_{(t)^m}$. Hence, by Lemma 12, $O(s,t)^{(\Theta,\theta)^{(V,v)}}$ is nonempty or $A(s,t)^{(\Theta,\theta)^{(V,v)}}$ is nonempty. Thus, there exists a substitution (Θ,θ) such that (Θ,θ) is normal with respect to $\boldsymbol{OVar}$ and $\boldsymbol{AVar}$ and $O(s,t)^{(\Theta,\theta)}$ is satisfiable or $A(s,t)^{(\Theta,\theta)}$ is satisfiable.
($\Leftarrow$) Suppose there exists a substitution (Θ,θ) such that (Θ,θ) is normal with respect to $\boldsymbol{OVar}$ and $\boldsymbol{AVar}$ and $O(s,t)^{(\Theta,\theta)}$ is satisfiable or $A(s,t)^{(\Theta,\theta)}$ is satisfiable, we demonstrate there exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s)^m \neq (t)^m$. Since $O(s,t)^{(\Theta,\theta)}$ is satisfiable or $A(s,t)^{(\Theta,\theta)}$ is satisfiable, there exists a formal context $\mathbb{K} = (G, M, \Delta)$ and a valuation (V,v) based on $\mathbb{K}$ such that $O(s,t)^{(\Theta,\theta)^{(V,v)}}$ is nonempty or $A(s,t)^{(\Theta,\theta)^{(V,v)}}$ is nonempty. Let $\underline{\mathcal{H}}(\mathbb{K}) = (\mathcal{H}(\mathbb{K}), \bot_l, \bot_r, \top_l, \top_r, \neg_l, \neg_r, \vee_l, \vee_r, \wedge_l, \wedge_r)$ and m be a valuation based on $\underline{\mathcal{H}}(\mathbb{K})$ such that for all positive integers i, if $i \leq n$ then $m(x_i) = ((\Theta(P_i))^{(V,v)}, (\theta(p_i))^{(V,v)})$. Interestingly,

Lemma 13. *Let u be a term. If $u \ll s$ or $u \ll t$ then $(u)^m = ((F(u))^{(\Theta,\theta)^{(V,v)}}, (f(u))^{(\Theta,\theta)^{(V,v)}})$.*

Continuing the proof of Proposition 2, since $O(s,t)^{(\Theta,\theta)^{(V,v)}}$ is nonempty or $A(s,t)^{(\Theta,\theta)^{(V,v)}}$ is nonempty, $F(s)^{(\Theta,\theta)^{(V,v)}} \neq F(t)^{(\Theta,\theta)^{(V,v)}}$ or $f(s)^{(\Theta,\theta)^{(V,v)}} \neq f(t)^{(\Theta,\theta)^{(V,v)}}$. Hence, by lemma 13, $(s)^m \neq (t)^m$. Thus, there exists a pure double Boolean algebra $\mathcal{D}$ and a valuation m based on $\mathcal{D}$ such that $(s)^m \neq (t)^m$. This ends the proof of Proposition 2.

Hence, $O(\cdot,\cdot)$ and $A(\cdot,\cdot)$ are reductions from the WP in pure double Boolean algebras to the satisfiability problem for K_2. Thus, by Theorem 3,

Corollary 2. *The WP in pure double Boolean algebras is in* **PSPACE**.

8 Conclusion

Our results implicitly assume that the set Var of all individual variables is infinite and the depth of nesting of the left operations with the right operations is not bounded. Following the line of reasoning suggested in [4], we may see what

happens if we assume that the set Var of all individual variables is finite and the depth of nesting of the left operations with the right operations is bounded. Do we get a linear time complexity in this case?
The unification problem is quite different from the WP discussed here: given terms s, t, decide whether there exists terms which can be substituted for the variables in s, t so that the terms thus obtained are identically interpreted in all pure double Boolean algebras. In Mathematics and Computer Science, unification problems are of the utmost importance. At the time of writing, we know nothing about the decidability/complexity of the unification problem in pure double Boolean algebras.

Acknowledgements

Special acknowledgement is heartly granted to Christian Herrmann who made several helpful comments for improving the correctness and the readability of this article.

References

1. Blackburn, P., de Rijke, M., Venema, Y.: *Modal Logic.* Cambridge University Press (2001).
2. Davey, B, Priestley, H.: *Introduction to Lattices and Order.* Cambridge University Press (2002).
3. Ganter, B, Wille, R.: *Formal Concept Analysis: Mathematical Foundations.* Springer (1999).
4. Halpern, J.: *The effect of bounding the number of primitive propositions and the depth of nesting on the complexity of modal logic.* Artificial Intelligence **75** (1995) 361–372.
5. Herrmann, C., Luksch, P., Skorsky, M., Wille, R.: *Algebras of semiconcepts and double Boolean algebras.* Technische Universität Darmstadt (2000).
6. Vormbrock, B.: *A first step towards protoconcept exploration.* In Eklund, P. (editor): *Concept Lattices.* Springer (2004) 208–221.
7. Vormbrock, B.: *Complete subalgebras of semiconcept algebras and protoconcept algebras.* In Ganter, B., Godin, R. (editors): *Formal Concept Analysis.* Springer (2005) 329–343.
8. Vormbrock, B.: *A solution of the word problem for free double Boolean algebras.* In Kuznetsov, S., Schmidt, S. (editors): *Formal Concept Analysis.* Springer (2007) 240–270.
9. Vormbrock, B., Wille, R.: *Semiconcept and protoconcept algebras: the basic theorems.* In Ganter, B., Stumme, G., Wille, R. (editors): *Formal Concept Analysis.* Springer (2005) 34–48.
10. Wille, R.: *Restructuring lattice theory: an approach based on hierarchies of concepts.* In Rival, I. (editor): *Ordered Sets.* D. Reidel (1982) 314–339
11. Wille, R.: *Boolean concept logic.* In Ganter, B., Mineau, G. (editors): *Conceptual Structures: Logical, Linguistic, and Computational Issues.* Springer (2000) 317–331.
12. Wille, R.: *Formal concept analysis as applied lattice theory.* In Ben Yahia, S., Mephu Nguifo, E., Belohlavek, R. (editors): *Concept Lattices and their Applications.* Springer (2008) 42–67

Annex

Proof of Lemma 10. By induction on A and a.
Basis. Remind that $Var(\alpha) \subseteq \{P_1, \ldots, P_n\}$. In this respect, for all positive integers i, if $i \leq n$ then $(T(P_i))^m = (x_i)^m = m(x_i) = (V(P_i), V(P_i)^{\triangleright}) = ((P_i)^{(V,v)}, (P_i)^{(V,v)^{\triangleright}})$.
Hypothesis. Suppose A, B are nice object formulas such that $A, B \ll \alpha$, $(T(A))^m = ((A)^{(V,v)}, (A)^{(V,v)^{\triangleright}})$ and $(T(B))^m = ((B)^{(V,v)}, (B)^{(V,v)^{\triangleright}})$ and a, b are nice attribute formulas such that $a, b \ll \alpha$, $(t(a))^m = ((a)^{(V,v)^{\triangleleft}}, (a)^{(V,v)})$ and $(t(b))^m = ((b)^{(V,v)^{\triangleleft}}, (b)^{(V,v)})$.
Step. We only consider the case of the nice object formula $\Box a$, the other cases being treated similarly. We have: $(T(\Box a))^m = (-_l -_l -_r -_r t(a))^m = \neg_l\neg_l\neg_r\neg_r(t(a))^m = \neg_l\neg_l\neg_r\neg_r((a)^{(V,v)^{\triangleleft}}, (a)^{(V,v)}) = \neg_l\neg_l(((a)^{(V,v)})^{\triangleleft}, (a)^{(V,v)}) = (((a)^{(V,v)})^{\triangleleft}, ((a)^{(V,v)})^{\triangleleft\triangleright}) = ((\Box a)^{(V,v)}, (\Box a)^{(V,v)^{\triangleright}})$.

Proof of Lemma 11. By induction on A and a.
Basis. Remind that $Var(\alpha) \subseteq \{P_1, \ldots, P_n\}$. In this respect, for all positive integers i, if $i \leq n$ then $(P_i)^{(V,v)} = V(P_i) = \mathcal{F}_{m(x_i)} = \mathcal{F}_{(x_i)^m} = \mathcal{F}_{(T(P_i))^m}$.
Hypothesis. Suppose A, B are nice object formulas such that $A, B \ll \alpha$, $(A)^{(V,v)} = \mathcal{F}_{(T(A))^m}$ and $(B)^{(V,v)} = \mathcal{F}_{(T(B))^m}$ and a, b are nice attribute formulas such that $a, b \ll \alpha$, $(a)^{(V,v)} = \mathcal{I}_{(t(a))^m}$ and $(b)^{(V,v)} = \mathcal{I}_{(t(b))^m}$.
Step. We only consider the case of the nice object formula $\Box a$, the other cases being treated similarly. We have: $(\Box a)^{(V,v)} = \{F \in \mathcal{F}_p(D)$: for all $I \in \mathcal{I}_p(D)$, if $I \in (a)^{(V,v)}$ then $F \mathrel{\Delta} I\} = \{F \in \mathcal{F}_p(D)$: for all $I \in \mathcal{I}_p(D)$, if $I \in \mathcal{I}_{(t(a))^m}$ then $F \mathrel{\Delta} I\} = \mathcal{I}_{(t(a))^m}{}^{\triangleleft} = \mathcal{F}_{\neg_r\neg_r(t(a))^m} = \mathcal{F}_{\neg_l\neg_l\neg_r\neg_r(t(a))^m} = \mathcal{F}_{(-_l-_l-_r-_r t(a))^m} = \mathcal{F}_{(T(\Box a))^m}$.

Proof of Lemma 12. By induction on u.
Basis. Remind that $Var(s) \cup Var(t) \subseteq \{x_1, \ldots, x_n\}$. In this respect, for all positive integers i, if $i \leq n$ then $(F(x_i))^{(\Theta,\theta)^{(V,v)}} = (P_i)^{(\Theta,\theta)^{(V,v)}} = \mathcal{F}_{m(x_i)} = \mathcal{F}_{(x_i)^m}$ and $(f(x_i))^{(\Theta,\theta)^{(V,v)}} = (p_i)^{(\Theta,\theta)^{(V,v)}} = \mathcal{I}_{m(x_i)} = \mathcal{I}_{(x_i)^m}$.
Hypothesis. Suppose u, v are terms such that $u \ll s$ or $u \ll t$, $v \ll s$ or $v \ll t$, $(F(u))^{(\Theta,\theta)^{(V,v)}} = \mathcal{F}_{(u)^m}$, $(f(u))^{(\Theta,\theta)^{(V,v)}} = \mathcal{I}_{(u)^m}$, $(F(v))^{(\Theta,\theta)^{(V,v)}} = \mathcal{F}_{(v)^m}$ and $(f(v))^{(\Theta,\theta)^{(V,v)}} = \mathcal{I}_{(v)^m}$.
Step. We only consider the case of the term $u \sqcap_l v$, the other cases being treated similarly. We have: $(F(u \sqcap_l v))^{(\Theta,\theta)^{(V,v)}} = (F(u) \wedge F(v))^{(\Theta,\theta)^{(V,v)}} = ((F(u))^{(\Theta,\theta)} \wedge (F(v))^{(\Theta,\theta)})^{(V,v)} = (F(u))^{(\Theta,\theta)^{(V,v)}} \cap (F(v))^{(\Theta,\theta)^{(V,v)}} = \mathcal{F}_{(u)^m} \cap \mathcal{F}_{(v)^m} = \mathcal{F}_{(u)^m \wedge_l (v)^m} = \mathcal{F}_{(u \sqcap_l v)^m}$ and $(f(u \sqcap_l v))^{(\Theta,\theta)^{(V,v)}} = (\Box(F(u) \wedge F(v)))^{(\Theta,\theta)^{(V,v)}} = (\Box((F(u))^{(\Theta,\theta)} \wedge (F(v))^{(\Theta,\theta)}))^{(V,v)} = ((F(u))^{(\Theta,\theta)} \wedge (F(v))^{(\Theta,\theta)})^{(V,v)^{\triangleright}} = ((F(u))^{(\Theta,\theta)^{(V,v)}} \cap (F(v))^{(\Theta,\theta)^{(V,v)}})^{\triangleright} = (\mathcal{F}_{(u)^m} \cap \mathcal{F}_{(v)^m})^{\triangleright} = \mathcal{I}_{(u)^m \wedge_l (v)^m} = \mathcal{I}_{(u \sqcap_l v)^m}$.

Proof of Lemma 13. By induction on u.
Basis. Remind that $Var(s) \cup Var(t) \subseteq \{x_1, \ldots, x_n\}$. In this respect, for all

positive integers i, if $i \leq n$ then $(x_i)^m = m(x_i) = ((\Theta(P_i))^{(V,v)}, (\theta(p_i))^{(V,v)}) = ((P_i)^{(\Theta,\theta)^{(V,v)}}, (p_i)^{(\Theta,\theta)^{(V,v)}}) = ((F(x_i))^{(\Theta,\theta)^{(V,v)}}, (f(x_i))^{(\Theta,\theta)^{(V,v)}})$.

Hypothesis. Suppose u, v are terms such that $u \ll s$ or $u \ll t$, $v \ll s$ or $v \ll t$, $(u)^m = ((F(u))^{(\Theta,\theta)^{(V,v)}}, (f(u))^{(\Theta,\theta)^{(V,v)}})$ and $(v)^m = ((F(v))^{(\Theta,\theta)^{(V,v)}}, (f(v))^{(\Theta,\theta)^{(V,v)}})$.

Step. We only consider the case of the term $u \sqcap_l v$, the other cases being treated similarly. We have: $(u \sqcap_l v)^m = (u)^m \wedge_l (v)^m = ((F(u))^{(\Theta,\theta)^{(V,v)}}, (f(u))^{(\Theta,\theta)^{(V,v)}}) \wedge_l ((F(v))^{(\Theta,\theta)^{(V,v)}}, (f(v))^{(\Theta,\theta)^{(V,v)}}) = ((F(u))^{(\Theta,\theta)^{(V,v)}} \cap (F(v))^{(\Theta,\theta)^{(V,v)}}, ((F(u))^{(\Theta,\theta)^{(V,v)}} \cap (F(v))^{(\Theta,\theta)^{(V,v)}})^{\triangleright}) = (((F(u))^{(\Theta,\theta)} \wedge (F(v))^{(\Theta,\theta)})^{(V,v)}, ((F(u))^{(\Theta,\theta)} \wedge (F(v))^{(\Theta,\theta)})^{(V,v)^{\triangleright}}) = ((F(u) \wedge (F(v))^{(\Theta,\theta)^{(V,v)}}, (\Box((F(u))^{(\Theta,\theta)} \wedge (F(v))^{(\Theta,\theta)}))^{(V,v)}) = ((F(u \sqcap_l v))^{(\Theta,\theta)^{(V,v)}}, (\Box(F(u) \wedge F(v)))^{(\Theta,\theta)^{(V,v)}}) = ((F(u \sqcap_l v))^{(\Theta,\theta)^{(V,v)}}, (f(u \sqcap_l v))^{(\Theta,\theta)^{(V,v)}})$.

Looking for analogical proportions in a formal concept analysis setting

Laurent Miclet[1], Henri Prade[2], and David Guennec[1]

[1] IRISA-ENSSAT, Lannion, France, miclet@enssat.fr, david.guennec@gmail.com,
[2] IRIT, Université Paul Sabatier, Toulouse, France, prade@irit.fr

Abstract. Categorization and analogical reasoning are two important cognitive processes, for which there exist formal counterparts (at least they may be regarded as such): namely, formal concept analysis on the one hand, and analogical proportions (modeled in propositional logic) on the other hand. This is a first attempt aiming at relating these two settings. The paper presents an algorithm that takes advantage of the lattice structure of the set of formal concepts for searching for analogical proportions that may hold in a formal context. Moreover, properties linking analogical proportions and formal concepts are laid bare.

1 Introduction

Categorization and analogical reasoning play important roles in cognitive processes. They both heavily rely on the ideas of similarity and dissimilarity. Items belonging to the same category should be similar, while they are dissimilar with respect to items belonging to other categories. Analogical proportions, which are statements of the form 'a is to b as c is to d', express the similarity of the relations linking a and b with the relations linking c and d (note that however a and b may be somewhat dissimilar (as well as c and d). In a Boolean setting, where items are described in terms of binary attributes, similarity amounts to the identity of properties, while dissimilarity refers to the presence of properties for an item which are absent in the other considered item.

Among formal approaches aiming at categorizing items, Formal Concept Analysis (FCA) provides a way for characterizing concepts both extensionally in terms of the objects that they cover and intensionally in terms of the properties that these objects share. FCA is known as a lattice-theoretic framework devised for knowledge extraction from Boolean data tables called *formal contexts* that relate objects and properties. Introduced under this name by Wille [13], FCA has been developed by Ganter and Wille [7] and their followers for thirty years.

Besides, there has been a renewal of interest for analogical proportions in the last decade, firstly in relation with computational linguistic concerns. Set-based, algebraic and logical models have been proposed [8, 12, 1, 9]. In the following, we more particularly use the Boolean view [9] of analogical proportions that is directly relevant for application to formal contexts. Then, it makes sense to look for analogical proportions in Boolean contexts, and to try to understand what formal concepts and analogical proportions may have in common.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

The paper is organized as follows. We first provide a short background on analogical proportions in Section 2. Then in Section 3, after a brief reminder of basic definitions in FCA, we present an efficient algorithm able to discover analogical proportions in a formal context by using the lattice of formal concepts. In Section 4, we further investigate the theoretical relations between FCA and analogical proportions, by showing how formal concepts are involved in analogical proportions, before indicating lines for further research and concluding.

2 Analogical proportions

An analogical proportion 'a is to b as c is to d', usually denoted a : b :: c : d, expresses that the way a and b differ is the same as the way c and $d$differ [9]. This leads to the following definitions, here stated for three closely related kinds of items: subsets of a finite set, Boolean truth values, and objects defined by Boolean properties (also called binary *attributes*).

Analogical proportion between sets First, let us consider four sets A, B, C and D, all subsets of some set X. The dissimilarity between A and B is evaluated by $A \cap \overline{B}$ and by $\overline{A} \cap B$, where $\overline{A}$ denotes the complement of A in X, while the similarity corresponds to $A \cap B$ and $\overline{A} \cap \overline{B}$. Viewing an analogical proportion as expressing that the differences between A and B and between C and D are the same, we get the following definition [9]:

Definition 1 *Four subsets A, B, C and D of a finite set X are in analogical proportion in this order when $A \cap \overline{B} = C \cap \overline{D}$ and $\overline{A} \cap B = \overline{C} \cap D$.*

Analogical proportion between Boolean objects This expression has an immediate logical counterpart when a, b, c, and d now denote Boolean variables:

$$((a \wedge \neg b) \equiv (c \wedge \neg d)) \wedge ((\neg a \wedge b) \equiv (\neg c \wedge d))$$

This formula is true for the 6 truth value assignments of a, b, c, d appearing in Table 1, and is false for the $2^4 - 6 = 10$ remaining possible assignments.

It can be checked that the above definitions of an analogical proportion satisfies the following characteristic postulates [8]:

- $a : b :: a : b$ (identity)
- $a : b :: c : d \implies c : d :: a : b$ (global symmetry)
- $a : b :: c : d \implies a : c :: b : d$ (central permutation)
- $a : b :: c : d$ and $\neg(b : a :: c : d)$ are consistent (local dissymmetry)

a	×			×		×
b	×		×			×
c	×			×	×	
d	×		×		×	

Table 1. The six Boolean 4-tuples that are in analogical proportion. The Boolean truth-values True and False are written as a cross and a blank.

Objects defined by Boolean properties Let us suppose that the objects (or *items*) a, b, c, d are described by sets of binary properties belonging to a set *Prop*. Then, each item can be viewed as a subset of *Prop*, made of the attributes that hold true on this item[3]. Then, we can apply Definition 1, namely $a \cap \overline{b} = c \cap \overline{d}$ and $\overline{a} \cap b = \overline{c} \cap d$. Another way of seeing this analogical proportion is given by the equivalent definition:

Definition 2 *Four objects* (a, b, c, d) *defined by binary properties are in analogical proportion iff the truth-values of each property on these objects make a 4-tuple of binary values corresponding to one of the six 4-tuples displayed in Table 1.*

Analogical dissimilarity We now introduce the concept of *analogical dissimilarity (AD)* between four objects defined by binary attributes. It is simply the sum on all the attributes of the analogical dissimilarity per attribute. The latter is defined according to the following table:

a								×	×	×	×	×	×	×	×
b				×	×	×	×					×	×	×	×
c		×	×			×	×			×	×			×	×
d	×		×		×		×		×		×		×		×
$AD =$	0	1	1	0	1	0	2	1	1	2	0	1	0	1	1	0

AD per attribute is merely the minimal number of bit(s) that has/have to be flipped in order to turn the four bits into an analogical proportion, according to Table 1. Notice that any 4-tuple with a zero AD is an analogical proportion, and vice-versa. For example, the four first objects of the formal context that we will later call $\mathcal{BASE}_{lm}$ (see Figure 2) are such that $AD(\text{leech}, \text{bream}, \text{frog}, \text{dog}) = 0+1+0+0+0+0+0+1+1 = 3$.

3 Searching for analogical proportions in formal concepts

This section is devoted to the following problem: given a formal context with n objects and d properties (or attributes), is it possible to discover 4-tuples of objects in analogical proportion, without running an $\mathcal{O}(n^4 \cdot d)$ algorithm? We give in this section an heuristic algorithm which uses the lattice of formal concepts, and has shown experimentally its efficiency for discovering analogical proportions. We start with a brief reminder on FCA.

[3]For an object x, this subset is called $R^{\uparrow}(x)$ in Formal Concept Analysis, see Sect. 3.1.

3.1 Formal concept analysis (FCA)

FCA starts with a binary relation R, called *formal context*, defined between a set Obj of objects and a set $Prop$ of Boolean properties. The notation $(x, y) \in R$ means that object x has property y. $R^{\uparrow}(x) = \{y \in Prop | (x, y) \in R\}$ is the set of properties of object x. Similarly, $R^{\downarrow}(y) = \{x \in Obj | (x, y) \in R\}$ is the set of objects having property y.

Given a set Y of properties, one can define the set of objects [7]: $R^{\downarrow}(Y) = \{x \in Obj | R^{\uparrow}(x) \supseteq Y\}$.

This is the set of objects sharing all properties in Y (and having maybe some others). Then a *formal concept* is defined as a pair made of its *extension* X and its *intension* Y such that

$$R^{\downarrow}(Y) = X \text{ and } R^{\uparrow}(X) = Y,$$

where $(X, Y) \subseteq Obj \times Prop$, and $R^{\uparrow}(X)$ is similarly defined as $\{y \in Prop | R^{\downarrow}(y) \supseteq X\}$. It can be also shown that formal concepts are maximal pairs (X, Y) (in the sense of inclusion) such that $X \times Y \subseteq R$.

Moreover, the set of all formal concepts is equipped with a partial order (denoted $\preceq$) defined as: $(X_1, Y_1) \preceq (X_2, Y_2)$ iff $X_1 \subseteq X_2$ (or, equivalently, $Y_2 \subseteq Y_1$), and forms a complete lattice, called the concept lattice of R.

Let us consider an example where R is a relation that defines links between eight objects $Obj = \{1, 2, 3, 4, 5, 6, 7, 8\}$ and nine properties $Prop = \{a, b, c, d, e, f, g, h, i\}$. There is a "×" in the cell corresponding to an object x and to a property y if the object x has property y, in other words the "×"s describe the relation R (or context). An empty cell corresponds to the fact that $(x, y) \notin R$, i.e., it is known that object x has not property y. The relation R in the example is given in Figure 1. There are 5 formal concepts. For instance, consider $X = \{a, b, c, d, e\}$. Then $R^{\uparrow\Delta}(X) = \{7, 8\}$; likewise if $Y = \{7, 8\}$. Then $R^{\downarrow\Delta}(Y) = \{a, b, c, d, e\}$.

	a	b	c	d	e	f	g	h	i
1							×		
2							×		
3							×	×	×
4							×	×	×
5			×	×	×	×			
6			×	×	×	×			
7	×	×	×	×	×				
8	×	×	×	×	×				

Fig. 1. R_2: a relation with 5 formal concepts (and 2 sub-contexts)

3.2 Organizing the search

The basic idea of our algorithm is to start from some 4-tuple of objects, to observe the attributes that contribute to make AD non-zero for this 4-tuple, and to replace one of the four objects by another object. Then we iterate the process. Two important features have been added to avoid a random walk in the space of the 4-tuples.

1. The replacement of one object by another is done according to the observation of the lattice of concepts. The idea is to try to decrease the value of AD. This point will be explained in the next section.
2. All 4-tuples of objects that are created are stored in a list, ordered by increasing value of AD. The next 4-tuple to be chosen is the first in the list.

This algorithm can therefore be seen as an optimization procedure, more precisely as a *best-first* version of the *GRAPHSEARCH* algorithm ([11]). The ordered list of 4-tuples is an *Open* list in this interpretation.

3.3 Decreasing the analogical dissimilarity

Let us come now to the heart of the algorithm, namely the replacement of one object by another. Can we find some information in the lattice that leads us to choose both an object in the 4-tuple and another object to replace it ? Remember that we are looking for a replacement that makes the AD decrease.

Now, let us consider the following situation, taken from $\mathcal{BASE}_{lm}$ (see Figure 2). Suppose that we are studying the 4-tuple of objects $(3, 4, 9, 12)$, with $AD = 1$. Attribute c is the only one to contribute in the AD of this 4-tuple. Actually, c has the values $(1, 1, 0, 1)$ on the 4-tuple $(3, 4, 9, 12)$. We notice now that there are two interesting concepts in the lattice with respect to c, the first one being $(\{b, c, h, g, a\}\ ,\ \{3\})$ and the second being $(\{b, h, g, a\}\ ,\ \{2, 3\})$, which are directly connected. What can we deduce from this pair of connected concepts ?

- Attribute c has value 1 on object 3.
- Attribute c has value 0 on object 2.
- Attribute c is the only attribute to switch from 1 to 0 to transform concept $(\{b, c, h, g, a\}\ ,\ \{3\})$ into concept $(\{b, h, g, a\}\ ,\ \{2, 3\})$.

We can conclude from this evidence that replacing object 3 by object 2 will decrease by 1 the value of AD, since c will take values $(0, 1, 0, 1)$ on the 4-tuple $(2, 4, 9, 12)$, and therefore that $(2, 4, 9, 12)$ is an analogical proportion.

Unfortunately this argument does not lead to a greedy algorithm, since there no insurance, given a 4-tuple, that there exists a couple of concepts having the three above properties. Most of the time, actually, there is more than one attribute switching to 1 between two connected concepts, and only one insuring the decreasing of AD.

Let us take another example from the same data base. The 4-tuple $(5, 4, 9, 12)$ has a AD of 4, because of the attributes d, f, g and h. Two interesting connected

concepts are $(\{c, f, d, e, i, a\}\ ,\ \{12\})$ and $(\{c, d, e, a\}\ ,\ \{7, 12\})$, since f switching from 1 to 0 will decrease the AD (see the table below). But replacing 12 by 7 in the 4-tuple $(5, 4, 9, 12)$ will switch not only f but also the attribute i, and we don't know what will happen when switching i: it may decrease as well as increase the AD. Actually, in this case, it increases the AD. Hence, the 4-tuple $(5, 4, 9, 7)$ has the same AD of 4.

	a	b	c	d	e	f	g	h	i
5	×	×		×		×			
4	×		×				×	×	×
9	×	×		×	×	×			
12	×		×	×	×	×			×

$AD = 4$

	a	b	c	d	e	f	g	h	i
5	×	×		×		×			
4	×		×				×	×	×
9	×	×		×	×	×			
7	×		×	×	×				

$AD = 4$

By generalizing these examples, we propose now an heuristic to try to decrease the AD that we call `h-Doap`.

> **Heuristic `h-Doap`.** Let a couple of connected concepts be $(A \cup B, Z)$ and $(B, Z \cup Y)$, A and B being subsets of attributes and Y and Z being subsets of objects such that $A \cap B = \emptyset$ and $Y \cap Z = \emptyset$. If there is a 4-tuple with one of its four objects in Z and if there is an attribute in B that decreases the AD of this 4-tuple when switching from 1 to 0, then create a new 4-tuple by replacing the object in Z by an object in Y.

Next section shows how this heuristic can be used to discover 4-tuples of objects with null AD in a formal context, i.e. analogical proportions of objects.

3.4 Algorithm

Discovering one analogical proportion. We explain in this section the algorithm used to discover one analogical proportion in a formal context. We call it '`Discover One Analogical Proportion`', in short `Doap`. As already stated, it is a simple version of *Graphsearch*, where the nodes to be explored are 4-tuples of objects. We denote *Start* the 4-tuple of objects chosen to begin, *Open* the current set of 4-tuples to be processed and *Closed* the set of 4-tuples already processed.

The choice of *Start* is either done randomly or by selecting objects which appear in small subsets of objects in the lattice. We also require that the explored 4-tuples are composed of four different objects, since we do not want to converge towards 4-tuples trivially in proportion, such as $(1, 3, 1, 3)$ or even $(2, 2, 2, 2)$.

The algorithm stops either by discovering an analogical proportion, or in failure. One has to notice that its failure does not insure that there is no analogical proportion, since there is no guarantee given by the heuristic. We have never met this failure case, but our experiments are very limited, as explained in section 3.5.

```
1: Algorithm Doap(Start)
2: begin
3: Closed ← ∅
4: Open ← {Start}
5: while Open ≠ ∅ do
6:    x ← the 4-tuple in Open having the lowest AD value
7:    if AD(x) = 0 then
8:       return x
9:    else
10:      Open ← Open \ {x} ; Closed ← Closed ∪ {x}
11:      decision ← 1
12:      while decision = 1 do
13:         Use heuristic h-Doap to construct a new 4-tuple y from x
14:         if y is composed of four different objects and y ∉ Closed and y ∉
            Open then
15:            Open ← Open ∪ {y} ; decision ← 0
16:         end if
17:      end while
18:   end if
19: end while
20: return failure
21: end
```

Discovering several analogical proportions. To discover more analogical proportions, the simplest manner is to imbed algorithm `Doap` in a procedure that discards the first two objects of a discovered analogical 4-tuple from the formal context before re-running the algorithm. Since the transitivity holds for analogical proportions on objects ($u : v :: w : x$ and $w : x :: y : z$ implies $u : v :: y : z$), we are loosing no information on analogical 4-tuples. However, we are not insured to find all proportions in that manner, due to the fact that algorithm `Doap` may not find an existing proportion.

3.5 Experiments

The size of *Close* when the algorithm `Doap` stops is a precise indication of its practical time complexity. Notice that a random algorithm, running on n objects (without any construction of a formal lattice), in which there are q 4-tuples in analogical proportion would in average try $((n^4)/8 \cdot q)$ 4-tuples before discovering a proportion. In the previous formula, the number "8" comes from the fact that, when there is one analogical proportion in a formal context, then there are in fact exactly 8 through suitable permutations. This property stems directly from the postulates an analogical proportion (see section 2).

We have used two different formal contexts to run the algorithm `Doap`. The first one is described in [2], except that we have added four objects 9, 10, 11 et 12 in order to have (at least) the analogical proportions $(3, 4, 9, 10)$ and $(1, 8, 11, 12)$. This leads to the formal context called $\mathcal{BASE}_{lm}$, Figure 2.

		a	b	c	d	e	f	g	h	i
leech	1	×	×					×		
bream	2	×	×					×	×	
frog	3	×	×	×				×	×	
dog	4	×		×				×	×	×
spike-weed	5	×	×		×		×			
reed	6	×	×	×	×		×			

		a	b	c	d	e	f	g	h	i
bean	7	×		×	×	×				
maize	8	×		×	×		×			
x	9	×	×		×	×	×			
y	10	×			×	×	×			×
z	11	×	×			×		×		×
t	12	×		×	×	×	×			×

Fig. 2. $\mathcal{BASE}_{lm}$: A formal context from Bělohlávek [2] increased with four objects 9, 10, 11 et 12 in order to have (at least) the analogical proportions $(3, 4, 9, 10)$ and $(1, 8, 11, 12)$.

The lattice constructed on this formal context (with the In-Close free software [14]) has 31 concepts. We have run `Doap` more than 600 times. It has always terminated by finding one of the three analogical proportions in the data[4]. The average size of the *Closed* list is 63 and its median size is 28. Figure 3 gives the details.

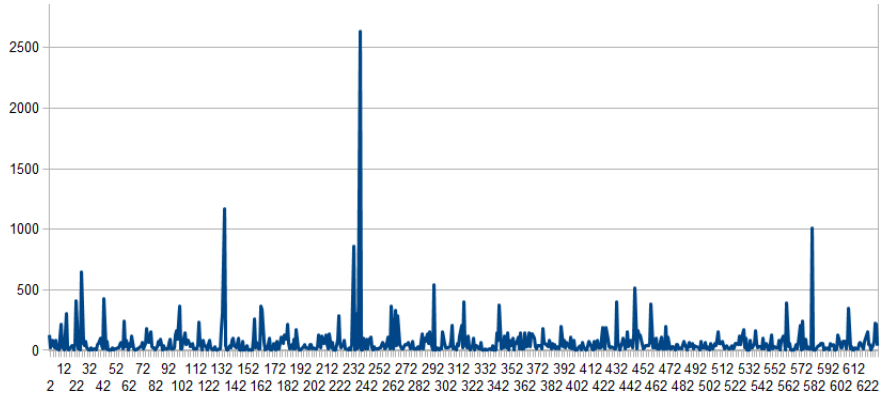


Fig. 3. Results of 622 runs of `Doap` on $\mathcal{BASE}_{lm}$. The size of the *Close* list for each run is on the Y axis.

To appreciate these results, we have compared with a random search, replacing line 13 of the `Doap` algorithm by picking a random 4-tuple. The detailed results are given in Figure 4. The average size of the *Closed* list is 253 and its median size is 174.

We also have tried to "symmetrize" the role of 0 and 1 in this formal context by adding the reverse attributes (indeed the Table 1 defining analogical proportions is left unchanged when exchanging 0 and 1). It leads to 12 objects and 18

[4] We actually had a good surprise: `Doap` found a third proportion, namely $(2, 4, 9, 12)$.

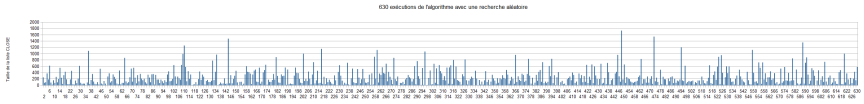


Fig. 4. Results of 630 runs of random `Doap` on $\mathcal{BASE}_{lm}$. The Y axis is graduated from 0 up to 2000.

attributes instead of 9. The size of the lattice of concepts is now 94. The algorithm `Doap` with the same parameters examines in average 93 4-tuples before finding an analogical proportion. The median value is 31. The symmetrization does not seem to be a good idea in this case. The random `Doap` algorithm has failed to give complete results on these data, due to overflows in the *Close* list.

The second experiment has been run on the Lenses data base, from UCI ML Repository [6]. The nominal attributes have been transformed into binary ones by simply creating as many binary attributes as the number of modalities. The number of objects is 24 with 7 binary parameters. The size of the lattice is 43, the average number of examined 4-tuples is 77 and the median number is 20. When adding the reverse attributes, we have a lattice of size 227, an average number of 39 and a median number of 16. In that experiment, the symmetrization of the data seems clearly to have a positive effect.

A first conclusion is that our heuristic algorithm seems to perform well. In the second context the basic search space has a size over 40.000 and we examine only 77 4-tuples in average. The construction of the lattice of concepts takes in practice much more time than the discovery of analogical proportions, which seems to suggest that it is a relevant space for looking for analogical proportions.

4 Analogical proportions between formal concepts

We have seen that discovering analogical proportions in a formal context benefits from the knowledge of the associated lattice of formal concepts. Then it raises the question of understanding how formal concepts are involved in analogical proportions. Clearly, four objects in the same formal concept form an analogical proportion – in a trivial way – w.r.t. the subset of attributes involved in the formal concept. Partial answers to the question, when two formal concepts are involved in the proportion, are given in this section.

4.1 The smallest formal context in complete proportion

We are interested in this section in examining the properties of the smallest context with an analogical proportion between objects. Obviously, this context will have exactly four objects. If we want to have, only one time, each of the possible analogical proportions between attributes, we need six of them (see table 1) and we obtain $\mathcal{BASE}_0$ (see Figure 5).

	f	a	b	c	d	e
u				×	×	×
v			×		×	×
w		×		×		×
x		×	×			×

	a	b	c	d
u			×	×
v		×		×
w	×		×	
x	×	×		

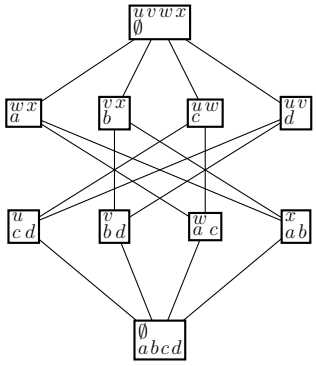


Fig. 5. $\mathcal{BASE}_0$, $\mathcal{BASE}_1$ and the concepts lattice of $\mathcal{BASE}_1$. The lattice of $\mathcal{BASE}_0$ is deduced from it by adding e to all subsets of attributes.

We can construct now the concept lattice of $\mathcal{BASE}_0$, but it is interesting to get rid of attributes f (which will not be present in any context) and e (present in every context). We call $\mathcal{BASE}_1$ the reduced context, shown at Figure 5.

Its lattice is displayed in Figure 5. Note that there is a perfect symmetry between attributes and objects. The third line of the lattice expresses that $u : v :: w : x$, but also in subsets terms that $\{c, d\} : \{b, d\} :: \{a, c\} : \{a, b\}$. The second line expresses that $a : b :: c : d$ and that $\{w, x\} : \{v, x\} :: \{u, w\} : \{u, v\}$. This is not surprising: as explained in section 2, we can see an object as the set of properties that hold true for it.

4.2 Some relations between analogical proportions and lattices of concepts

Firstly, let us remark that the two following propositions are equivalent. This is immediate from section 2, in which these two equivalent definitions of analogical proportion have been presented.

1. x_1, x_2, x_3 and x_4 are four objects, in analogical proportion in this order.
2. $R^{\uparrow}(x_1)$, $R^{\uparrow}(x_2)$, $R^{\uparrow}(x_3)$ and $R^{\uparrow}(x_4)$ are four subsets of properties in analogical proportion in this order.

Property 1 *Let x_1, x_2, x_3 and x_4 be four objects in analogical proportion in this order. Let (X_1, Y_1) be the[5] concept with the smallest set X_1 of objects in which x_1 is present. Let us define (X_2, Y_2), (X_3, Y_3) and (X_4, Y_4) in the same way. Then the four sets of attributes Y_1, Y_2, Y_3 and Y_4 are in analogical proportion, in this order.*

Proof. Since $x_1 \in X_1$, all the attributes in Y_1 take value 1 on x_1. Since X_1 is the smallest set of objects including x_1, there is no attribute outside Y_1 having

[5] If they were two, x_1 would be present in the intersection of the two.

value 1 on x_1. Hence, Y_1 is exactly $R^{\uparrow}(x_1)$, the extension of x_1, i.e. the subset of attributes that take value 1 on x_1. This is also true for x_2, x_3 and x_4.
We have to prove now that $x_1 : x_2 :: x_3 : x_4$ implies $R^{\uparrow}(x_1) : R^{\uparrow}(x_2) :: R^{\uparrow}(x_3) : R^{\uparrow}(x_4)$. It is immediate from the remark above. □

For example, in $\mathcal{BASE}_0$, we know that $1 : 8 :: 11 : 12$. We have $X_1 = \{1, 2, 3, 11\}$, $Y_1 = \{a, b, g\}$, $X_2 = \{6, 8, 12\}$, $Y_2 = \{a, c, d, f\}$, $X_3 = \{11\}$, $Y_3 = \{a, b, e, g, i\}$ and $X_4 = \{12\}$, $Y_4 = \{a, c, d, e, f, i\}$. The proportion Y_1:Y_2::Y_3:Y_4 holds, since: $\{a, b, g\}$:$\{a, c, d, f\}$::$\{a, b, e, g, i\}$:$\{a, c, d, e, f, i\}$.

Property 2 *Let (X_1, Y_1), (X_2, Y_2), (X_3, Y_3) and (X_4, Y_4) be four concepts of a lattice of concepts, such that the four sets of attributes Y_1, Y_2, Y_3 and Y_4 are in analogical proportion, in this order. Let $\widehat{X}_1$ be the subset of X_1 composed of all objects that are in X_1 but cannot be found in any subset of X_1 belonging to a concept. We define in the same manner $\widehat{X}_2$, $\widehat{X}_3$ and $\widehat{X}_4$. The following property holds true: $\forall x_1 \in \widehat{X}_1, x_2 \in \widehat{X}_2, x_3 \in \widehat{X}_3, x_4 \in \widehat{X}_4 : \quad x_1\ x_2\ ::\ x_3\ :\ x_4$.*

Proof. *It is the reciprocal of Property 1: Y_1 is the extension of all objects in $\widehat{X}_1$, and we take x_1 in $\widehat{X}_1$. We derive the conclusion from the remark above.* □

Property 3 *Let x_1, x_2, x_3 and x_4 be four objects, in analogical proportion in this order.*

Let $A_{1111} = \{y | y \in R^{\uparrow}(x_1), y \in R^{\uparrow}(x_2), y \in R^{\uparrow}(x_3), y \in R^{\uparrow}(x_4))\}$
Let $A_{1100} = \{y | y \in R^{\uparrow}(x_1), y \in R^{\uparrow}(x_2), y \notin R^{\uparrow}(x_3), y \notin R^{\uparrow}(x_4))\}$
Let $A_{0011} = \{y | y \notin R^{\uparrow}(x_1), y \notin R^{\uparrow}(x_2), y \in R^{\uparrow}(x_3), y \in R^{\uparrow}(x_4))\}$
Let $A_{1010} = \{y | y \in R^{\uparrow}(x_1), y \notin R^{\uparrow}(x_2), y \in R^{\uparrow}(x_3), y \notin R^{\uparrow}(x_4))\}$
Let $A_{0101} = \{y | y \notin R^{\uparrow}(x_1), y \in R^{\uparrow}(x_2), y \notin R^{\uparrow}(x_3), y \in R^{\uparrow}(x_4))\}$
Then
$(\{x_1, x_2\}, A_{1111} \cup A_{1100})$ *is included into a formal concept.*
$(\{x_3, x_4\}, A_{1111} \cup A_{0011})$ *is included into a formal concept.*
$(\{x_1, x_3\}, A_{1111} \cup A_{1010})$ *is included into a formal concept.*
$(\{x_2, x_4\}, A_{1111} \cup A_{0101})$ *is included into a formal concept.*

The result follows from the definition of the subsets of attributes considered and their clear relation with the definition of analogical proportions. The fact that we only have an inclusion in the above property should not come as a surprise. Indeed, when describing objects, attributes that are nor not relevant w.r.t. the analogical proportion may be present.

5 Lines for further research and concluding remarks

Beyond the already introduced set function, $R^{\downarrow}(Y) = \{x \in Obj | R^{\uparrow}(x) \supseteq Y\}$, which is at the core of FCA,and which leads to the definition of formal concepts, it has been noticed [5], on the basis of a parallel with possibility theory that, given a set Y of properties, four remarkable sets of objects can be defined in this setting (here the overbar denotes set complementation):

- $R^{\downarrow\Pi}(Y) = \{x \in Obj | R^{\uparrow}(x) \cap Y \neq \emptyset\} = \cup_{y \in Y} R^{\downarrow}(y)$. This is the set of objects having at least one property in Y.
- $R^{\downarrow N}(Y) = \{x \in Obj | R^{\uparrow}(x) \subseteq Y\} = \cap_{y \notin Y} \overline{R^{\downarrow}(y)}$. This is the set of objects having no property outside Y.
- $R^{\downarrow\Delta}(Y) = R^{\downarrow}(Y) = \cap_{y \in Y} R^{\downarrow}(y)$. This is the set of objects sharing all properties in Y.
- $R^{\downarrow\nabla}(Y) = \{x \in Obj | R^{\uparrow}(x) \cup Y \neq Obj\} = \cup_{y \notin Y} \overline{R^{\downarrow}(y)}$. This is the set of objects that are missing at least one property outside Y.

It has been recently pointed out [3] that pairs (X, Y) such that $R^{\downarrow N}(Y) = X$ and $R^{\uparrow N}(X) = Y$ are characterizing independent sub-contexts (X, Y) such that $((X \times Y) + (\overline{X} \times \overline{Y}) \supseteq R$, in the sense that they do not share any object or property. Thus, in Figure 1, $(\{a, b, c, d, e, f\}, \{5, 6, 7, 8\})$ and $(\{g, h, i\}, \{1, 2, 3, 4\})$ are two formal sub-contexts.

When comparing the features underlying FCA and analogical proportions, one can notice that the same 4 "indicators" are involved from the beginning: $a \cap b$, $a \cap \overline{b}$, $\overline{a} \cap b$, and $\overline{a} \cap \overline{b}$. Indeed $R^{\downarrow\Pi}(Y)$ is based on the condition $R^{\uparrow}(x) \cap Y \neq \emptyset$, $R^{\downarrow N}(Y)$ on the condition $R^{\uparrow}(x) \cap \overline{Y} = \emptyset$, $R^{\downarrow\Delta}(Y)$ on the condition $\overline{R^{\uparrow}(x)} \cap Y = \emptyset$, and $R^{\downarrow\nabla}(Y)$ on the condition $\overline{R^{\uparrow}(x)} \cap \overline{Y} \neq \emptyset$. Moreover, with these 4 indicators, one can define other so-called logical proportions [4], including some that are closely related to analogical proportions such as 'paralogy' which reads "what a and b have in common, c and d have it also" and is defined by $a \wedge b = c \wedge d$ and $\overline{a} \wedge \overline{b} = \overline{c} \wedge \overline{d}$ [10]. This more generally raises the question of the relations between FCA and these logical proportions.

Finally, the experiments with `Doap` have obviously to be scaled on larger formal contexts, in order to estimate its practical complexity more accurately. Some more thought has also to be given about the choice of the *Start* 4-tuples, especially to take advantage of the addition of the reverse attributes. An interesting point would be to be able to choose the *Start* in order to insure that every analogical proportion can be discovered. We also believe that the speed of `Doap` can be increased, since there are still a lot of parameters to tune, for example breaking ties in the head of the *Close* list in a non random fashion.

An interesting question is whether or not the construction of the lattice must precede the heuristic search. It would certainly be of great interest to construct only the parts that are required by the running od the *Doap* algorithm. This would lead to merge the two parts of the method, rather than computing the whole lattice (a very costly operation) before its exploration.

More generally, it would be clearly of interest to have an algorithm also able to find out the analogical proportions that hold in some sub-context (since as already said, irrelevant attributes may hide interesting analogical proportions), rather than in the initial formal context. This will open a machine learning point of view [1].

6 Aknowledgements

We would like to thank the anonymous reviewers for their careful reading of this article and their interesting suggestions.

References

1. S. Bayoudh, L. Miclet, and A. Delhay. Learning by analogy: a classification rule for binary and nominal data. Proc. 20th Inter. Joint Conf. on Artificial Intelligence, (M. M. Veloso, ed.), Hyderabad, India, AAAI Press, 678–683, 2007.
2. R. Bělohlávek. Introduction to formal context analysis. Internal report. Dept of Computer science. Palacký University, Olomouk, Czech Republic. 2008.
3. Y. Djouadi, D. Dubois, H. Prade. Possibility theory and formal concept analysis: Context decomposition and uncertainty handling. Proc. 13th Inter. Conf. on Information Processing and Management of Uncertainty (IPMU'10), (E. Hüllermeier, R. Kruse and F. Hoffmann, eds.), Dortmund, Springer, LNCS 6178, 260–269, 2010.
4. H. Prade, G. Richard. Logical proportions - Typology and roadmap. Proc. Inter. Conf. on Information Processing and Management of Uncertainty in Knowledge-based Systems (IPMU 2010), Dortmund, (E. Hüllermeier, R. Kruse, F. Hoffmann, eds.), Springer, LNCS 6178, 757–767, 2010.
5. D. Dubois, F. Dupin de Saint-Cyr, H. Prade. A possibility-theoretic view of formal concept analysis. Fundamentae Informaticae, 75, 195–213, 2007.
6. A. Frank and A. Asuncion. (2010). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
7. B. Ganter and R. Wille. Formal Concept Analysis. Mathematical Foundations. Springer Verlag, 1999.
8. Y. Lepage. De l'analogie rendant compte de la commutation en linguistique. http://www.slt.atr.jp/ lepage/pdf/dhdryl.pdf, Grenoble, 2001. HDR.
9. L. Miclet and H. Prade. Handling analogical proportions in classical logic and fuzzy logics settings. Proc. 10th Europ. Conf. on (ECSQARU'09), Verona, Springer, LNCS 5590, 2009, 638–650.
10. H. Prade and G. Richard. Analogy, paralogy and reverse analogy: Postulates and Inferences. Proc. Annual German Conf. on Artificial Intelligence (KI 2009), Padernorn, Sept. 15-18, (B. Mertsching, M. Hund, Z. Aziz, eds.), Springer, LNAI 5803, 306–314, 2009.
11. N. Nilsson. Principles of Artificial Intelligence. Tioga, 1980.
12. N. Stroppa and F. Yvon. Analogical learning and formal proportions: Definitions and methodological issues. Technical Report ENST-2005-D004, http://www.tsi.enst.fr/publications/enst/techreport-2007-6830.pdf, June 2005.
13. R. Wille. Restructuring lattice theory: an approach based on hierarchies of concepts. In: Ordered Sets, (I. Rival, ed.), D. Reidel, Dordrecht, 445–470, 1982.
14. The Inclose software. http://inclose.sourceforge.net/. Downloaded on March 2011.

Random extents and random closure systems

Bernhard Ganter

Institut für Algebra
Technische Universität Dresden

Abstract. We discuss how to randomly generate extents of a given formal context. Our basic method involves counting the generating sets of an extent, and we show how this can be done using the Möbius function. We then show how to generate closure systems on seven elements uniformly at random.

1 Introduction

Let RANDOM(0,1] denote an operator that generates a random number between 0 and 1 with equal probability. From such a (memoryless) random number generator an operator RANDOM_SUBSET(S) can be derived that produces, upon each invocation, a random subset of a given finite set S, such that all subsets are equally likely (see, e.g., [6]).

Building on this we derive in this article an operator that randomly selects a closed set from a given closure system[1] on a finite set.

Note that this is a trivial task for moderately sized systems of which you can label the closed sets by numbers $1, \ldots, n$. For such you could simply randomly pick a number between 1 and n and select the closed set labeled by this number. Since the size of a closure system is at most exponential in the size of its carrier, this trivial algorithm clearly requires polynomial time. However, a potentially exponential list of closed sets must be pre-computed and stored.

For example we aim at generating *closure systems* at random[2]. But there are many closure systems, even for small carrier sets. On seven elements the number was recently computed by Colomb, Irlande, and Raynaud [3] to be 14 087 648 235 707 352 472. Maintaining a list of this size is not an inviting idea, and thus the trivial approach is not very realistic.

Our motivation comes from recent experimental computer investigations by D. Borchmann that yielded surprising results. Borchmann raised the question if these were artefacts caused by the non-uniform choice of the random input data.

Have a look at Figure 1. It shows five diagrams, each with 27 rows and 13 columns, corresponding to the possible number of meet reducible and irreducible closed sets in a closure system on a five element set (the trivial system with zero irreducibles is omitted). A system with r reducibles and i irreducibles corresponds

[1] That is, from an intersection-closed familiy of sets. Such families are also called *Moore families*.
[2] The family of all closure systems on a fixed set is itself a closure system.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

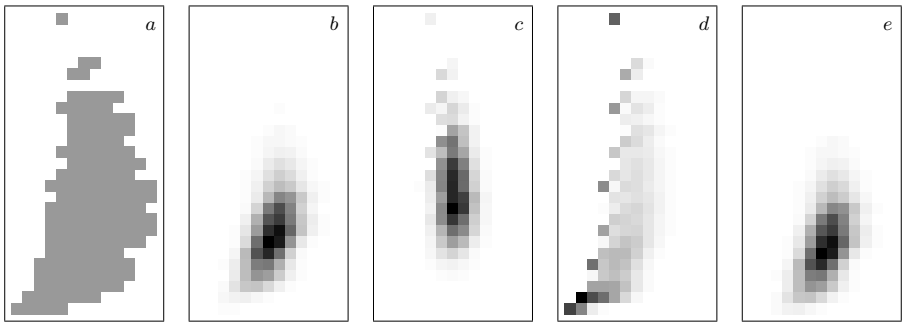


Fig. 1. Closure systems on five elements by their number of meet-irreducible (horizontal) and reducible closed sets (vertical). Tile a shows the possible values, tile b the true relative frequencies, tile c and d come from "random contexts" and tile e from picking systems uniformly at random.

to the cell in the r-th row from the bottom and the i-th column from the left. The first diagram depicts which combinations of r and i are possible, while the other four display relative frequencies (the darker, the higher). The second diagram shows the true frequencies, counted over all 1 385 552 closure systems on five elements. The other three show frequencies of randomly chosen closure systems (1 000 000 samples each). For the diagram in the middle, the systems were made by putting "random crosses" in a 5×13–context. The fourth diagram was obtained by putting "random crosses" with random density in a formal context with a random number of columns. The fifth diagram shows the distribution of a sample picked with uniform distribution.

We use the language of Formal Concept Analysis [4] and, in particular, that every closure system is the set $\mathrm{Ext}(G,M,I)$ of all extents of some formal context (G,M,I). We construct an operator $\textsc{Random\_extent}(G,M,I)$ which selects, upon each invocation, randomly an extent of (G,M,I), with equal probability for all extents.

The closure operator for the extents will be denoted by

$$X \mapsto X''.$$

If $X = Y''$, then Y is called a *generating set* of the extent X (not necessarily a *minimal* one). The number of generating sets of an extent E shall be denoted by $egen(E)$. We extend this definition to arbitrary subsets so that

$$egen(Y) := |\{X \subseteq G \mid X'' = Y''\}|$$

gives the number of generating sets of the extent generated by Y. Of course then, $egen(Y) = egen(Y'')$. Therefore if E is an extent then obviously

$$\sum_{\{Y|Y''=E\}} \frac{1}{egen(Y)} = 1.$$

Computing the function *egen*() is a nontrivial task. We shall discuss this below.

Our method could theoretically be applied to many instances, such as generating random partitions, random subgroups, etc. However, its runtime performance is very bad. For most such situations algorithms are known that are much more efficient than what we suggest. Indeed, we do not believe that our method will be very useful in practice. Our contribution is meant as a challenge to come up with a more efficient approach.

We are grateful to the referees for several useful hints. We were unaware of the paper by Boley, Gärtner, and Grosskreutz [2], which addresses the same problem, but with a different and more general approach. It may well be that their algorithm yields better results even for generating random closure systems. We have also learnt that the problem of generating random extents is known to belong to a (difficult) complexity class: it is equivalent to the $\#\mathrm{RH}\Pi_1$-hard problem of counting formal concepts (again, see [2] and the references given there). We already knew (because our colleagues of the stochastics group told us so and recommended the book by Asmussen and Glynn [1] as a standard reference) that our approach is an instance of the so-called *acceptance-rejection method*.

2 Random Extent

Our innocent looking algorithm for generating a random extent of a given formal context (G, M, I) goes like this:

Algorithm 1 RANDOM_EXTENT: Generating a random extent

Input: A formal context (G, M, I).
Output: A random extent of (G, M, I)
repeat
  $S :=$ RANDOM_SUBSET(G)
until RANDOM$(0,1] \leq \frac{1}{egen(S)}$
return S''.

What the algorithm does essentially is to pick a random subset and output its closure with probability one over the number of generating sets[3]. It is quite elementary to prove that it does what it is supposed to do:

Proposition 1 *The algorithm* RANDOM_EXTENT *generates extents of* (G, M, I) *with equal probability.*

The proposition is an instance of the following lemma from elementary stochastics, for which we provide a proof. To obtain the proposition from the lemma, let

[3] One of the referees pointed out that a much simpler algorithm with the same number of expected iterations is obtained by replacing the "until" statement by "until S is closed". We see however no straightforward way to a recursive version of this algorithm.

A be the set of all subsets of G, let B be the set of all extents, and let f be the map that associates a subset to the extent it generates.

Lemma 1 *Let $f : A \to B$ be a surjective (i.e., onto) map between finite sets A and B and let* RANDOM(A) *be an operator that picks elements from A with equal probability. Then Algorithm 2 outputs elements of B with equal probability.*

Algorithm 2 RANDOM IMAGE: Random image of a mapping

Input: An onto map $f : A \to B$ and an operator RANDOM(A)
Output: A random element of B

```
repeat
    a := RANDOM(A)
    r := RANDOM(0,1]
    b := f(a);
until r ≤ 1/|f^{-1}(b)|
return b.
```

Proof It is obvious that the algorithm produces elements of B. In order that a given element b is produced in one iteration of the loop, the element a must belong to $f^{-1}(b)$ and, independently, $r \leq \frac{1}{|f^{-1}(b)|}$. The probability that this happens is

$$\frac{|f^{-1}(b)|}{|A|} \cdot \frac{1}{|f^{-1}(b)|} = \frac{1}{|A|},$$

independently of b. The probability that some element is selected after one step thus is $\frac{|B|}{|A|}$. The probability that the element b is produced after k steps is

$$\left(1 - \frac{|B|}{|A|}\right)^{k-1} \cdot \frac{1}{|A|}.$$

The probability that b is produced is

$$\sum_{k=1}^{\infty} \left(1 - \frac{|B|}{|A|}\right)^{k-1} \cdot \frac{1}{|A|} = \frac{|A|}{|B|} \cdot \frac{1}{|A|} = \frac{1}{|B|},$$

as claimed. □

The expected number of iterations until success is

$$\frac{|A|}{|B|} = \frac{\#\text{subsets}}{\#\text{extents}}.$$

The algorithm may therefore need quite some time. For example, would this algorithm be applied to the standard context of closure systems to generate a random

closure system on a 6-element set, it requires, on average, 121 402 088 iterations of the loop, since that context has $2^6 - 1$ objects and 75 973 751 474 extents ([5]). For closure systems on a seven-element set the average number of loop iterations for obtaining a single random closure system would be 12 077 330 482 260 320 447. As already mentioned we shall develop a better method for this case below. Before we do so, we study the problem of computing the value of $egen(A)$.

3 Counting generating sets and hitting sets

The algorithm in the previous section uses the number $egen(A)$ of a given extent A, and that by itself is not easy. Of course, each such generating set must be a subset of A. On the other hand, a subset $S \subseteq A$ is a generating set of A iff it is not contained in a lower neighbor of A. It is worthwhile to consider the formal context

$$(A, \mathcal{N}, \in),$$

where $\mathcal{N}$ is the family of lower neighbor extents of A. For this context, the elements of $\mathcal{N}$ are precisely the maximal extents below A, and thus the generating sets of A are the same as before. Counting generating sets thereby has been reduced to counting generating sets of the unit element in a co-atomistic lattice.

Every subset of A is generating set of exactly one extent of $(A, \mathcal{N}, \in)$. The total number of generating sets thus is $2^{|A|}$. Indeed, for every extent B we obtain

$$\sum_{E \leq B} egen(E) = 2^{|B|},$$

where E runs over extents. By Möbius inversion we obtain

$$egen(A) = \sum_{E \leq A} \mu(E, A) \cdot 2^{|E|},$$

where μ is the Möbius function of the lattice $\underline{\mathfrak{B}}(A, \mathcal{N}, \in)$.

The evaluation of this formula poses no algorithmic difficulties. Using the standard NEXT_INTENT algorithm ([4]) to generate the extents in descending order, and using, for every constructed extent E, the same algorithm again for producing all extents F between E and A, suffices to compute the Möbius function by the well known recursion

$$\mu(E, A) = -\sum_{E < F \leq A} \mu(F, A).$$

Note that this also counts the hitting sets of any finite hypergraph. A **hitting set** of a family $\mathcal{H} \subseteq \mathfrak{P}(A)$ of subsets of A is a set $T \subseteq A$ that has nonempty intersection with each $H \in \mathcal{H}$ (such sets are also called **(weak) transversal**s).

Obviously, T is a hitting set iff T is not a subset of a complement of some $H \in \mathcal{H}$, that is, iff T is a generating set of the extent A in the formal context

$$(A, \mathcal{H}^c, \in),$$

where

$$\mathcal{H}^c := \{A \setminus H \mid H \in \mathcal{H}\}.$$

The algorithm given above therefore applies. However, using the Möbius function for counting generating sets is costly. Fortunately, it is unnecessary in many cases, as we shall explain in the next section.

4 Splitting the loop

The algorithm poses no conditions on *how* the random set and the random number are generated, and in which order. A huge number of iterations is necessary when we first generate the extent and then perform a random experiment with very low success probability. A better strategy is to perform the random experiment in several steps and interrupt the loop at an early stage, if necessary. For given $0 \leq p < 1$ and $p \leq \alpha < 1$ we may replace the experiment

pick a random number $r = \textsc{Random}(0,1]$ *and test if* $r \leq p$

by

pick two random numbers $r = \textsc{Random}(0,1]$ *and* $s = \textsc{Random}(0,1]$
and test if $s \leq \alpha$ *and* $r \leq \frac{p}{\alpha}$,

because they have the same success probability. This is the key to the following lemma.

Lemma 2 *Let $f : A \to B$ and $g : B \to C$ be surjective mappings between finite sets and let* $\textsc{Random}(B)$ *be an operator that picks elements from B with equal probability. Then Algorithm 3 outputs elements of C with equal probability.*

Proof Suppose that we start with a random choice $a \in A$ and apply Algorithm 3 to the mapping $g \circ f$. We would draw a random number $r := \textsc{Random}(0,1]$ and output $c := g(f(a))$ if

$$r \leq p := \frac{1}{|f^{-1}(g^{-1}(c))|}.$$

Equivalently we may split the drawing of r as described above. We first compute $b := f(a)$ and $\alpha := \frac{1}{f^{-1}(b)}$, draw a random number $s = \textsc{Random}(0,1]$ and continue only if

$$s \leq \alpha := \frac{1}{|f^{-1}(b)|}.$$

Algorithm 3 RANDOM IMAGE 2: Random image of a composed mapping

Input: Two onto maps $f : A \to B$ and $g : B \to C$ and an operator RANDOM(B) that randomly outputs elements of B

Output: A random element of C

repeat
  $b :=$ RANDOM(B)
  $r :=$ RANDOM(0,1]
  $c := g(b)$;
until $r \leq \frac{|f^{-1}(b)|}{|f^{-1}(g^{-1}(c))|}$
return c.

We then draw another random number r and output c if

$$r \leq \frac{p}{\alpha} = \frac{|f^{-1}(b)|}{|f^{-1}(g^{-1}(c))|}.$$

According to Lemma 1 the first part of this process generates the elements of B with equal probability. The first part therefore may be replaced by a random choice of elements of B, as stated in the lemma. □

Lemma 2 can be applied in several ways to the random extent problem. Typically, A is the set of all subsets of G and C is the set of all extents, while B is a selected family of generating sets. For example we may take some subset $G_0 \subseteq G$ and let B consists of all sets of the form $E \cup R$, where E is an extent of the formal context $\mathbb{K}_0 := (G_0, M, I \cap (G_0 \times M))$ and $R \cap G_0 = \emptyset$. This yields Algorithm 4.

Algorithm 4 RECURSIVE RANDOM EXTENT

Input: A formal context $\mathbb{K}$ and an operator RANDOM_EXTENT($\mathbb{K}_0$) producing random extents of a subcontext $\mathbb{K}_0 \leq \mathbb{K}$

Output: A random extent of $\mathbb{K}$.

repeat
  $E :=$ RANDOM_EXTENT($\mathbb{K}_0$)
  $A := E \cup$ RANDOM_SUBSET($G \setminus G_0$)
  $r :=$ RANDOM(0,1];
until $r \leq \frac{egen(E,\mathbb{K}_0)}{egen(A)}$
return A''.

5 Random Moore family

Our original motivation was generating closure systems (also called Moore families) at random. The number of generating sets of a closure system is easy to

determine without using the Möbius function. Every closure system has a unique *minimal* generating set, consisting of all meet-irreducible elements. The total number of generating sets therefore is 2 to the power s, where s is the number of *reducible* closed sets.

The natural formal context for the closure systems on a set S is given by

$$(\mathfrak{P}(S), \mathrm{Imp}(S), \models),$$

where

$$\mathrm{Imp}(S) := \{A \to b \mid A \subseteq S,\ b \in S \setminus A\}$$

is the set of all implications with singleton conclusion over S and

$$X \models A \to b \quad :\Longleftrightarrow \quad A \not\subseteq X \text{ or } b \in X.$$

It is well known and easy to see that the extents of this formal context are precisely the closure systems on S. However, this context contains a reducible object, which is S. The standard context therefore is

$$(\mathfrak{P}(S) \setminus \{S\}, \mathrm{Imp}(S), \models).$$

The case we are interested in is $S := \{0, 1, \ldots, n-1\}$. The object set G of the context then consists of all proper subsets of S. We choose

$$G_0 := \{X \subseteq S \mid n-1 \notin X\}$$
$$G_1 := \{X \subset S \mid n-1 \in X\}.$$

The extents of $(G_1, \mathrm{Imp}(S), \models)$ are (when S is added) precisely those closure systems on S that contain $n-1$ in every closed set. These are in an obvious bijection to the closure systems on $S \setminus \{n-1\}$.

The context $(G_0, \mathrm{Imp}(S), \models)$ has precisely twice as many extents as there are closure systems on $\{0, \ldots, n-2\}$: Each closure system $\mathcal{C}$ on $\{0, \ldots, n-2\}$ is an extent, but also $\mathcal{C} \setminus \{S\}$ is.

Let $S := \{0, \ldots, n-1\}$. For applying Lemma 2 we let A be the set of all subsets of $\mathfrak{P}(S) \setminus \{S\}$, C be the family of all closure systems on S, let G_0 and G_1 be as above and B consist of those set families $\mathcal{F} \subseteq \mathfrak{P}(S) \setminus \{S\}$ for which $\mathcal{F} \cap G_0$ and $\mathcal{F} \cap G_1$ are extents of the respective subcontexts. Every closure system $\mathcal{C}$ on S is in B, but not conversely. The closure system generated by a family $\mathcal{F}$ from B may contain additional sets, obtained as intersections of a set in G_0 and a set in G_1. However, such new sets can only be contained in G_0, since a set containing the element $n-1$ cannot be obtained as an intersection involving one not containing $n-1$.

Algorithm 5 encodes the subsets of $S := \{0, \ldots, n-1\}$ in the natural manner as the integers from 0 to $2^n - 1$, using the bitwise AND-operation for intersecting

sets. A closure system on S is represented by an array $F[0\,..\,2^n-2]$ with

$$F[i] := \begin{cases} 0 & \text{if } i \text{ is not closed} \\ 1 & \text{if } i \text{ is closed and reducible} \\ 2 & \text{if } i \text{ is closed and irrreducible.} \end{cases}$$

The algorithm starts with two closure systems on $\{0,\ldots,n-2\}$ and concatenates them. In addition, a random decision is made if the set $\{0,\ldots,n-2\}$ is to be counted as a closed set. The result is not necessarily a closure system and needs to be made intersection closed. This is done in the second part of the algorithm (beginning with "SUCCESS:= **true**"). Whenever a new reducible element is encountered, the generation process is abandoned with probability 0.5 and is started over. Only if this never happens, a closure system on $\{0,..,n-1\}$ is achieved for output.

Algorithm 5 RANDOM_MOORE(n): Random Moore family.

Input: An operator RANDOM_MOORE($n-1$) generating random Moore families on $\{0,\ldots,n-2\}$.
Output: A random Moore family on $\{0,\ldots,n-1\}$.

repeat
  $F_0 :=$ RANDOM_MOORE$(n-1)$
  $F[0\,..\,2^{n-1}-2] := F_0[0\,..\,2^{n-1}-2]$
  $F[2^{n-1}-1] := \begin{cases} 0 & \text{if RANDOM}(0,1] < 0.5 \\ 1 & \text{else.} \end{cases}$
  $F_1 :=$ RANDOM_MOORE$(n-1)$
  $F[2^{n-1}\,..\,2^n-2] := F_1[0\,..\,2^{n-1}-2]$
  $i := 2^n-1$
  SUCCESS := **true**
  while SUCCESS **and** $(i > 2^{n-1})$ **do**
    if $F[i] = 2$ **then**
      $j := 2^{n-1}-1$
      while SUCCESS **and** $(j > 0)$ **do**
        MEET := i AND j
        if $(j \neq$ MEET$)$ **and** $(F[\text{MEET}] \neq 1)$ **then**
          SUCCESS := (RANDOM$(0,1] < 0.5$)
          $F[\text{MEET}] := 1$
        $j := j-1$
    $i := i-1$
until SUCCESS
return F.

We have implemented Algorithm 5 for $n = 7$ and present first experimental results. Note that the number of random samples produced by this experiment is small compared to the number of closure systems: we have generated less than 0.000 000 000 000 4% of all closure systems on seven points.

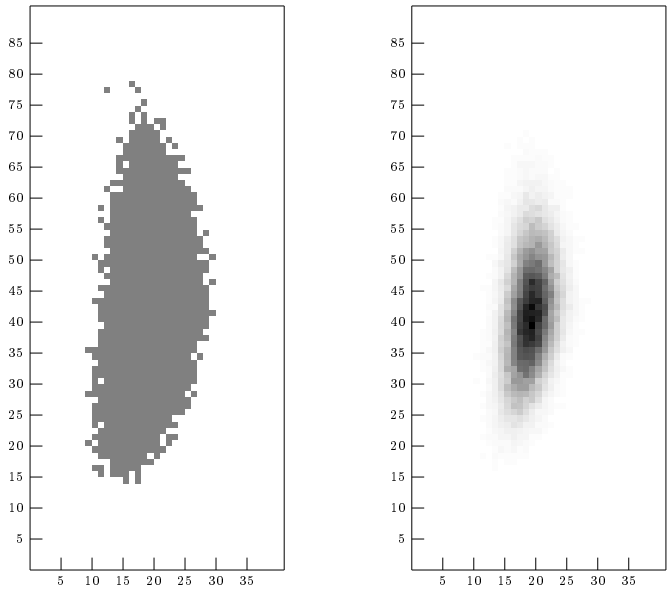


Fig. 2. A random sample of 50 000 closure systems on a seven element set, plotted according to their number of irreducible closed sets (horizontal) and reducible closed sets (vertical). The left image shows which sizes occurred at least once. The right image expresses higher frequencies by darker shadings.

The computation took one night on a 1.4 GHz PC. We did not even attempt to generate random closure systems on eight elements using Algorithm 5. We believe that a substantially better idea is needed for that case and beyond.

References

1. S. Asmussen and P. W. Glynn. *Stochastic Simulation.* Springer-Verlag, New York, 2007.
2. Mario Boley, Henrik Grosskreutz, and Thomas Gärtner. Formal concept sampling for counting and threshold–free local pattern mining. In *Proc. of the SIAM Int. Conf. on Data Mining (SDM 2010).* SIAM, 2010.
3. Pierre Colomb, Alexis Irlande, and Olivier Raynaud. Counting of Moore families for $n = 7$. In *ICFCA'10*, pages 72–87, 2010.
4. Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis - mathematical foundations.* Springer Verlag, 1999.
5. Michel Habib and Lhouari Nourine. The number of Moore families on $n = 6$. *Discrete Mathematics*, pages 291–296, 2005.
6. Albert Nijenhuis and Herbert S. Wilf. *Combinatorial algorithms.* Academic Press, 1975.

Extracting Decision Trees from Interval Pattern Concept Lattices

Zainab Assaghir[1], Mehdi Kaytoue[2], Wagner Meira Jr.[2] and Jean Villerd[3]

[1] INRIA Nancy Grand Est / LORIA, Nancy, France
[2] Universidade Federal de Minas Gerais, Belo Horizonte, Brazil
[3] Institut National de Recherche Agronomique / Ensaia, Nancy, France
Zainab.Assaghir@loria.fr, {kaytoue,meira}@dcc.ufmg.br,
Jean.Villerd@nancy.inra.fr

Abstract. Formal Concept Analysis (FCA) and concept lattices have shown their effectiveness for binary clustering and concept learning. Moreover, several links between FCA and unsupervised data mining tasks such as itemset mining and association rules extraction have been emphasized. Several works also studied FCA in a supervised framework, showing that popular machine learning tools such as decision trees can be extracted from concept lattices. In this paper, we investigate the links between FCA and decision trees with numerical data. Recent works showed the efficiency of "pattern structures" to handle numerical data in FCA, compared to traditional discretization methods such as conceptual scaling.

1 Introduction

Decision trees (DT) are among the most popular classification tools, especially for their readability [1]. Connexions between DT induction and FCA have been widely studied in the context of binary and nominal features [2], including structural links between decision trees and dichotomic lattices [8], and lattice-based learning [7]. However the numerical case faces issues regarding FCA and numerical data. In this paper, we investigate the links between FCA and decision trees with numerical data and a binary target attribute. We use an extension of Formal Concept Analysis called *interval pattern structures* to extract sets of positive and negative hypothesis from numerical data. Then, we propose an algorithm thats extract decision trees from minimal positive and negative hypothesis.

The paper is organised as follows. Section 2 presents the basics of FCA and one of its extensions called interval pattern structureq for numerical data. Section 3 recalls basic notions of decision trees. Then, we introduce some definitions in section 4 showing the links between interval pattern structures and decision trees, and a first algorithm for building decision trees from minimal positive and negative hypothesis extracted from the pattern structures.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

2 Pattern structures in formal concept analysis

Formal contexts and concept lattices. We assume that the reader is familiar with FCA, and recall here most important definitions from [3]. Basically, data are represented as a binary table called *formal context* (G, M, I) that represents a relation I between a set of objects G and a set of attributes M. The statement $(g, m) \in I$ is interpreted as "the object g has attribute m". The two operators $(\cdot)'$ define a Galois connection between the powersets $(2^G, \subseteq)$ and $(2^M, \subseteq)$, with $A \subseteq G$ and $B \subseteq M$:

$$A' = \{m \in M \mid \forall g \in A : gIm\} \qquad for\ A \subseteq G,$$
$$B' = \{g \in G \mid \forall m \in B : gIm\} \qquad for\ B \subseteq M$$

For $A \subseteq G$, $B \subseteq M$, a pair (A, B), such that $A' = B$ and $B' = A$, is called a *(formal) concept*. In (A, B), the set A is called the *extent* and the set B the *intent* of the concept (A, B). The set of all concepts is partially ordered by $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2$ $(\Leftrightarrow B_2 \subseteq B_1)$ and forms a complete lattice called the *concept lattice* of the formal context (G, M, I).

In many applications, data usually consist in complex data involving numbers, intervals, graphs, etc. (e.g. Table 1) and require to be conceptually scaled into formal contexts. Instead of transforming data, leading to representation and computational difficulties, one may directly work on the original data. Indeed, to handle complex data in FCA, Ganter & Kuznetsov [4] defined pattern structures: it consists of objects whose descriptions admit a *similarity operator* which induces a semi-lattice on data descriptions. Then, the basic theorem of FCA naturally holds. We recall here their basic definitions, and present interval pattern structures from [5] to handle numerical data.

Patterns structures. 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. Then $(G, (D, \sqcap), \delta)$ is called a *pattern structure*. Elements of D are called *patterns* and are ordered by a subsumption relation $\sqsubseteq$ such that given $c, d \in D$ one has $c \sqsubseteq d \Longleftrightarrow c \sqcap d = c$. A pattern structure $(G, (D, \sqcap), \delta)$ gives rise to the following derivation operators $(\cdot)^\square$, given $A \subseteq G$ and an interval pattern $d \in (D, \sqcap)$:

$$A^\square = \prod_{g \in A} \delta(g)$$

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

These operators form a Galois connection between $(2^G, \subseteq)$ and $(D, \sqsubseteq)$. *(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 called a *pattern intent* and is the common description of all objects in A, called *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 a *(pattern) concept lattice*.

Interval pattern structures. Pattern structures allow us to consider complex data in full compliance with FCA formalism. This requires to define a meet

operator on object descriptions, inducing their partial order. Concerning numerical data, an interesting possibility presented in [5] is to define a meet operator as an interval convexification. Indeed, one should realize that "similarity" or "intersection" between two real numbers (between two intervals) may be expressed in the fact that they lie within some (larger) interval, this interval being the smallest interval containing both two. Formally, given two intervals $[a_1, b_1]$ and $[a_2, b_2]$, with $a_1, b_1, a_2, b_2 \in \mathbb{R}$, one has:

$$[a_1, b_1] \sqcap [a_2, b_2] = [min(a_1, a_2), max(b_1, b_2)]$$
$$[a_1, b_1] \sqsubseteq [a_2, b_2] \Leftrightarrow [a_1, b_1] \supseteq [a_2, b_2]$$

The definition of $\sqcap$ implies that smaller intervals subsume larger intervals that contain them. This is counter intuitive referring to usual intuition, and is explained by the fact that $\sqcap$ behaves as an union (actually convex hull is the union of intervals, plus the holds between them).

These definitions of $\sqcap$ and $\sqsubseteq$ can be directly applied component wise on vectors of numbers or intervals, e.g. in Table 1 where objects are described by vectors of values, each dimension corresponding to an attribute. For example, $\langle [5, 7.2], [1, 1.8] \rangle \sqsubseteq \langle [5, 7], [1, 1.4] \rangle$ as $[5, 7.2] \sqsubseteq [5, 7]$ and $[1, 1.8] \sqsubseteq [1, 1.4]$.

Now that vectors of interval forms a $\sqcap$-semi-lattice, numerical data such as Table 1 give rise to a pattern structure and a pattern concept lattice. An example of application of concept forming operators $(.)^{\square}$ is given below. The corresponding pattern structure is $(G, (D, \sqcap), \delta)$ with $G = \{p_1, ..., p_4, n_1, ..., n_3\}$ and $d \in D$ is a vector with i^{th} component corresponding to attribute m_i.

$$\{p_2, p_3\}^{\square} = \delta(p_2) \sqcap \delta(p_3) = \langle [5, 5.9], [2, 3.2], [3.5, 4.8], [1, 1.8] \rangle$$
$$\{p_2, p_3\}^{\square\square} = \{p_2, p_3, p_4\}$$

As detailed in [5], vectors of intervals can be seen as hyperrectangles in Euclidean space: first $(.)^{\square}$ operator gives the smallest rectangle containing some object descriptions while second $(.)^{\square}$ operator returns the set of objects whose descriptions are rectangles included in the rectangle in argument. Accordingly, $(\{p_2, p_3, p_4\}, \langle [5, 5.9], [2, 3.2], [3.5, 4.8], [1, 1.8] \rangle)$ is a pattern concept. All pattern concepts of an interval pattern structure form a concept lattice. Intuitively, lowest concepts have few objects and "small" intervals while higher concepts have "larger" intervals. An example of such lattice is given later.

3 Decision trees

Among all machine leaning tools, decision trees [6, 1] are one of the most widely used. They belong to the family of supervised learning techniques, where data consist in a set of explanatory attributes (binary, nominal or numerical) that describe each object, called example, and one target class attribute that affects each example to a nominal class. Many extensions have been proposed, e.g. to consider a numerical class attribute (regression trees) or other particular cases depending on the nature of attributes. In this paper we focus on data consisting

	m_1	m_2	m_3	m_4	ϵ
p_1	7	3.2	4.7	1.4	+
p_2	5	2	3.5	1	+
p_3	5.9	3.2	4.8	1.8	+
p_4	5.5	2.3	4	1.3	+
n_1	6.7	3.3	5.7	2.1	-
n_2	7.2	3.2	6	1.8	-
n_3	6.2	2.8	4.8	1.8	-

Table 1: Example 1: numerical context with an external target attribute ϵ

of numerical explanatory attributes and a binary class attribute. The aim of decision tree learning is to exhibit the relation between explanatory attributes and the class attribute through a set of decision paths. A decision path is a sequence of tests on the value of explanatory attributes that is a sufficient condition to assign a new example to one of the two classes. A decision tree gathers a set of decision paths through a tree structure where nodes contain tests on explanatory attributes. Each node has two branches, the left (resp. right) corresponds to the next test if the new example passed (resp. failed) the current test. When there is no more test to perform, the branch points to a class label, that represents a leaf of the tree. The links between FCA and decision tree learning have been investigated in the case where explanatory attributes are binary [7–10, 2]. However, to our knowledge, no research has been carried out until now in the case of numerical explanatory attributes. In the next section, we show how pattern structures can be used to extract decision trees from numerical data with positive and negative examples.

4 Learning in interval pattern structures

In [7], S. Kuznetsov considers a machine learning model in term of formal concept analysis. He assumes that the cause of a target property resides in common attributes of objects sharing this property. In the following, we adapt this machine learning model to the case of numerical data.

Let us consider an interval pattern structure $(G, (D, \sqcap), \delta)$ with an external target property ϵ. The set of objects G (the training set) is partitioned into two disjoints sets: positive G_+ and negative G_-. Then, we obtain two different pattern structures $(G_+, (D, \sqcap), \delta)$ and $(G_-, (D, \sqcap), \delta)$.

Definition 1 (Positive hypothesis). *A positive hypothesis h is defined as an interval pattern of $(G_+, (D, \sqcap), \delta)$ that is not subsumed by any interval pattern of $(G_-, (D, \sqcap), \delta)$, i.e. not subsumed by any negative example. Formally, $h \in D$ is a positive hypothesis iff*

$$h^{\square} \cap G_- = \emptyset \quad \textit{and} \quad \exists A \subseteq G_+ \quad \textit{such that} \quad A^{\square} = h$$

Definition 2 (Negative hypothesis). *A negative hypothesis h is defined as an interval pattern of $(G_-, (D, \sqcap), \delta)$ that is not subsumed by any interval pattern*

of $(G_+, (D, \sqcap), \delta)$*, i.e. not subsumed by any positive example. Formally,* $h \in D$ *is a negative hypothesis iff*

$$h^{\square} \cap G_+ = \emptyset \quad \textit{and} \quad \exists A \subseteq G_- \quad \textit{such that} \quad A^{\square} = h$$

Definition 3 (Minimal hypothesis). *A positive (resp. negative) hypothesis* h *is minimal iff there is no positive (resp. negative) hypothesis* $e \neq h$ *such that* $e \sqsubseteq h$*.*

Going back to numerical data in Table 1, we now consider the external binary target property and split accordingly the object set into $G_+ = \{p_1, p_2, p_3, p_4\}$ and $G_- = \{n_1, n_2, n_3\}$. The pattern concept lattice of $(G_+, (D, \sqcap), \delta)$, where D is the semi-lattice of intervals and δ is a mapping associating for each object its pattern description is given in Figure 1 where positive hypothesis are marked. Note that neither the interval pattern $\langle [5.5, 7], [2.3, 3.2], [4, 4.8], [1.3, 1.8] \rangle$ nor $\langle [5, 7], [2, 3.2], [3.5, 4.8], [1, 1.8] \rangle$ are positive hypothesis since they are both subsumed by the interval pattern $\delta(n_3) = \langle [6.2, 6.2], [2.8, 2.8], [4.8, 4.8], [1.8, 1.8] \rangle$. Therefore, there are two minimal positive hypothesis: $P_1 = \langle [5, 7], [2, 3.2], [3.5, 4.7], [1, 1.4] \rangle$ and $P_2 = \langle [5, 5.9], [2, 3.2], [3.5, 4.8], [1, 1.8] \rangle$. From $(G_-, (D, \sqcap), \delta)$ (not shown), we obtain the unique minimal negative hypothesis: $N_1 = \langle [6.2, 7.2], [2.8, 3.3], [4.8, 6], [1.8, 2.1] \rangle$.

Now, we consider decision trees more formally. Let the training data be described by $\mathbb{K}_{+-} = (G_+ \cup G_-, (D, \sqcap), \delta)$ with the derivation operator denoted by $(.)^{\square}$. This operator is called *subposition* in term of FCA.

Definition 4 (Decision path). *A sequence* $\langle (m_1, d_1), (m_2, d_2), \ldots, (m_k, d_k) \rangle$*, for different attributes* $m_1, m_2, \cdots, m_k$ *chosen one after another, is a called decision path of length* k *if there is no* m_i *such that* $(m_i, d_i), (m_i, e_i)$ *and* d_i *and* e_i *are not comparable, and there exists* $g \in G_+ \cup G_-$ *such that* $\langle d_1, d_2, \ldots, d_k \rangle^{\square} \sqsubseteq \delta(g)$ *(i.e. there is at least one example* g *such that* $d_i \sqsubseteq \delta(g)$ *for each attribute* m_i*). For instance,* $\langle (m_3, [4.8, 6]), (m_1, [6.2, 7.2]) \rangle$ *is a decision path for Example 1.*

If $i \leq k$ (respectively $i < k$), the sequence $\langle (m_1, d_1), (m_2, d_2), \ldots, (m_i, d_i) \rangle$ is called subpath (proper subpath) of a decision path $\langle (m_1, d_1), (m_2, d_2), \ldots, (m_k, d_k) \rangle$.

Definition 5 (Full decision path). *A sequence* $\langle (m_1, d_1), (m_2, d_2), \ldots, (m_k, d_k) \rangle$*, for different attributes* $m_1, m_2, \ldots, m_k$ *chosen one after another, is called full decision path of length* k *if all object having* $(m_1, d_1), (m_2, d_2), \ldots, (m_k, d_k)$ *(i.e.* $\forall g \in G, d_i \sqsubseteq \delta(g)$ *for the attribute* m_i*) are either positive or negative examples (i.e. have either* $+$ *or* $-$ *value of the target attribute).*

We say that a full decision path is non-redundant if none of its subpaths is a full decision path. The set of all chosen attributes in a full decision path can be considered as a sufficient condition for an object to belong to a class $\epsilon \in \{+, -\}$. A decision tree is then defined as the set of full decision paths.

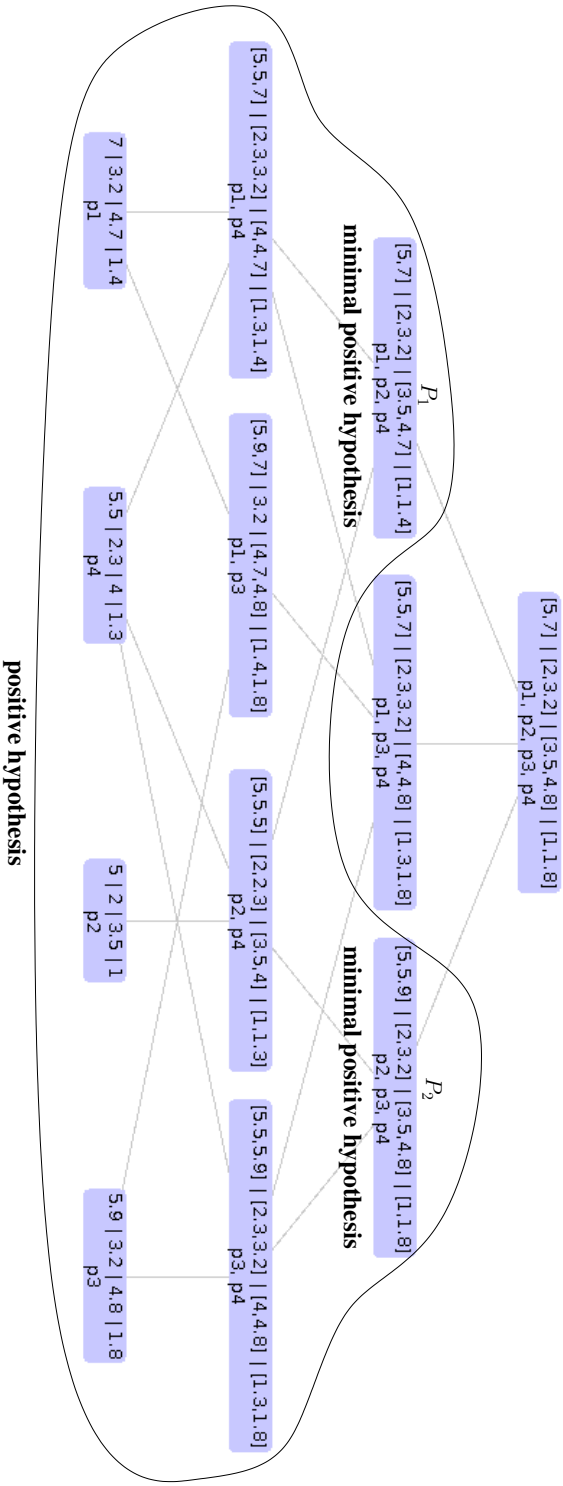


Fig. 1: Lattice of the pattern structure $(G_{+}, (D, \sqcap), \delta)$.

4.1 A first algorithm for building decision trees from interval pattern structures

In this section, we propose a first algorithm for extracting full decision paths from the sets of minimal positive hypothesis $\mathcal{P}$ and minimal negative hypothesis $\mathcal{N}$. Intuitively, minimal positive (resp. negative) hypothesis describe the largest areas in the attribute space that gathers the maximum number of positive (resp. negative) examples with no negative (resp. positive) example. Positive and negative areas may intersect on some dimensions. In Example 1 (see Table 1), $\mathcal{P} = \{P_1, P_2\}$ and $\mathcal{N} = \{N_1\}$ and we denote by $P_i \cap N_j$ the interval vector for which the k-the component is the intersection of the P_i and N_j intervals for the k-the component. Recall that $P_1 = \langle[5,7],[2,3.2],[3.5,4.7],[1,1.4]\rangle$, $P_2 = \langle[5,5.9],[2,3.2],[3.5,4.8],[1,1.8]\rangle$ and $N_1 = \langle[6.2,7.2],[2.8,3.3],[4.8,6],[1.8,2.1]\rangle$. Then we have:

$$P_1 \cap N_1 = \langle[6.2,7],[2.8,3.2],\emptyset,\emptyset\rangle$$

$$P_2 \cap N_1 = \langle\emptyset,[2.8,3.2],[4.8],[1.8]\rangle$$

We note that P_1 and N_1 have no intersection for attributes m_3 and m_4. This means that any example that has a value for m_3 (resp. m_4) that is contained in P_1's interval for m_3 (resp. m_4) can directly be classified as positive. Similarly, any example having a value for m_3 (resp. m_4) contained in N_1's interval for m_3 (resp. m_4) can directly be classified as negative. The same occurs for P_2 and N_1 for m_1.

Therefore a full decision path for a minimal positive hypothesis P is defined as a sequence $\langle(m_i, m_i(P))\rangle_{i\in\{1\ldots|\mathcal{N}|\}}$ where m_i is an attribute such that $m_i(P \cap N_i) = \emptyset$[4]. A full decision path for a minimal negative hypothesis N is defined as a sequence $\langle(m_j, m_j(N))\rangle_{j\in\{1\ldots|\mathcal{P}|\}}$ where m_j is an attribute such that $m_j(N \cap P_i) = \emptyset$.

Here examples of such decision paths (built from P_1, P_2 and N_1 respectively) are:

$$\langle(m_3,[3.5,4.7])\rangle(P_1)$$

$$\langle(m_4,[1,1.4])\rangle(P_1)$$

$$\langle(m_1,[5,5.9])\rangle(P_2)$$

$$\langle(m_3,[4.8,6]),(m_1,[6.2,7.2])\rangle(N_1)$$

$$\langle(m_4,[1.8,2.1]),(m_1,[6.2,7.2])\rangle(N_1)$$

Decision paths built from P_1 and P_2 are sequences that contain a single element since $|\mathcal{N}| = 1$. Decision paths built from N_1 are sequences that contain two elements since $|\mathcal{P}| = 2$. Two distinct full decision paths can be built from P_1 since there are two attributes for which P_1 and N_1 do not intersect.

A positive (resp. negative) decision tree is therefore a set of full decision paths, one for each minimal positive (resp.negative) hypothesis. For instance:

[4] For any interval pattern P, the notation $m_i(P)$ denotes its interval value for the attribute m_i.

"if $m_3 \in [3.5, 4.7]$ *then +, else if* $m_1 \in [5, 5.9]$ *then + else -"* is an example of positive decision path. An example of negative decision path is *"if* $m_1 \in [6.2, 7.2]$ *and* $m_3 \in [4.8, 6]$ *then -, else +"*.

Algorithm 1 describes the computation of full decision paths for minimal positive hypothesis. The dual algorithm for minimal negative hypothesis is obtained by interchanging $\mathcal{P}$ and $\mathcal{N}$.

1 $Res \leftarrow$ empty array of size $|\mathcal{P}|$;
2 **foreach** $P \in \mathcal{P}$ **do**
3 **foreach** $N \in \mathcal{N}$ *such that* $\exists m_i, m_i(P \cap N) = \emptyset$ **do**
4 **if** $(m_i, m_i(P)) \notin Res[P]$ **then**
5 $Res[P] \leftarrow Res[P] \cup (m_i, m_i(P))$;
6 **foreach** $N \in \mathcal{N}$ *such that* $\nexists m_i, m_i(P \cap N) = \emptyset$ **do**
7 $Res[P] \leftarrow Res[P] \cup \{\bigvee_{m \in M}(m, m(P) \setminus m(P \cap N))\}$;

Algorithm 1: Modified algorithm for extracting full decision paths *Res* (including non-redundant) for minimal positive hypothesis

The different steps of the algorithm are detailed below:

line 1: *Res* will contain a full decision path for each minimal positive hypothesis.
line 2: Process each minimal positive hypothesis P.
line 3: For each minimal negative hypothesis N that has at least one attribute m such that $m(P \cap N) = \emptyset$, choose one of these attribute, called m_i below.
line 4: Ensure that m_i has not already been selected for another N, this enables to produce non redundant full decision paths (see Example 2).
line 5: Add the interval $m_i(P)$ in the full decision path of P. The test $m_i \in m_i(P)$ will separate between positive examples covered by P and negative examples covered by N.
line 6: For each minimal negative hypothesis N that has no attribute m such that $m(P \cap N) = \emptyset$.
line 7: Positive examples covered by P and negative examples covered by N can be separated by a disjunction of tests $m \in m(P) \setminus m(P \cap N))$ on each attribute m. Hence, there is at least one attribute for which a positive example from P belongs to $m(P)$ and not to $m(N)$. Otherwise, N would not be a negative hypothesis.

Note that Example 1 is a particular case where all negative examples are gathered in a unique minimal negative hypothesis.

A few values have been modified in Table 2 in order to produce two minimal negative hypothesis.

	m_1	m_2	m_3	m_4	ϵ
p_1	7	3.2	4.7	1.4	+
p_2	5	2	3.5	1	+
p_3	5.9	3.2	4.8	1.8	+
p_4	5.5	2.3	4	1.3	+
n_1	5.9	3.3	5.7	1.4	-
n_2	7.2	3.2	6	1.8	-
n_3	6.2	2.8	4.8	1.8	-

Table 2: Example 2: training set

Minimal positive hypothesis P_1 and P_2 remain unchanged while there are two minimal negative hypothesis:

$$N_1 = \langle [5.9, 7.2], [3.2, 3.3], [5.7, 6], [1.4, 1.8] \rangle$$

$$N_2 = \langle [6.2, 7.2], [2.8, 3.2], [4.8, 6], [1.8, 1.8] \rangle$$

This leads to the following intersections:

$$P_1 \cap N_1 = \langle [5.9, 7], [3.2], \emptyset, [1.4] \rangle$$

$$P_1 \cap N_2 = \langle [6.2, 7], [2.8, 3.2], \emptyset, \emptyset \rangle$$

$$P_2 \cap N_1 = \langle [5.9], [3.2], \emptyset, [1.4, 1.8] \rangle$$

$$P_2 \cap N_2 = \langle \emptyset, [2.8, 3.2], [4.8, 4.8], [1.8] \rangle$$

Examples of full decision path computed by Algorithm 1 from P_1 are

$$\langle (m_3, [3.5, 4.7]), (m_4, [1, 1.4]) \rangle (1)$$

$$\langle (m_3, [3.5, 4.7]), (m_3, [3.5, 4.7]) \rangle (2)$$

Note that neither N_1 nor N_2 intersect P_1 on m_3, therefore the full decision path (2) can be simplified as $\langle (m_3, [3.5, 4.7]) \rangle$. More generally, following previous definitions, $\langle (m_3, [3.5, 4.7]) \rangle$ is a non-redundant full decision path while $\langle (m_3, [3.5, 4.7]), (m_4, [1, 1.4]) \rangle$ and $\langle (m_3, [3.5, 4.7]), (m_3, [3.5, 4.7]) \rangle$ are not. A conditional test has been added in Algorithm 1 in order to also produce such non-redundant full decision paths.

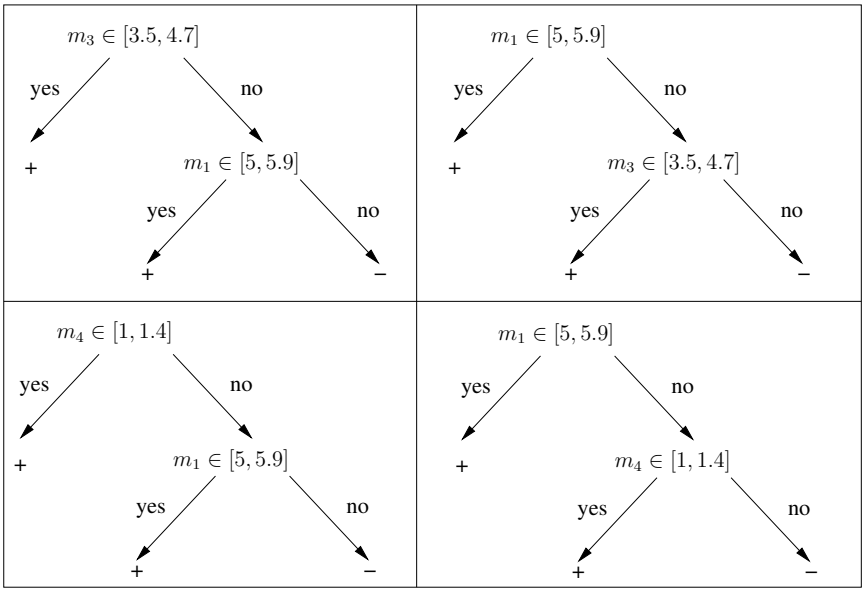
Finally a concrete positive decision tree is built from the set of full decision paths, each node corresponds to a minimal positive hypothesis P_i and contains a test that consists in the conjunction of the elements of a full decision path. The left child contains + and the right child is a node corresponding to another minimal positive hypothesis P_j or - if all minimal positive hypothesis have been processed.

An example of decision tree for example 2 is: *"if* $m_3 \in [3.5, 4.7]$ *and* $m_4 \in [1, 1.4]$ *then +, else (if* $m_3 \in [3.5, 4.8]$ *and* $m_1 \in [5, 5.9]$ *then +, else -)"*.

We detail below the complete process for examples 1 and 2.

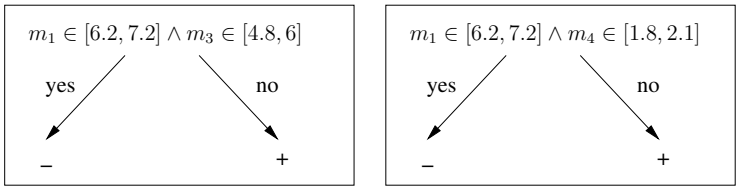
4.2 Example 1

$\mathcal{P}$	$P_1 = \langle [5,7], [2,3.2], [3.5,4.7], [1,1.4] \rangle$ $P_2 = \langle [5,5.9], [2,3.2], [3.5,4.8], [1,1.8] \rangle$
$\mathcal{N}$	$N_1 = \langle [6.2,7.2], [2.8,3.3], [4.8,6], [1.8,2.1] \rangle$
intersections	$P_1 \cap N_1 = \langle [6.2,7], [2.8,3.2], \emptyset, \emptyset \rangle$ $P_2 \cap N_1 = \langle \emptyset, [2.8,3.2], [4.8], [1.8] \rangle$
full decisions paths from $\mathcal{P}$	$\langle (m_3, [3.5,4.7]) \rangle (P_1)$ $\langle (m_4, [1,1.4]) \rangle (P_1)$ $\langle (m_1, [5,5.9]) \rangle (P_2)$
full decisions paths from $\mathcal{N}$	$\langle (m_3, [4.8,6]), (m_1, [6.2,7.2]) \rangle (N_1)$ $\langle (m_4, [1.8,2.1]), (m_1, [6.2,7.2]) \rangle (N_1)$



full paths from P1, then from P2

full paths from P2, then from P1



full paths from N1

Fig. 2: Decision trees built from Example 1

4.3 Example 2

$\mathcal{P}$	$P_1 = \langle[5,7],[2,3.2],[3.5,4.7],[1,1.4]\rangle$ $P_2 = \langle[5,5.9],[2,3.2],[3.5,4.8],[1,1.8]\rangle$
$\mathcal{N}$	$N_1 = \langle[5.9,7.2],[3.2,3.3],[5.7,6],[1.4,1.8]\rangle$ $N_2 = \langle[6.2,7.2],[2.8,3.2],[4.8,6],[1.8,1.8]\rangle$
intersections	$P_1 \cap N_1 = \langle[5.9,7],[3.2],\emptyset,[1.4]\rangle$ $P_1 \cap N_2 = \langle[6.2,7],[2.8,3.2],\emptyset,\emptyset\rangle$ $P_2 \cap N_1 = \langle[5.9],[3.2],\emptyset,[1.4,1.8]\rangle$ $P_2 \cap N_2 = \langle\emptyset,[2.8,3.2],[4.8,4.8],[1.8]\rangle$
full decisions paths from $\mathcal{P}$	$\langle(m_3,[3.5,4.7])\rangle(P_1)$ $\langle(m_3,[3.5,4.7]),(m_4,[1,1.4])\rangle(P_1)$ (redundant) $\langle(m_3,[3.5,4.8]),(m_1,[5,5.9])\rangle(P_2)$
full decisions paths from $\mathcal{N}$	$\langle(m_3,[5.7,6])\rangle(N_1)$ $\langle(m_3,[4.8,6]),(m_1,[6.2,7.2])\rangle(N_2)$ $\langle(m_4,[1.8]),(m_1,[6.2,7.2])\rangle(N_2)$

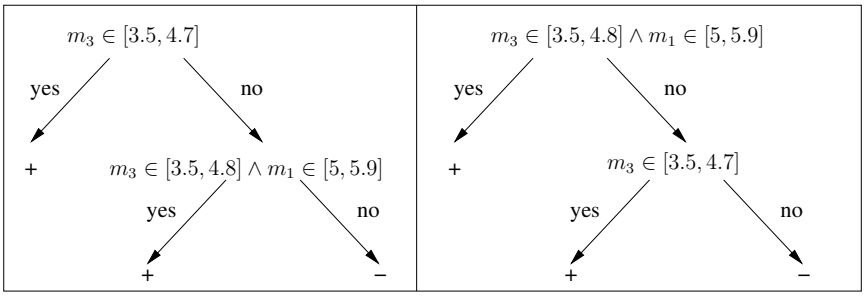
4.4 Comparison with traditional decision tree learning approaches

Standard algorithms such as C4.5 produce decision trees in which nodes contain tests of the form $a \leq v$, i.e. the value for attribute a is less or equal to v, while our nodes contain conjunctions of tests of the form $a \in [a_1, a_2] \wedge b \in [b_1, b_2]$. A solution consists in identifying minimal and maximal values for each attribute in the training set, and by replacing them by $-\infty$ and $+\infty$ respectively in the resulting trees (see Figure 4). Moreover, common decision tree induction techniques use Information Gain maximization (or equivalently conditional entropy minization) to choose the best split at each node. The conditional entropy of a split is null when each child node is pure (contains only positive or negative examples). When this perfect split can not be expressed as an attribute-value test, it can be shown that the optimal split that minimize conditional entropy consists in maximizing the number of examples in one pure child node (proof is ommited due to space limitation). This optimal split exactly matches our notion of positive (resp. negative) minimal hypothesis, which corresponds to descriptions that gathers the maximum number of only positive (resp. negative) examples.

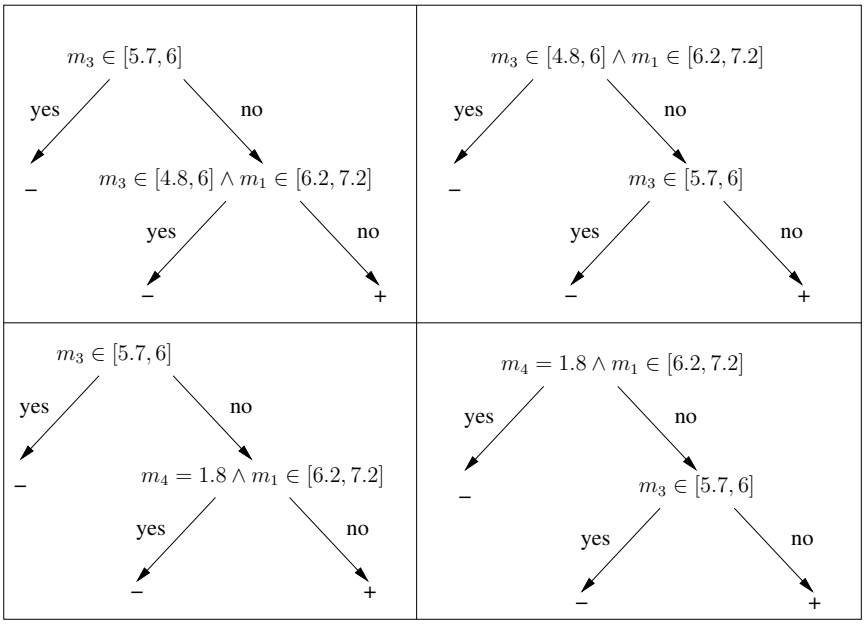
However we insist that our algorithm is only a first and naive attempt to produce decision trees from multi-valued contexts using pattern structures. Its aim is only to clarify the links between decision tree learning and pattern structures. Therefore it obviously lacks of relevant data structures and optimization. However we plan to focus our efforts on algorithm optimization and then on rigorous experimentations on standard datasets.

5 Concluding remarks

In this paper, we studied the links between decision trees and FCA in the particular context of numerical data. More precisely, we focused on an extension



full paths from P1, then from P2 | full paths from P2, then from P1



full paths from N1, then from N2 | full paths from N2, then from N1

Fig. 3: Decision trees built from Example 2

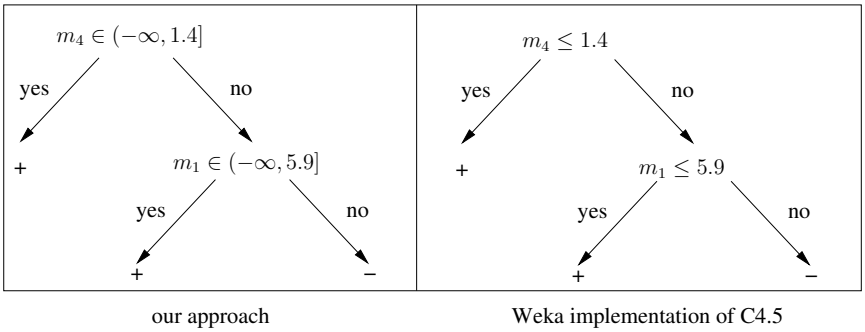


Fig. 4: Comparison of decisions trees produced by our approach and by C4.5 for Example 1

of FCA for numerical data called interval pattern structures, that has recently gained popularity through its ability to handle numerical data without any discretization step. We showed that interval pattern structures from positive and negative examples are able to reveal positive and negative hypothesis, from which decision paths and decision trees can be built.

In future works, we will focus on a comprehensive and rigorous comparison of our approach with traditional decision tree learning techniques. Moreover, we will study how to introduce in our approach pruning techniques that avoid overfitting. We will also investigate solutions in order to handle nominal class attributes (i.e. more than two classes) and heterogeneous explanatory attributes (binary, nominal, ordinal, numerical). Finally, notice that interval patterns are closed since $(.)^{\square\square}$ is a closure operator. In a recent work [11], it has been shown that whereas a closed interval pattern represents the smallest hyper-rectangle in its equivalence class, interval pattern generators represent the largest hyper-rectangles. Accordingly, generators are favoured by minimum description length principle (MDL), since being less constrained. An interesting perspective is to test their effectiveness to describe minimal hypothesis in the present work.

References

1. Quinlan, J.: Induction of decision trees. Machine learning **1**(1) (1986) 81–106
2. Fu, H., Njiwoua, P., Nguifo, E.: A comparative study of fca-based supervised classification algorithms. Concept Lattices (2004) 219–220
3. Ganter, B., Wille, R.: Formal Concept Analysis. Springer-Verlag (1999)
4. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: ICCS '01: Proceedings of the 9th International Conference on Conceptual Structures, Springer-Verlag (2001) 129–142
5. Kaytoue, M., Kuznetsov, S.O., Napoli, A., Duplessis, S.: Mining gene expression data with pattern structures in formal concept analysis. Inf. Sci. **181**(10) (2011) 1989–2001
6. Breiman, L.: Classification and regression trees. Chapman & Hall/CRC (1984)

7. Kuznetsov, S.O.: Machine learning and formal concept analysis. Int. Conf. on Formal Concept Analysis, LNCS 2961, (2004) 287–312
8. Guillas, S., Bertet, K., Ogier, J.: A generic description of the concept lattices classifier: Application to symbol recognition. Graphics Recognition. Ten Years Review and Future Perspectives (2006) 47–60
9. Nijssen, S., Fromont, E.: Mining optimal decision trees from itemset lattices. In: Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining, ACM (2007) 530–539
10. Nguifo, E., Njiwoua, P.: Iglue: A lattice-based constructive induction system. Intelligent data analysis **5**(1) (2001) 73
11. Kaytoue, M., Kuznetsov, S.O., Napoli, A.: Revisiting Numerical Pattern Mining with Formal Concept Analysis. In: International Joint Conference on Artificial Intelligence (IJCAI), Barcelona, Espagne (2011)

A New Formal Context for Symmetric Dependencies

Jaume Baixeries

Departament de Llenguatges i Sistemes Informàtics.
Universitat Politècnica de Catalunya.
08024 Barcelona. Catalonia.
jbaixer@lsi.upc.edu

Abstract. In this paper we present a new formal context for symmetric dependencies. We study its properties and compare it with previous approaches. We also discuss how this new context may open the door to solve some open problems for symmetric dependencies.

1 Introduction and Motivation

In database theory there are different types of dependencies, yet, two of them appear to be the most popular: functional dependencies and multivalued dependencies. The reason is that both dependencies come handy in order to explain the normalization of a database scheme. But some of these dependencies are not only confined to the database domain. For instance, implications (the equivalent of functional dependencies for binary data) are present in datamining and learning ([4,19,20]).

In general terms, a dependency states a relationship between sets of attributes in a table. Let us suppose that we have the following set of attributes: $\mathcal{U} = \{name, income, age\}$ in a table that contains the following records:

id	**Name**	**Income**	**Age**
1	Smith	30.000	26-10-1956
2	Hart	35.000	14-02-1966
3	Smith	30.000	02-01-1964

In such a case, we have that the relationship between *age* and the attributes *income* and *name* is functional, this is, that given a value of *age*, the value of *income* and *name* can be determined. We also have that the value of *name* can be determined by *income* and viceversa. In such a case, given these functional relationships between the attributes, we say that the functional dependencies $age \rightarrow \{name, income\}$, $name \rightarrow income$ and $income \rightarrow name$ hold in that table.

Functional dependencies and multivalued dependencies have their own set of axioms ([9,21]), which state what dependencies hold in the presence of other dependencies. For instance, an axiom for functional dependencies states that

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

transitivity holds, which means that, in the previous case, if we had that $name \rightarrow income$ and $income \rightarrow age$ hold in that table (which is not true in that table, but just as a supposition), it must follow necessarily that $name \rightarrow age$ holds. Given a set of dependencies Σ, we define as Σ^+ the set of all dependencies that hold according to those axioms.

These axioms, in turn, are also shared by other dependencies: implications share the same axioms of functional dependencies ([4]), and degenerate multivalued dependencies share the same axioms of multivalued dependencies ([5]). That is why we generically call Armstrong Dependencies (AD) those dependencies that share the same axioms of the former, and Symmetric Dependencies (SD) those that share the axioms of the latter.

Since in this paper we are focusing on the syntactical properties of those dependencies, we will only talk of Armstrong and symmetric dependencies, rather than functional or multivalued dependencies.

The lattice characterization of a set of Armstrong dependencies has been widely studied in [10,11,13,14,15], and their characterization with a formal context in [7,17]. However, the lattice characterization of symmetric dependencies has not been so widely studied. The main work is in [12], and the characterization of symmetric dependencies with a formal contexts was studied in [3,5] (we talk indistinctly of a lattice characterization and a characterization with a formal context). In the case of AD's, the formal context yields a powerset lattice ([17]), whereas in the case of symmetric dependencies, it yields a partition lattice ([3]).

The fact that some problems related to AD's have been solved using their lattice characterization, suggests that the same problems for SD's could also be solved using their corresponding lattice characterization. We name three of those problems already solved for AD's, not yet for SD's: **learning** SD's, the **finding of a minimal basis** for a set of dependencies for SD's and the **characterization of mixed sets** of SD's and AD's.

In general terms, **query learning** consists in deducing a function (a formula) via membership queries to an oracle. This method has been used to learn sets of Horn clauses, which can also be seen as implications ([8]), or, more generally, sets of Armstrong dependencies. Thus, the same general algorithm for learning Horn clauses ([1]) has been adapted to learn Armstrong dependencies ([2]). This adaptation was obviously easied by the fact that Horn clauses and Armstrong dependencies share the same set of axioms. Yet, no such algorithm for symmetric dependencies exists (to the best of the author's knowledge).

The **minimal base** (also: Duquenne-Guigues basis [16]) is the minimal set of Armstrong dependencies needed to compute Σ^+. In [11], [16] and [17] it is characterized and computed in terms of the (powerset) lattice characterization of Σ^+.

We have been dealing with unmixed sets of AD's and SD's, but there exists an **axiomatizations of mixed sets of AD's and SD's** ([21]), but no lattice characterization of mixed sets.

Although AD's and SD's are related, the lattice characterization yielded by the formal context in [3] is quite different in nature to that for AD's. *Potentially*, it may pose different problems. The first is that the solutions that have been found for AD's (based on their lattice characterization) may not be applied directly to the case of SD's. We do not mean that having AD's characterized with a powerset lattice and SD's with a partition lattice makes it impossible to solve the same problems for SD's. What we mean is that having a similar characterization for SD's would make it easier to try and find an answer using existing solutions for AD's.

A second drawback is that the size of the formal context for SD's is much larger, in comparison with that for AD's. This may cause a problem in case the context is used in practical applications, but, more importantly, there are partitions that play no rôle in that characterization. A simple analysis of [3] yields that partitions that contain no singleton are completly useless, but a more detailed analysis (out of the scope of this paper) indicates that there are more redundant partitions.

Finally, although partitions may be intuitive when dealing with SD's, *they do not reflect the $B \Leftrightarrow \neg B$ symmetry of the definition of symmetric dependencies* (as stated by Alan Day in [12]). It seems that the connection between AD's and SD's is stronger than what the partition lattice characterization suggests.

As a step towards solving the learning problem and computation of a minimal basis for SD's as well as the characterization of mixed sets of SD's and AD's, in this paper we present a new formal context for symmetric dependencies, following the work started in [3]. The results presented in this paper parallel those results, but from a different perspective that, we think, improve both the understanding and the possibilities to solve the open problems previously listed.

This paper starts with the Notation section, followed by a Previous Work section that explains the departing point of this paper. In the Results section, we present a new formal context for SD's. We also present an example in a separate section to illustrate the results. Finally, we discuss some aspects of this new formal context and present the conclusions and future work.

2 Notation

We depart from a set of attributes $\mathcal{U}$. We use non capital letters for single elements of that set, starting with $a, b, c, \ldots$, and capital letters for subsets of $\mathcal{U}$.

The complement of a set $X \subseteq \mathcal{U}$ is $\overline{X}$. We drop the union operator and use juxtaposition to indicate set union. For instance, instead of $X \cup Y$ we write XY. Generally, we also drop the set notation, and write abc instead of $\{ a, b, c \}$.

We define the powerset of a set $\mathcal{U}$ as $\wp(\mathcal{U})$. The set of partitions that can be formed with $\mathcal{U}$ is $\mathrm{Part}(\mathcal{U})$. The notation for a partition is $P = [P_1 \mid P_2 \mid \cdots \mid P_n]$, where P_i are the classes (subsets) of P. If needed, we indicate that the attributes in a set X are in fact a set of singletons with this notation: $\underline{X}$. For instance, $\underline{\{ a, b, c, d \}} = \{ \{ a \}, \{ b \}, \{ c \}, \{ d \} \}$. We overload $P \geq Q$ to indicate that a

partition P refines a partition Q and $P \leq Q$ to indicate that P is coarser than Q. More details of this (reversed) order can be found in [18].

As for Formal Concept Analysis, we use the usual notation ([17]), which includes the use of $'$ as the (overloaded) function that relates the set of attributes and that of objects and viceversa.

2.1 Symmetric Dependencies

A symmetric dependency is a relation between two sets of attributes, and it is stated as $X \Rightarrow Y$. Given a set of attributes $\mathcal{U}$, we define $SD_{\mathcal{U}}$ as the set of all symmetric dependencies that can be formed with $\mathcal{U}$. Although they will only be mentioned in this paper, we say that $X \rightarrow Y$ is an Armstrong dependency. Given a set of SD's $\Sigma \subseteq SD_{\mathcal{U}}$, we say that the **closure** of Σ is Σ^+, and consists of Σ plus the set of all SD's that can be derived from Σ applying recursively the following axioms:

Definition 1 (Axioms for SD's).

1. *Reflexivity: If $Y \subseteq X$, then, $X \Rightarrow Y$ holds.*
2. *Complementation: If $X \Rightarrow Y$ holds, then, $X \Rightarrow \overline{XY}$ holds.*
3. *Augmentation: If $X \Rightarrow Y$ holds and $W' \subseteq W \subseteq \mathcal{U}$, then, $XW \Rightarrow YW'$ holds.*
4. *Transitivity: If $X \Rightarrow Y$ and $Y \Rightarrow Z$ hold, then, $X \Rightarrow Z \setminus Y$ holds.*

Because of complementation, we give a symmetric dependency as $X \Rightarrow Y \mid Z$, where $Z = \overline{XY}$. We always assume that the rightest set in the right-hand side of a symmetric dependency is the complementary of the union of the other two. However, sometimes we will state it explicitly, as in $X \Rightarrow Y \mid \overline{XY}$ and sometimes we will simply use $X \Rightarrow Y \mid Z$. In both cases, X is the left-hand side of the dependency, Y its first right-hand side, and Z its second right-hand side. The set $SD_{\mathcal{U}}$ is the set of all non-trivial symmetric dependencies that can be formed using all the attributes in $\mathcal{U}$. By non-trivial we mean those SD's $X \Rightarrow Y \mid Z$ such that:

Definition 2. *A symmetric dependency $X \Rightarrow Y \mid Z$ is non-trivial if:*

1. $X \cup Y \cup Z = \mathcal{U}$.
2. $X \cap Y = X \cap Z = Y \cap Z = \emptyset$.
3. $X \neq \emptyset$, $Y \neq \emptyset$, $Z \neq \emptyset$.

As it can be seen, according to the axioms for symmetric dependencies, this limitation incurs in no loss of information, since the remaining symmetric dependencies can easily be derived from $SD_{\mathcal{U}}$ ([21]).

It is precisely the complementation rule that states the relation between Armstrong dependencies and symmetric dependencies. Broadly speaking, we could say that a symmetric dependency $X \Rightarrow Y \mid Z$ is equivalent to the fact that either the Armstrong dependencies $X \rightarrow Y$ or $X \rightarrow Z$ hold. This is a too general statement, but if, as an example, we take, functional dependencies and its symmetric

counterpart, degenerate multivalued dependencies, we see that the definition of a functional dependency $X \to Y$ states that whenever two tuples agree on X they also agree on Y, whereas the definition of a degenerate multivalued dependency $X \Rightarrow Y \mid Z$ states that whenever two tuples agree on X they also agree on Y **or** they agree in Z. In fact, there are also a set of two axioms in the case we are dealing with mixed sets of AD's and SD's. One of this axioms state that if $X \to Y$ holds, then, $X \Rightarrow Y \mid \overline{XY}$ holds as well. This example is just to indicate that the relationship between AD's and SD's is strong, and that SD's can be as a generalization of AD's.

Given a set of symmetric dependencies Σ, we say that the **dependency basis** of a set of attributes $X \subseteq \mathcal{U}$ (that is: $DB_\Sigma(X)$) is the coarsest partition of $\mathcal{U}$ such that all the dependencies $X \Rightarrow Y \mid Z$ that hold in Σ^+ are those such that Y (symmetrically Z) is the union of one or more classes of $DB(X)$. This partition always exists ([21]) and defines all the symmetric dependencies that hold in Σ^+ such that their left-hand side is X.

We also have that, since reflexivity holds for SD's, all the attributes of $X \subseteq \mathcal{U}$ are singletons in $DB_\Sigma(X)$.

2.2 Previous Work

The origins of defining a formal context to characterize the closure of a set of Armstrong dependencies started in [17]. This formal context was defined as:

$$\mathbb{K}_{AD}(\mathcal{U}) = (AD_\mathcal{U}, \wp(\mathcal{U}), I)$$

where $AD_\mathcal{U}$ is the set of Armstrong dependencies that can be formed with the set of attributes $\mathcal{U}$, and I was a binary relation between an Armstrong dependency and a set of attributes.

In [3], it was presented a formal context for symmetric dependencies with identical properties:

$$\mathbb{K}_{SD}(\mathcal{U}) = (SD_\mathcal{U}, \mathrm{Part}(\mathcal{U}), I')$$

The relations I and I' are generically called "respect" relations: a set of attributes (a partition) respects an Armstrong (symmetric) dependency.

Both formal contexts, in spite of its obvious structural differences, characterized the closure of a set of dependencies of its kind. In fact, both contexts provided the following results for each respective kind of dependencies:

1. $\Sigma^+ = \Sigma''$.
2. Σ' is the lattice characterization of Σ^+.

When we say that Σ' was the lattice characterization of Σ^+, it may seem redundant, since we already have that $\Sigma^+ = \Sigma''$. What we mean is that Σ' alone, without the application of the operator $'$, also characterized all the dependencies of Σ^+. This was done with the definition of a closure operator on Σ':

$$\Gamma_{\Sigma'}(X) = \bigwedge \{ Y \in \Sigma' \mid Y \supseteq X \}$$

The fact that this function is total indicates that $\wedge$ is always defined in Σ'. Depending on the formal context we were dealing with, we would have that $X \in \wp(\mathcal{U})$ (AD's) or that $X \in \mathrm{Part}(\mathcal{U})$ (SD's). In the case of Armstrong dependencies, we would then have that $X \to Y \in \Sigma^+$ if and only if:

$$\Gamma_{\Sigma'}(X) = \Gamma_{\Sigma'}(XY)$$

In the case of a symmetric dependency, it is a little bit more elaborated from a syntactical point of view, yet, equivalent to the previous case: $X \Rightarrow Y \mid Z \in \Sigma^+$ if and only if:

$$\Gamma_{\Sigma'}([\underline{X} \mid YZ]) = \Gamma_{\Sigma'}([\underline{X} \mid Y \mid Z])$$

Clearly, Σ' alone gives us the information of which dependencies are in Σ^+ by querying the (closure) operator $\Gamma_{\Sigma'}$. In both cases, and oversimplifying, we can say that a dependency holds in Σ^+ if and only if there is some kind of relationship between its left-hand side and its right-hand side, being this relationship defined by the formal context.

3 Results

The results in this paper try to overcome the potential problems that may represent the differnet nature of the current formal contexts for AD's and SD's (powerset versus partitions), as well as the larger size of a partition set, by presenting a characterization of symmetric dependencies based on a formal context whose set of attributes is the powerset of $\mathcal{U}$ instead of its partitions. This context will generalize that for AD's in [17] as it will seen in Section 5.

We define a formal context, and prove that it characterizes the set of symmetric dependencies Σ^+, in a way similar to that in [3]: $(SD_{\mathcal{U}}, \wp(\mathcal{U}),\ \mathrm{I}\ )$, where the relation I is defined as follows:

Definition 3. *$A \subseteq \mathcal{U}$ **respects** a symmetric dependency $X \Rightarrow Y \mid Z$ (that is: $X \Rightarrow Y \mid Z\ I\ A$) if and only if:*

$$A \not\supseteq X \text{ or } A \supseteq XY \text{ or } A \supseteq XZ$$

We have that $\Sigma' \subseteq \wp(\mathcal{U})$. As a trivial consequence of Definition 3 we have the following proposition:

Proposition 1. *$X \Rightarrow Y \mid Z \in \Sigma''$ if and only if*

$$\nexists A \in \Sigma' : A \supseteq X \text{ and } A \not\supseteq XY \text{ and } A \not\supseteq XZ$$

We now study the properties of this contexts and how they characterize Σ^+. We first see that all the dependencies that are in Σ^+ are also present in Σ''. To prove this claim, we must prove axiom by axiom that the dependencies derived by those axioms are also present in Σ'', but since reflexivity and complementation are trivial, we only prove augmentation and transitivity.

Proposition 2 (Augmentation). *If $X \Rightarrow Y \mid \overline{XY} \in \Sigma$, and $W' \subseteq W$ then, $XW \Rightarrow YW' \mid \overline{XWY} \in \Sigma''$.*

Proof. By the way of contradiction, we suppose that there is a set $A \subseteq \mathcal{U}, A \in \Sigma'$ such that (note that $XW\overline{XYW} = XW\overline{Y}$)

$$A \supseteq XW \text{ and } A \not\supseteq XWY \text{ and } A \not\supseteq XW\overline{Y}$$

Since $X \Rightarrow Y \mid \overline{XY} \in \Sigma$, we have that $A \not\supseteq X$ or $A \supseteq XY$ or $A \supseteq X\overline{Y}$ (because $X\overline{XY} = X\overline{Y}$). We have that $A \supseteq XW$ discards $A \not\supseteq X$. So, we only have two possible options:

(i) $A \supseteq XY$, which in combination with $A \supseteq XW$ yields $A \supseteq XYW$, which contradicts $A \not\supseteq XWY$.
(ii) $A \supseteq X\overline{Y}$, which in combination with $A \supseteq XW$ yields $A \supseteq XW\overline{Y}$, which contradicts $A \not\supseteq XW\overline{Y}$.

■

Proposition 3 (Transitivity). *If $X \Rightarrow Y \mid \overline{XY} \in \Sigma$ and $Y \Rightarrow Z \mid \overline{YZ} \in \Sigma$, then, $X \Rightarrow Z \setminus Y \mid \overline{X(Z \setminus Y)} \in \Sigma$.*

Proof. By the way of contradiction, we suppose that there is $A \subseteq \mathcal{U}, A \in \Sigma'$ such that

$$A \supseteq X \text{ and } A \not\supseteq X(Z \setminus Y) \text{ and } A \not\supseteq X\overline{X(Z \setminus Y)}$$

We have to note that $X\overline{X(Z \setminus Y)} = X(\overline{Z \setminus Y})$, and that $\overline{Z \setminus Y} = Y\overline{Z}$, we finally have that $X\overline{X(Z \setminus Y)} = XY\overline{Z}$. Therefore, we suppose that there is a set $A \subseteq \mathcal{U}, A \in \Sigma'$ such that:

(i) $A \supseteq X$.
(ii) $A \not\supseteq X(Z \setminus Y)$.
(iii) $A \not\supseteq XY\overline{Z}$.

On the other hand, we have that:
$X \Rightarrow Y \mid \overline{XY} \in \Sigma$ implies that $A \not\supseteq X$ or $A \supseteq XY$ or $A \supseteq X\overline{Y}$.
$Y \Rightarrow Z \mid \overline{YZ} \in \Sigma$ implies that $A \not\supseteq Y$ or $A \supseteq YZ$ or $A \supseteq Y\overline{Z}$.
Since we are assuming that $A \supseteq X$, we can discard $A \not\supseteq X$. We also have that the case $A \supseteq XY$ discards $A \not\supseteq Y$. This leaves three possibilities, either:

(i) $A \supseteq XY$ and $A \supseteq YZ$, that is, $A \supseteq XYZ \supseteq X(Z \setminus Y)$. This contradicts $A \not\supseteq X(Z \setminus Y)$.
(ii) $A \supseteq XY$ and $A \supseteq Y\overline{Z}$, that is, $A \supseteq XY\overline{Z}$. This contradicts $A \not\supseteq XY\overline{Z}$.
(iii) $A \supseteq X\overline{Y}$. $Y \cap Z = \emptyset$ implies that $Z \setminus Y \subseteq \overline{Y}$. All this yields $A \supseteq X\overline{Y} \supseteq X(Z \setminus Y)$. This contradicts $A \not\supseteq X(Z \setminus Y)$.

■

Therefore, we have proved that any Σ'' contains, at least, all the symmetric dependencies that are in Σ^+.

Corollary 1. *$\Sigma^+ \subseteq \Sigma''$.*

Proof. By Propositions 2 and 3. ■

We now prove completeness, that is, that Σ'' **only** contains all the dependencies in Σ^+.

Theorem 1. *$\Sigma'' \subseteq \Sigma^+$.*

Proof. We prove that $X \Rightarrow Y \mid Z \notin \Sigma^+$ implies that $X \Rightarrow Y \mid Z \notin \Sigma''$.

We have that $X \Rightarrow Y \mid Z \notin \Sigma^+$. It means that the dependency basis of X is such that in $DB_\Sigma(X) = [\underline{X} \mid P_1 \mid \cdots \mid P_n]$ (with $n \supseteq 1$) there is, at least, a class P_k such that $P_k \cap Y \neq \emptyset$ and $P_k \cap Z \neq \emptyset$. We fix P_k in this proof. We note that $|P_k| \supseteq 2$, since it contains, at least, one attribute from Y and one from Z.

Let $P = (\bigcup_{j=1}^{n} P_j) \setminus P_k$, that is, P is the union of all partitions in $DB_\Sigma(X)$ which are not X, except P_k. Therefore, $XP = \overline{P_k}$. We now claim that $XP \in \Sigma'$. We prove this statement by the way of contradiction. Assume that $XP \notin \Sigma'$. That is because there is a dependency $R \Rightarrow S \mid T \in \Sigma$ such that $X \supseteq R$ and $X \not\supseteq RS$ and $X \not\supseteq RT$. This implies that there is, at least, one attribute in RS which is not i XP, and, at least, one attribute in RT which is not in XP. Let them be $s \in RS, s \notin XP$ and $t \in RT, t \notin XP$. Since $X \supseteq R$, then, $s \in S, s \notin XP$ and $t \in T, t \notin XP$. Necessarily, since $s, t \notin XP$, then, $s, t \in P_k$.

Since $XP \supseteq R$, by reflexivity, $XP \Rightarrow R \mid \overline{XPR}$, and by transitivity, $XP \Rightarrow S \setminus R \mid \overline{XP(S \setminus R)}$. Without lack of generality, we assume that R, S, T are disjoint, so, finally, we have $XP \Rightarrow S \mid \overline{XPS}$. By the definition of $DB_\Sigma(X)$, then, $X \Rightarrow P \mid \overline{XP}$, and by transitivity, we have $X \Rightarrow S \setminus XP \mid \overline{X(S \setminus XP)}$. Since $s \in S, s \notin XP$, then, $s \in S \setminus XP$, and since $t \in T$ (assuming R, S, T disjoint), $t \notin S$, that, together with $t \notin XP$ yields that $t \in \overline{X(S \setminus XP)}$. It means that the attributes s, t are in different classes in $DB_\Sigma(X)$, but this contradicts the previous assumption that $P_k \in DB_\Sigma(X)$.

Now, we have that $XP \in \Sigma'$. Since $s, t \notin XP$, we have that $XP \not\supseteq XY$ and $XP \not\supseteq XZ$ and $XP \supseteq X$, which implies $X \Rightarrow Y \mid Z \notin \Sigma''$.

■

We have that Σ'' is exactly the set Σ^+. But, as we have already discussed in the previous section, in [3] and [17] we had a method to query Σ' whether a dependency was in Σ^+, and consisted in the closure operator $\Gamma_{\Sigma'}$ that, given a set of attributes, returned the meet of its up-set. In this present case, we may have that Σ' is not a lattice (but a partial lattice) and the same operator would not be a total function. Therefore, we use the up-set, instead of its meet:

Definition 4. *Let $\Sigma \subseteq SD_{\mathcal{U}}$. We define the up-set of $X \subseteq \mathcal{U}$ as follows:*

$$UP_{\Sigma}(X) = \{\, Y \in \Sigma' \mid Y \supseteq X \,\}$$

This definition is the standard one in lattice theory ([18]) when Σ' is an ordered set. The proof of the following proposition is trivial, yet, it will come handy to prove the last result of this paper.

Proposition 4. *Let $X, Y, Z \subseteq \mathcal{U}$ such that $Y \supseteq X$ and $Z \supseteq X$.*

$$UP_{\Sigma}(X) = UP_{\Sigma}(Y) \cup UP_{\Sigma}(Z)$$

if and only if

$$\nexists A \in \Sigma' : A \supseteq X \text{ and } A \not\supseteq Y \text{ and } A \not\supseteq Z$$

We need to remark that, although the set Σ' may not be closed under set intersection, the set of all up-sets of Σ' is closed under intersection. We are now ready to prove that it can be tested whether a dependency is in Σ^+ querying Σ' alone:

Proposition 5. $X \Rightarrow Y \mid Z \in \Sigma^+$ *if and only if*

$$UP_{\Sigma}(X) = UP_{\Sigma}(XY) \cup UP_{\Sigma}(XZ)$$

Proof.

$$X \Rightarrow Y \mid Z \in \Sigma^+$$

if and only if (by Corollary 1 and Theorem 1)

$$X \Rightarrow Y \mid Z \in \Sigma''$$

if and only if (by Proposition 1)

$$\nexists A : A \supseteq X \text{ and } A \not\supseteq XY \text{ and } A \not\supseteq XZ$$

if and only if (by Proposition 4)

$$UP_{\Sigma}(X) = UP_{\Sigma}(XY) \cup UP_{\Sigma}(XZ)$$

■

4 Example

We provide a running example in order to illustrate and clarify the results that are contained in the previous section. We depart from a set of attributes $\mathcal{U} = \{a, b, c, d\}$. The resulting formal context is presented in Figure 1.

As stated in Theorem 1, this contexts computes the set Σ^+. For instance, let us take the set

$$\Sigma = \{a \Rightarrow b \mid cd, b \Rightarrow ad \mid c\}$$

According to this context, we have that

$$\Sigma' = \{c, d, bc, cd, abc, abd, acd, bcd, abcd\}$$

and, finally,

$$\begin{aligned}\Sigma'' = \Sigma^+ = \{&a \Rightarrow b \mid cd, b \Rightarrow ad \mid c, a \Rightarrow c \mid bd, a \Rightarrow d \mid bc,\\ &ab \Rightarrow c \mid d, ac \Rightarrow b \mid d, ad \Rightarrow b \mid c, bd \Rightarrow a \mid c\}\end{aligned}$$

To check these results, we see that $ac \Rightarrow b \mid d$, $ad \Rightarrow b \mid c$ and $bd \Rightarrow a \mid c$ are derived from Σ by the reflexivity, transitivity and complementation. For instance, given $a \Rightarrow b \mid cd$, by reflexivity we have $ac \Rightarrow a \mid bd$, and by transitivity $ac \Rightarrow b \mid d$ (complementation comes from the notation $X \Rightarrow Y \mid Z$ used in this paper). Dependencies $ad \Rightarrow b \mid c$ and $bd \Rightarrow a \mid c$ can be derived alike.

As for the remaining SD's:

$$a \Rightarrow c \mid bd, a \Rightarrow d \mid bc$$

by applying transitivity to $a \Rightarrow b \mid cd$ and $b \Rightarrow ad \mid c$, we obtain $a \Rightarrow c \mid bd$, and with complementation we have $a \Rightarrow d \mid bc$.

	a	b	c	d	ab	ac	ad	bc	bd	cd	abc	abd	acd	bcd	$abcd$
$a \Rightarrow b \mid cd$		×	×	×	×			×	×	×	×	×	×	×	×
$b \Rightarrow a \mid cd$	×		×	×	×	×	×			×	×	×	×	×	×
$a \Rightarrow c \mid bd$		×	×	×		×		×	×	×	×	×	×	×	×
$c \Rightarrow a \mid bd$	×	×		×	×	×	×		×		×	×	×	×	×
$a \Rightarrow d \mid cb$		×	×	×			×	×	×	×	×	×	×	×	×
$d \Rightarrow a \mid cb$	×	×	×		×	×	×	×			×	×	×	×	×
$b \Rightarrow c \mid ad$	×		×	×		×	×	×		×	×	×	×	×	×
$c \Rightarrow b \mid ad$	×	×		×	×		×	×	×		×	×	×	×	×
$b \Rightarrow d \mid ac$	×		×	×		×	×		×	×	×	×	×	×	×
$d \Rightarrow b \mid ac$	×	×	×		×	×		×	×		×	×	×	×	×
$c \Rightarrow d \mid ab$	×	×		×	×		×		×	×	×	×	×	×	×
$d \Rightarrow c \mid ab$	×	×	×		×	×		×		×	×	×	×	×	×
$ab \Rightarrow c \mid d$	×	×	×	×		×	×	×	×	×	×	×	×	×	×
$ac \Rightarrow b \mid d$	×	×	×	×	×		×	×	×	×	×	×	×	×	×
$bc \Rightarrow a \mid d$	×	×	×	×	×	×	×		×	×	×	×	×	×	×
$ad \Rightarrow b \mid c$	×	×	×	×	×	×		×	×	×	×	×	×	×	×
$bd \Rightarrow a \mid c$	×	×	×	×	×	×	×	×		×	×	×	×	×	×
$cd \Rightarrow a \mid b$	×	×	×	×	×	×	×	×	×		×	×	×	×	×

Fig. 1. Formal context $(SD_{\mathcal{U}}, \wp(\mathcal{U}), I)$ for $\mathcal{U} = \{ a, b, c, d \}$

We now present an example with one more attribute, which may provide more insight in the details, but in this case, we do not present the context explicitly. The set of attributes is now $\mathcal{U} = \{ a, b, c, d, e \}$. Let Σ be the set of symmetric dependencies:

$b \Rightarrow a \mid cde$ $b \Rightarrow c \mid ade$ $c \Rightarrow a \mid bde$ $c \Rightarrow b \mid ade$ $d \Rightarrow a \mid bce$ $d \Rightarrow e \mid abc$
$e \Rightarrow a \mid bcd$ $e \Rightarrow d \mid abc$

According the formal context $(SD_{\mathcal{U}}, \wp(\mathcal{U}),\ \mathrm{I}\ )$, we have that:

$$\Sigma' = \{ a, abc, ade, abcd, abce, abde, acde, bcde, abcde \}$$

We can see that, applying the axioms of symmetric dependencies in Definition 1, the set Σ^+ is:

$b \Rightarrow a \mid cde$ $b \Rightarrow c \mid ade$ $c \Rightarrow a \mid bde$ $c \Rightarrow b \mid ade$ $d \Rightarrow a \mid bce$ $d \Rightarrow e \mid abc$
$e \Rightarrow a \mid bcd$ $e \Rightarrow d \mid abc$ $abd \Rightarrow c \mid e$ $acd \Rightarrow b \mid e$ $ce \Rightarrow a \mid bd$ $de \Rightarrow a \mid bc$
$bcd \Rightarrow a \mid e$ $abe \Rightarrow c \mid d$ $ace \Rightarrow b \mid d$ $bce \Rightarrow a \mid d$ $bde \Rightarrow a \mid c$ $cde \Rightarrow a \mid b$
$ab \Rightarrow c \mid de$ $ac \Rightarrow b \mid de$ $ad \Rightarrow bc \mid e$ $ae \Rightarrow bc \mid d$ $bc \Rightarrow a \mid de$ $bd \Rightarrow ac \mid e$
$bd \Rightarrow a \mid ce$ $bd \Rightarrow ae \mid c$ $be \Rightarrow ac \mid d$ $be \Rightarrow ad \mid c$ $be \Rightarrow a \mid cd$ $cd \Rightarrow a \mid be$
$cd \Rightarrow ae \mid b$ $cd \Rightarrow ab \mid e$ $ce \Rightarrow ab \mid d$ $ce \Rightarrow ad \mid b$

We only state the non-trivial dependencies as in Definition 2. We take, for instance, the dependencies:

$$bd \Rightarrow ac \mid e, bd \Rightarrow a \mid ce, bd \Rightarrow ae \mid c$$

They are derived from the dependencies $b \Rightarrow a \mid cde$ and $b \Rightarrow c \mid ade$. They are in Σ^+ because the sets that include bd are $abcd, abde, bcde, abcde$. This obviously means that all of them respect all the dependencies in Σ^+. We take, for instance, the set $abcd$ and see that it respects $bd \Rightarrow ae \mid c$ because $abcd \geq bcd$ (the left-hand side plus the second right-hand side) and that it also respects $bd \Rightarrow a \mid ce$ because $abcd \geq abd$ (the left-hand side plus the first right-hand side). We can see in this example the duality of the definition of the relation respect. This is one case of derivation by augmentation, which means that the dependencies that derive another dependency remove the sets that would prevent the derived dependency from appearing in Σ''. In this latter particular case, the sets that could be forbitting any of these dependencies from appearing in Σ'' have been cleared by $b \Rightarrow a \mid cde$ and $b \Rightarrow c \mid ade$. We take, for instance, the set bde, (which would prevent $bd \Rightarrow a \mid ce$ from being in Σ'') is not in Σ' because it does not respect the dependency $b \Rightarrow a \mid cde$.

We now illustrate one case of derivation by transitivity with the following set:

$$a \Rightarrow bc \mid de, bc \Rightarrow d \mid ae$$

By transitivity, we have that $a \Rightarrow d \mid bce \in \Sigma^+$. If we take $\Sigma = \{ a \Rightarrow bc \mid de \}$, we have:

$$\Sigma' = \{b, c, d, e, bc, bd, be, cd, ce, de, abc, ade, bcd, bce, bde, cde, abcd, abce, abde, acde, bcde, abcde\}$$

It is clear that $a \Rightarrow d \mid bce \notin \Sigma''$ since the sets $abc, acde \in \Sigma'$ do not respect this dependency. Now, if we include $bc \Rightarrow d \mid ae$ in Σ, we have:

$$\Sigma' = \{ b, c, d, e, bd, be, cd, ce, de, ade, bcd, bde, cde, abcd, abce, abde, acde, bcde, abcde \}$$

It has precisely been the dependency $bc \Rightarrow d \mid ae$ the one that has cleared both abc and $acde$ from Σ' and, therefore, allows $a \Rightarrow d \mid bce$ to appear in Σ''.

We now illustrate how Σ' alone can be used to query what dependencies hold in Σ^+. Again, we have that Σ is the set:

$$b \Rightarrow a \mid cde \quad b \Rightarrow c \mid ade \quad c \Rightarrow a \mid bde \quad c \Rightarrow b \mid ade \quad d \Rightarrow a \mid bce \quad d \Rightarrow e \mid abc$$
$$e \Rightarrow a \mid bcd \quad e \Rightarrow d \mid abc$$

and, therefore:

$$\Sigma' = \{ a, abc, ade, abcd, abce, abde, acde, bcde, abcde \}$$

We can see that Σ' in not closed ($abcd, abde \in \Sigma'$, but $ab \notin \Sigma'$). Now, suppose that we want to test whether a dependency is in Σ^+. For instance, we take a dependency that is not in Σ^+, as $a \Rightarrow bc \mid de$ and query Σ':

$$\begin{aligned} UP_\Sigma(a) &= \{\, a, abc, ade, abcd, abce, abde, acde, abcde \,\} \\ UP_\Sigma(abc) &= \{\, abc, abcd, abce, abcde \,\} \\ UP_\Sigma(ade) &= \{\, ade, abde, acde, abcde \,\} \end{aligned}$$

According to Proposition 5, since the sets $UP_\Sigma(a)$ and $UP_\Sigma(abc) \cup UP_\Sigma(ade)$ do not coincide, then, this dependency does not hold in Σ^+. We see that the set that does not allow this equality to hold is the set a, which is in Σ' because all dependencies in Σ are respected by this set. We take now a positive example of a dependency that is in Σ^+ but not in Σ, as for instance $ab \Rightarrow c \mid de$:

$$\begin{aligned} UP_\Sigma(ab) &= \{\, abc, abcd, abce, abde, abcde \,\} \\ UP_\Sigma(abc) &= \{\, abc, abcd, abce, abcde \,\} \\ UP_\Sigma(abde) &= \{\, abde, abcde \,\} \end{aligned}$$

In this case, the sets $UP_\Sigma(ab)$ and $UP_\Sigma(abc) \cup UP_\Sigma(abde)$ coincide.

5 Discussion

We have seen in Section 2 that the different characterizations of dependencies with formal contexts follow a common pattern, regardless of the type of dependencies or the definition of the context. Yet, the definition of formal contexts for AD's and SD's as in [3] was structurally different (powersets versus partitions) and that made it difficult to find a relationship and generalization between both contexts, in spite of the clear structural similarities that exist between AD's and SD's.

Now, we have that the relation I is a **generalization** of the relation defined in the context $\mathbb{K}_{AD}(\mathcal{U}) = (AD_\mathcal{U}, \wp(\mathcal{U}), I)$. We recall the definition of this relation ([17]):

Definition 5. *$A \subseteq \mathcal{U}$ **respects** an Armstrong dependency $X \to Y$ iff:*

$$A \not\supseteq X \text{ or } A \supseteq XY$$

We see that this definition avoids the reference to the second right-hand side, precisely because in AD's, complementation does not hold. If we drop this part from Definition 3, we have the definition of the respect relation for Armstrong dependencies.

This generalization seems to suggest that the solutions that have been developed based on the lattice characterization of sets of Armstrong dependencies, may also be applied to symmetric dependencies, namely:

1. To define a formal context for mixed sets of AD's and SD's.

2. To adapt the classical query algorithm for learning Armstrong dependencies.
3. To characterize the generating set of a set of symmetric dependencies.

Yet, although we are now in a better position to attack those problems, it does not seem to be a trivial task. For instance, the intuition would tell us that defining a formal context for mixed sets of dependencies, such that the relation would be the union of the relations already defined for AD's and SD's would work, but this is not the case. In fact, although this is out of the scope of this paper, this mixed formal contexts characterizes the symmetric dependencies that are in Σ^+, where Σ is a mixed set of AD's and SD's, but fails in characterizing the AD's that are in Σ^+. However, this simple strategy allows to advance towards the definition of a mixed formal context, which would have not been that simple departing from a partition context.

Adapting the classic learning algorithm for learning AD's and the characterization of the generating set of a set of SD's may encounter some difficulties. The main difference between Σ' for Armstrong and symmetric dependencies is that for the former, Σ' is always a powerset lattice closed under intersection, whereas for symmetric dependencies, this is not necessarily the case, and, therefore, not all the existing solutions for Armstrong dependencies, based on lattices, may be applied out of the box to symmetric dependencies. Yet, the fact that now we are dealing with contexts of the same nature, offers a much clearer perspective and understanding than before.

It must be said too that whereas this new characterization may make it potentially easier to find methods for finding minimal basis and query learning for SD's, it is true that SD's have not yet been used outside the database domain. We think that advancing in the study of lattice characterization for SD's and finding algorithmic similarities with FD's may introduce the use of SD's in other domains, namely knowledge discovery and machine learning, or in database theory, where it is already present: it would be of interest to have algorithms to compute minimal basis for SD's, profiting from the important collection of algorithms that compute the minimal basis of a set of AD's.

Finally, we would like to remark that the size of the formal context is greatly improved w.r.t the context in [3], since we have replaced the set $\mathrm{Part}(\mathcal{U})$ by the set $\wp(\mathcal{U})$. Yet, and for the sake of algorithmic solutions already existing in the FCA community, we have to say that the size of the context remains exponential.

6 Conclusions and Future Work

We have presented a new formal context for symmetric dependencies. This contexts provides the same functionalities as previous approaches, and it is much simpler. Yet, it offers the same expressivity power and, in fact, reduces the conceptual gap between Armstrong and symmetric dependencies that existed in a previous approach. We strongly believe that this may be the first step towards the resolution via formal concept analysis, of the learning, minimal bases and mixed sets of dependencies problems for symmetric dependencies, profiting from solutions already existing for Armstrong dependencies.

References

1. Angluin D., Frazier M., Pitt L. Learning Conjunctions of Horn Clauses. Machine Learning, 9:147-164, 1992.
2. Arias M., Balcázar, José L. Canonical Horn Representations and Query Learning. Lecture notes in computer science, vol. 5809, p. 156-17, 2009.
3. Baixeries, Jaume. A Formal Context for Symmetric Dependencies. ICFCA 2008. LNAI 4933.
4. Baixeries, Jaume and Balcázar, José L. Discrete Deterministic Data Mining as Knowledge Compilation. Proceedings of Workshop on Discrete Mathematics and Data Mining in SIAM International Conference on Data Mining, 2003.
5. Baixeries, Jaume and Balcázar, José L. Characterization and Armstrong Relations for Degenerate Multivalued Dependencies Using Formal Concept Analysis. Formal Concept Analysis, Third International Conference, ICFCA 2005, Lens, France, February 14-18, 2005, Proceedings. Lecture Notes in Computer Science, 2005
6. Baixeries, Jaume and Balcázar, José L. Unified Characterization of Symmetric Dependencies with Lattices. Contributions to ICFCA 2006. 4th International Conference on Formal Concept Analysis 2005.
7. Baixeries, Jaume. A Formal Concept Analysis framework to model functional dependencies. Mathematical Methods for Learning, 2004.
8. Balcázar, José L. and Baixeries, Jaume. Discrete Deterministic Data Mining as Knowledge Compilation. Workshop on Discrete Mathematics and Data Mining in SIAM Int. Conf. 2003.
9. Beeri, Catriel and Fagin, Roland and Howard, John H. A Complete Axiomatization for Functional and Multivalued Dependencies in Database Relations. Proceedings of the 1977 ACM SIGMOD International Conference on Management of Data, Toronto, Canada, August 3-5, 1977.
10. Caspard, Nathalie and Monjardet, Bernard. The Lattices of Closure Systems, Closure Operators, and Implicational Systems on a Finite Set: a Survey. Proceedings of the 1998 Conference on Ordinal and Symbolic Data Analysis (OSDA-98). Discrete Applied Mathematics, 2003.
11. Day, Alan. The Lattice Theory of Functional Dependencies and Normal Decompositions. International Journal of Algebra and Computation Vol. 2, No. 4 409-431. 1992.
12. Day, Alan. A Lattice Interpretation of Database Dependencies. Semantics of Programming Languages and Model Theory, 1993.
13. Demetrovics, János and Hencsey, Gusztav and Libkin, Leonid and Muchnik, Ilya. Normal Form Relation Schemes: a New Characterization. Acta Cybernetica, 1992.
14. Demetrovics, János and Huy, Xuan. Representation of Closure for Functional, Multivalued and Join Dependencies. Computers and Artificial Intelligence, 1992.
15. Demetrovics, János and Libkin, Leonid and Muchnik, Ilya. Functional Dependencies in Relational Databases: a Lattice Point of View. Discrete Applied Mathematics, 1992.
16. Duquenne, Vincent and Guigues, J.L. Familles Minimales d'Implications Informatives Resultant d'un Tableau de Donées Binaires. Mathematics and Social Sciences, 1986.
17. Ganter, Bernhard and Wille, Rudolf. Formal Concept Analysis: Mathematical Foundations. Springer, 1999.
18. Grätzer, George. General Lattice Theory. Academic Press, 1978.

19. Pfaltz, John L. Using Concept Lattices to Uncover Causal Dependencies in Software. Formal Concept Analysis, 4th International Conference, ICFCA 2006, Dresden, Germany, February 13-17, 2006.
20. Pfaltz, John L. Incremental Transformation of Lattices: A Key to Effective Knowledge Discovery. In *Proc. of the First Intl. Conf. on Graph Transformation (ICGT'02)*, pages 351–362, Barcelona, Spain, Oct 2002.
21. Ullman, Jeffrey D. Principles of Database Systems. Computer Science Press, 1982.

Cheating to achieve Formal Concept Analysis over a large formal context*

Victor Codocedo[1,3], Carla Taramasco[2], and Hernán Astudillo[1]

[1] Universidad Técnica Federico Santa María, Av. España 1640. Valparaíso, Chile.
[2] École Polytechnique, 32 Boulevard Victor 75015 Paris, France.
[3] LORIA, BP 70239, F-54506 Vandoeuvre-lès-Nancy, France.

Abstract. Researchers are facing one of the main problems of the *Information Era*. As more articles are made electronically available, it gets harder to follow trends in the different domains of research. Cheap, coherent and fast to construct knowledge models of research domains will be much required when information becomes unmanageable. While Formal Concept Analysis (FCA) has been widely used on several areas to construct knowledge artifacts for this purpose [17] (Ontology development, Information Retrieval, Software Refactoring, Knowledge Discovery), the large amount of documents and terminology used on research domains makes it not a very good option (because of the high computational cost and humanly-unprocessable output). In this article we propose a novel heuristic to create a taxonomy from a large term-document dataset using Latent Semantic Analysis and Formal Concept Analysis. We provide and discuss its implementation on a real dataset from the Software Architecture community obtained from the ISI Web of Knowledge (4400 documents).

1 Introduction

Research communities are facing one of the main problems of the Information Era and Formal Concept Analysis is not prepared to solve it. The amount of articles available online is growing each year yielding difficult to track trends, following ideas, looking for new terminology, etc. While some communities have understood the need for an artifact representing the knowledge within the domain (such as an ontology, a body-of-knowledge or a taxonomy) the problem remains in its construction since it is hard (highly technical), expensive (researchers are scarce) and complex (information is dynamic).

Automatic and semi-automatic creation of a terms taxonomy have been widely boarded in several fields [3,4,5,13,24]. In this work we focus on the approach described by Roth et al. [19] in which a taxonomy is derived from a corpus of documents by the use of Formal Concept Analysis (FCA). In particular, they describe an application used to *"represent a meaningful structure of*

* We would like to thank Chilean project FONDEF D08I1155 ContentCompass, intra-basal project FB/20SO/10 in the context of the Chilean basal project FB0821 and ECOS-CONICYT project C09E08 for funding this work.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

a given knowledge community in a form of a lattice-based taxonomy". This application is illustrated using a set of abstracts of the embryologist community obtained from MedLine spanning 5 years where a random set of 25 authors and 18 terms were analyzed. Although the lattice-based taxonomy obtained was a fair representation of the domain, real-size corpora of research communities are rather much larger than this example.

Handling large datasets has been defined as one of the open problems in the community of FCA[4] for two main reasons: first, the computational costs involved in the calculation of the concept lattice can make the use of FCA prohibitive and second, the concept lattice structure yielded could be so complex that its use may be impossible [10].

Iceberg lattices [21] help in improving readability by eliminating *"not representative"* data, but useful information, such as *"emerging behaviors* [12,15], is lost in the process. Stabilized lattices (using a stability measure [16]) also improves readability by eliminating *"noisy elements"* from data, but being a post-process tool it also raises computational costs.

We describe in this document a novel heuristic to create a lattice-based taxonomy from a large corpus using Formal Concept Analysis and a widely used Information Retrieval technique called Latent Semantic Analysis (LSA). In particular, we describe a process to compress a *formal context* into a smaller *reduced context* in order to obtain a lattice of terms that can be used to describe the knowledge on a given research domain. We illustrate our approach using a real-size dataset from a research community of Computer Sciences.

The remainder of this paper proceeds as follows: Section 2 explains the basis of FCA, section 3 presents our approach and section 4, a case study over a real dataset from a research community. Section 5 presents the results and a comparison of the obtained taxonomy with a human-expert handmade thesaurus. Finally, the conclusions are described in section 6.

2 Formal Concept Analysis

Formal Concept Analysis, originally developed as a subfield of applied mathematics [23], is a method for data analysis, knowledge representation and information management. It organizes information in a lattice of formal concepts. A formal concept is constituted by its *extension* (the objects that compose the concept) and its *intension* (the attributes that objects share). Objects and attributes are placed as rows and columns (resp.) in a cross-table or *formal context* where each cell indicates whether the object of that row have the attribute of that column. In what follows, we describe the Formal Concept Analysis framework as synthesized by Wille [22].

[4] `http://www.upriss.org.uk/fca/problems06.pdf`

2.1 Framework

Let G be a set of objects, M a set of attributes and I a binary relation between G and M ($I \subseteq (G \times M)$) indicating by gIm that the object g contains the attribute m and $\mathbb{K} = (G, M, I)$ be the *formal context* defined by G, M and I. For $A \subseteq G$ and $B \subseteq M$ it is defined the *derivation operator* ($'$) as follows:

$$A' = \{m \in M \mid gIm, \forall g \in A\}, \text{ with } A \subseteq G \tag{1}$$

$$B' = \{g \in G \mid gIm, \forall m \in B\}, \text{ with } B \subseteq M \tag{2}$$

A *formal concept* of the *formal context* $\mathbb{K}$ is defined by (A, B) with $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$, where A is called the *extent* and B is called the *intent* of the concept. The set of all formal concepts is defined as $L(G, M, I)$.

For two formal concepts $(A_1, B_1), (A_2, B_2) \in \mathbb{K}$, the *hierarchy* of concepts is given by the relation subconcept-superconcept as follows:

$$(A_1, B_1) \leq (A_2, B_2) \iff A_1 \subseteq A_2 (\iff B_1 \supseteq B_2) \tag{3}$$

Where $(A_1, B1)$ is called the *subconcept* and (A_2, B_2) is called the *superconcept*.

$\mathfrak{B}(\mathbb{K}) = (L(G, M, I), \leq)$ is the complete lattice or *concept lattice* of context $\mathbb{K}$

2.2 Iceberg Concept Lattices

Let (A, B) be a concept of $\mathfrak{B}(\mathbb{K})$, its *support* is defined as:

$$supp(A, B) = \frac{|A|}{|G|} \tag{4}$$

Given a threshold **minsupp** $\in [0, 1]$, the concept (A, B) is called a *"frequent concept"* if $supp(A, B) \geq$**minsupp**.

An **Iceberg lattice** [21] is the set of all frequent concepts for a given **minsupp**.

2.3 Stability

Stability was proposed by Kuznetsov in [14,16] as a mechanism to prune *"noisy concepts"*. It was extended by Roth et al. We use and provide their definition from [19] and [15]:

Let $\mathbb{K} = (G, M, I)$ be a formal context and (A, B) be a formal concept of $\mathbb{K}$. The stability index, σ, of (A, B) is defined as follows:

$$\sigma(A, B) = \frac{|\{C \subseteq A \mid C' = B\}|}{2^{|A|}} \tag{5}$$

Stability measures how much the intent of a concept depends on particular objects of its extent, meaning that if the formal context changes and some objects disappear, then stability indicates how likely it is for a concept to remain in the concept lattice. Stability can also be used to construct a *stabilized lattice* for a given threshold similarly to an *iceberg lattice*.

Analogous to definition 5, the **extensional stability** of a concept (A, B) can be defined as:

$$\sigma_e(A, B) = \frac{|\{D \subseteq B \mid D' = A\}|}{2^{|B|}} \tag{6}$$

Extensional stability measures how likely is for a concept to remain if some attributes are eliminated from the context. We will use both definitions in this work differentiating them as *intensional stability* (on (5)) and *extensional stability* (on (6)).

3 Reducing a large formal context

Different from Roth's approach [19], we are not interested in tracking groups of people working on groups of topics, but rather in the relations among topics. These relations occur in the articles that authors write, where topics or terms can appear in sets and each one can appear one or more times. To elaborate:

Given a corpus of articles G, a list of terms M and the relation among them $I \subseteq (G \times M)$ indicating by gIm that the article g contains the term m, the document-article formal context is defined as:

$$\mathbb{K}_O = (G, M, I) \tag{7}$$

3.1 Rationale

Even for a small set of terms, the amount of articles for a small research community can reach thousands of articles making the processing of $\mathbb{K}_O$ impossible or useless. The problem gets worse over time, because it can be expected that each year hundreds of articles will be added to the corpus.

What happens with terms over time? In taxonomy evolution, as described in [18], symmetric patterns arise: some fields will *progress or decline*; some fields will contain more or less concepts (*enrichment or impoverishment*); and some fields will *merge or split*. In any case, it is not expected that the amount of terms would vary greatly.

Latent Semantic Analysis (LSA) or Latent Semantic Indexing (LSI) [6] is a technique used commonly in Information Retrieval (IR) as a tool for indexation, clusterization and query answering. LSA is based on the idea that for a given set of terms and documents, *the relation among terms can be explained by a set of dimensions whose size is much smaller than the amount of documents*. We exploit this feature of LSA to construct a **reduced formal context of dimensions and terms** having as conditions that information regarding relations of

terms cannot be lost and that it has to produce a coherent taxonomy using less computational time. In what follows, we provide a brief description of LSA to elaborate on how we used it to produce a *reduced formal context*. For further reading, please refer to [6].

3.2 Latent Semantic Analysis

Given a list of m terms and a corpus of n documents, let A be a term-document matrix of rank-min(m,n) as defined in 8, where a_{ij} is the weight[5] of the term i in the document j. The Single-Value Decomposition of matrix A (in equation (9)) produces its factorization in three matrices where Σ contains the single-values of matrix A at the diagonal in descending order and the columns of matrices U and V are called left and right singular vectors of A.

$$A_{m\times n} = [a_{ij}] \; ; \; i = [1..m], j = [1..n] \tag{8}$$

$$A_{m\times n} = U_{m\times m} \cdot \Sigma_{m\times n} \cdot V^T_{n\times n} \tag{9}$$

$$A'_{m\times n} = U_{m\times k} \cdot \Sigma_{k\times k} \cdot V^T_{k\times n} \tag{10}$$

Since singular values drops quickly, we can create a new approximation of matrix A using $k \ll min(m,n)$ as shown in (10). Matrix $A' \approx A$ is the closest k-rank matrix approximation to A by the Frobenius measure [11]. Two new matrices can be calculated:

$$B_{m\times k} = U_{m\times k} \cdot \Sigma_{k\times k} \tag{11}$$

$$C_{n\times k} = V_{n\times k} \cdot \Sigma_{k\times k} \tag{12}$$

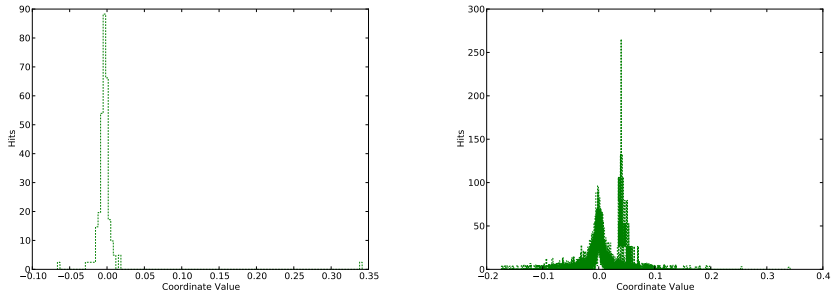
where B holds the vector-space representation in k dimensions of terms; and C the one of documents. Both of these matrices are used for clusterization since, on them, similar elements are closer on each dimension. In particular, each dimension on B (each column) has a Gaussian-like distribution where terms group around the mean value (see figure 1), except for dimension 0 (the different behavior in figure 1(b)) where terms have almost the same value[6]. We exploit this feature to define a *conversion-function* that allows us to construct the *reduced context*.

3.3 A probabilistic-based *conversion-function*

Which terms are related within a given dimension? Since each dimension holds continuous values, it is hard to define a region for them. Nevertheless, we know

[5] Several weighting functions can be used, being the most used frequency of term and term frequency-inverse document frequency (tf.idf)

[6] We do not use the information in this dimension for our analysis and exclude it from our results.



(a) Distribution of values in dimension 1 (b) Distribution of values in all dimensions

Fig. 1. Distribution of Dimensions' values in matrix B

that such a region has to be centered at the mean value of the dimension. Hence, we define a *"belonging region"* centered at the mean with a modifiable width. Terms in this region are related because they *belong* in the dimension and hence, the pair dimension-term will appear in the *reduced context*. The width of the *"belonging region"* is a parameter that allows us to manage the density of the context. The conversion function is defined as:

$$b_l(x,k) = \begin{cases} 1 & G_k(x) \in [\alpha, 1-\alpha] \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

where function G_k is the probability density function (PDF) for dimension k and $\alpha \in]0{,}0.5[$ defines the limits of the *"belonging region"*.

3.4 Creating the reduced context

For a document-article formal context $\mathbb{K}_O$ as defined in (7) (*original context*) and a term-document matrix A as defined in (8) analogous to $\mathbb{K}_O$:

Given the factorization of matrix A as defined in (10), the vector-space representation of its terms in k dimensions B as defined in (11) and a conversion-function $b_l(x,k)$ as defined in (13):

Let D be the set of k dimensions in B

$$I_R \subseteq (D \times M) = \{(j,i) : \forall j \in D \wedge \forall i \in M \iff b_l(B_{ij}, j) = 1\} \tag{14}$$

we define the **reduced context** of $\mathbb{K}_O$ as $\mathbb{K}_R = (D, M, I_R)$.

Notice the inversion of pair (j,i) and B_{ij} performed to respect LSA conventions that require term-document matrices and FCA that uses document as objects and terms as attributes. In the *reduced context* we say *"dimension j contains term i if the evaluation of the conversion-function b_l over the value of the coordinate j for the term i is 1"*.

Summarizing, in order to get a *reduced context*, the values for α and k must be found.

3.5 Related approaches

Similar techniques have been proposed before. Gajdos et al[9] used LSA to reduce complexity in the structure of the lattice by eliminating noise in the formal context. While this approach is useful, it does not reduce the amount of data, but it *"tunes it"* to get a clearer result. Snasel et al. [20,9] proposed a matrix-reduction algorithms based on NMF. [7] and SVD[8]. While they state that these methods are successful to reduce the amount of concepts obtained using FCA, they do not describe a real life use of their technique (their experiment was performed over a 17x16 matrix) neither do they discuss about the performance of their approach. Kumar and Srinivas [1] approach consists of using fuzzy K-Means clustering [9] to reduce the attributes in a formal term-document context. In their approach, documents are categorized in k clusters using the *cosine similarity measure*. Cheung et al. [2] introduced *term-document* lattices complexity reduction by defining a set of equivalence relations that allows to reduce the set of objects. Finally, Dias et al. introduced JBOS [7] (junction based on objects similarity) which proposed a similar method where objects where group into prototype objects by calculating its similarity according to certain weights assigned manually to attributes.

4 Case Study: Software Architecture Community

The Software Architecture Corpus (SAC) was composed by extracting metadata from papers retrieved by the ISI Web of Knowledge search engine [10] using the query "software architecture". It is assumed that the keyword "software architecture" is present in each paper on their titles and/or abstracts.

While the search engine retrieved 4701 articles, not all of them have an abstract to work with. Those are excluded from our analysis leaving 4565 articles spanning from 1990 to 2009 (retrieved documents span from 1973 to 2009).

4.1 Term list

A term list was assembled by using Natural Language Processing over the articles' titles and abstracts. In order to avoid common words, a *stopword* list and a lexical tagger were used as a filter. A list of candidate terms was then manually filtered to obtain a final list of 120 terms, which included words and multi-words (such as *"Unified Model Language"*). Table 1 shows a sample of selected terms.

Each term was looked up on each document and its frequency of use was calculated. Then, a weighting measure was applied (tf.idf[11]) to each value. The

[7] Non-negative matrix factorization

[8] Single-Value Decomposition

[9] K-Means Clustering is a classic clustering technique for vector-space models

[10] http://isiwebofknowledge.com

[11] Term Frequency-Inverse Document Frequency is a weighting measure commonly used on IR based on the notion that term infrequency on a global scale makes it important.

Table 1. Top 10 more frequent terms

Term	Frequency
design	1710
development	1450
component	1253
process	1083
implementation	1006
datum	874
requirement	869
analysis	851
framework	817
control	801

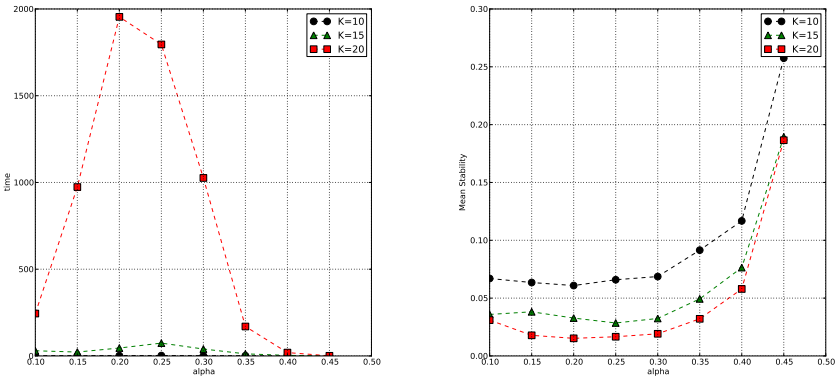
term-document matrix $A_w = a_{ij}$ was constructed using the final list of terms (M) and the corpus of documents (G) where a_{ij} represents the weight of term i in document j. We defined the relation $I \subseteq (G \times M) = \{(j,i) : \forall j \in G \wedge \forall i \in M \iff a_{ij} > 0\}$ to build up the **original context** $\mathbb{K}_O = (G, M, I)$ describing that a document *contains* a term only if its weight on it is over 0. The formal context $\mathbb{K}_O$ was used later to compare our reductions.

4.2 Reducing the SAC

As we stated at the end of section 3.3, two parameters had to be set in order to create the *reduced context*. Sadly, in LSA there is not a known method to find the best value for k, and not knowing that, it is not possible to find a good value for α. We defined a set of goals to observe which were the values of k and α that best accomplished them. The goals defined were:

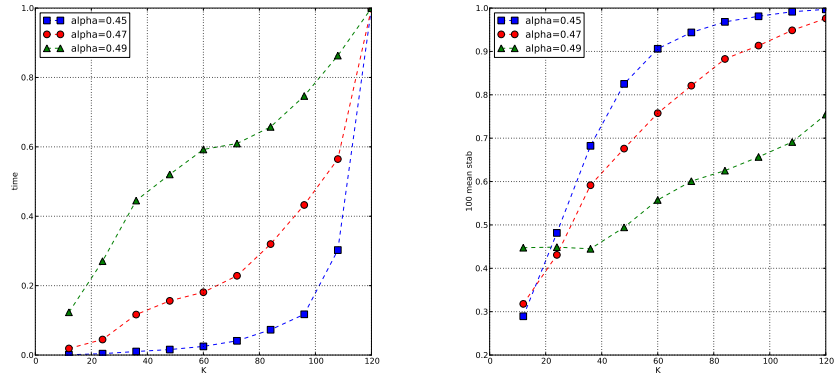
- Low Execution time
- High Stability
- Few Concepts in the final lattice

Using three fixed values for k we reduced several contexts and processed them through FCA in order to find the best value for α. As shown in figure 2, it was found that higher values of α (close to 0.5) yields the best results. Repeating the experience with 3 fixed values for α (0.45, 0.47 and 0.49) to find the best value for k we found a trade-off between stability and execution time as it can be observed in figure 3. Higher values of k yield higher stability but also a high execution time, and vice-versa. Since stability drops fast on k=60 and in the same value the execution time grows greatly, we selected it to obtain our results. α was set on 0.45 and 0.47.



(a) alpha vs Execution Time (b) alpha vs Stability

Fig. 2. Fixed K, Variable α



(a) K vs Execution Time Normalized (b) K vs Top 100 Mean Stability

Fig. 3. Variable K, Fixed α

5 Results and Discussion

Table 2 shows a comparative of the characteristics of the lattices yielded from two *reduced contexts* ($\mathbb{K}_R$) and the *original context* ($\mathbb{K}_O$). The lattices were processed using the FCA suite Coron System[12].

[12] `http://coron.loria.fr/`

Table 2. Comparison of characteristics

	$\alpha = 0,45$	$\alpha = 0,47$	Original
Objects	60	60	4565
Attributes	120	120	120
Density [%]	17,24	10,59	6,59
Concepts	6309	1207	170606
Coincidental Intents	3029	815	-
Mean attributes per concept	20,52	12,6	7,91
Intensional Stability	0,2170	0,3041	0.3995
Extensional Stability	0.2277	0.3211	0.1103
100 Top Int. Stab.	0,9061	0,7576	1
100 Top Ext. Stab.	0.9515	0.8287	0.9837
Levels	10	7	10
Time [s]	6,869	1,145	2865,723
Time to reduce [s]	39,333	39,325	-

Results shows that using LSA before FCA performs a **clear reduction in the formal context** from a size of 4565×120 (original context) to 60×120 (reduced context), specifically a reduction of 76 times the amount of data to be processed. It also **lowers the amount of concepts** yielded in the final lattice (27 and 141 times for α equal to 0.45 and 0.47 resp.), and because of that the **time required to calculate the full concept lattice is considerably reduced**, even considering the time required to create the *reduced contexts*.

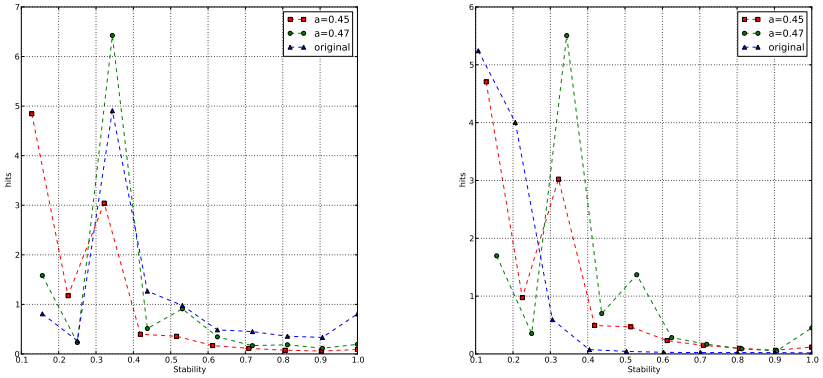
Stability gives more clues about the good quality of the reduction. Figure 4 shows intensional and extensional stability distribution. As it can be observed, the original context's lattice has a better intensional stability than the reduced contexts but a worst extensional stability. Mean values for these two measures are shown in table 2.

Since we have eliminated redundant data, each dimension is almost equally important meaning that in *reduced contexts* we cannot afford to eliminate a subset of them without affecting greatly the structure of the lattice obtained. In this case, we have eliminated a big part of the noise (k=60 was in fact a very good choice). On the other hand, the growth in extensional stability reflects that the structure of the reduced lattices is not tied to some specific terms. Some terms can be removed and the structure of the lattice would not vary greatly, which is what happens each year (see section 3.1).

5.1 A Software Architecture Taxonomy

Figure 5 shows the reduced notation of the lattice for the *reduced context* (k=60 and $\alpha = 0.45$). This lattice was drawn with Coron-drawer[13] a set of scripts specially written for large lattices. For the sake of space and simplicity we provide

[13] `http://code.google.com/p/coron-drawer/`



(a) Intensional Stability (b) Extensional Stability

Fig. 4. Stability distribution (k=60)

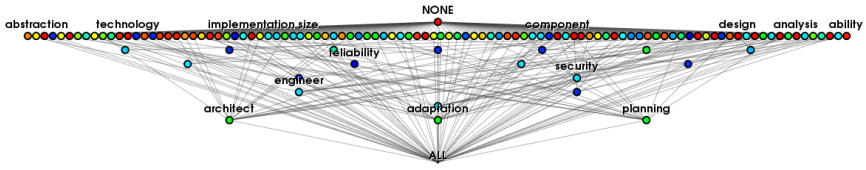


Fig. 5. Filtered Lattice (K=60, $\alpha = 0.45$, minsupp=0)

a small comparison of the terms in the reduced lattice-based taxonomy with a human-expert handmade thesaurus of Software Architecture [8].

Software Architecture Thesaurus Comparison The thesaurus contains 494 elements (we call them elements to differentiate them from lattice's concepts and taxonomy's terms) organized in a hierarchical fashion. They have at most one parent and the hierarchy has multiple roots. The thesaurus is exhaustive and comprises mainly definitions of Software Architecture's concepts and entities (such as framework's names or important authors in the domain). The comparison shows:

– From our 120 term list, 50 terms (42%) match exactly with a term on the thesaurus. 25 terms (21%) have a semi-match, meaning that they are part of a term on the thesaurus (*database* in our hierarchy and *shared database* in the thesaurus) and 45 (37%) terms do not have a simile in the thesaurus.

- The three main concepts *design, analysis and framework* (with support over 50%) found in our taxonomy, also remain being main elements in the thesaurus.
- Even when some elements in the thesaurus are not found in our taxonomy, they actually exists as relations among terms. For instance, the relation among the terms *design* and *pattern* describe the thesaurus' element *design pattern*. This is also true for *design decision*, *information view*, *knowledge reuse*, *quality requirements*, *business methodology* and several more elements.

6 Conclusions

In this work we have presented a method and a technique to apply Formal Concept Analysis (FCA) to large contexts of data in order to obtain a lattice-based taxonomy. We have outlined that large-size datasets are not suitable to be processed by FCA and that, this fact is an important problem in the domain.

The solution presented here, is based on an Information Retrieval technique called Latent Semantic Analysis which is used to reduce a term-document matrix to a much smaller matrix where terms are related to a set of dimensions instead of documents. Using a probabilistic approach, this matrix is converted into a binary formal context where FCA can be applied.

The approach was illustrated with a case study using a research domain from computational sciences called *Software Architecture*. The corpus created for this domain consists of more than 4500 documents and 120 terms. We have compared the characteristics of the lattice obtained through FCA from the original formal context of terms and documents and the reduced contexts generated by our approach. We have found that not only our approach is considerably more economic in execution time as well as in the amount of concepts obtained in the final lattice but intensional and extensional stabilities give us elements to be certain of the quality of our approach.

A small comparison with a human expert handmade thesaurus of the community of Software Architecture is provided in order to illustrate that a real and coherent taxonomy can be obtained using our approach.

References

1. Ch. Aswani Kumar and S. Srinivas. Concept lattice reduction using fuzzy K-Means clustering. *Expert Systems with Applications*, 37(3):2696–2704, March 2010.
2. Karen S. K. Cheung and Douglas Vogel. Complexity Reduction in Lattice-Based Information Retrieval. *Information Retrieval*, 8(2):285–299, April 2005.
3. Philipp Cimiano, Andreas Hotho, and Steffen Staab. Learning concept hierarchies from text corpora using formal concept analysis. *J. Artif. Int. Res.*, 24:305–339, August 2005.
4. Víctor Codocedo and Hernán Astudillo. No mining, no meaning: relating documents across repositories with ontology-driven information extraction. In *Proceeding of the eighth ACM symposium on Document engineering*, DocEng '08, pages 110–118, New York, NY, USA, 2008. ACM.

5. Wisam Dakka, Panagiotis G. Ipeirotis, and Kenneth R. Wood. Automatic construction of multifaceted browsing interfaces. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM '05, pages 768–775, New York, NY, USA, 2005. ACM.
6. Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the american society for Information Science*, 41(6):391–407, 1990.
7. Sergio M. Dias and Newton J. Vieira. Reducing the size of concept lattices: The JBOS Approach. In *Proceedings of the 8ht international conference on Concept Lattices and their Applications*, CLA 2010, pages 80–91, 2010.
8. Anabel Fraga and Juan Lloréns. Training initiative for new software/enterprise architects: An ontological approach. In *WICSA*, page 19. IEEE Computer Society, 2007.
9. Petr Gajdos, Pavel Moravec, and Václav Snásel. Concept lattice generation by singular value decomposition. In Václav Snásel and Radim Belohlávek, editors, *International Workshop on Concept Lattices and their Applications (CLA)*, volume 110 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2004.
10. Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin/Heidelberg, 1999.
11. Gene H. Golub and Charles F. Van Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.
12. Nicolas Jay, François Kohler, and Amedeo Napoli. Analysis of social communities with iceberg and stability-based concept lattices. In *Proceedings of the 6th international conference on Formal concept analysis*, ICFCA'08, pages 258–272, Berlin, Heidelberg, 2008. Springer-Verlag.
13. John Kominek and Rick Kazman. Accessing multimedia through concept clustering. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '97, pages 19–26, New York, NY, USA, 1997. ACM.
14. Sergei Kuznetsov. Stability as an estimate of the degree of substantiation of hypotheses derived on the basis of operational similarity. nauchn. tekh. inf., ser.2 (automat. document. math. linguist.). 12:21–29, 1990.
15. Sergei Kuznetsov, Sergei Obiedkov, and Camille Roth. Reducing the representation complexity of lattice-based taxonomies. In Uta Priss, Simon Polovina, and Richard Hill, editors, *Conceptual Structures: Knowledge Architectures for Smart Applications*, volume 4604 of *Lecture Notes in Computer Science*, pages 241–254. Springer Berlin / Heidelberg, 2007.
16. Sergei O. Kuznetsov. On stability of a formal concept. *Annals of Mathematics and Artificial Intelligence*, 49:101–115, April 2007.
17. Uta Priss. Formal concept analysis in information science. *Annual Review of Information Science and Technology*, 40(1):521–543, September 2007.
18. Camille Roth and Paul Bourgine. Lattice-based dynamic and overlapping taxonomies: The case of epistemic communities. *SCIENTOMETRICS*, 69(2):429–447, NOV 2006.
19. Camille Roth, Sergei Obiedkov, and Derrick Kourie. Towards concise representation for taxonomies of epistemic communities. In *Proceedings of the 4th international conference on Concept lattices and their applications*, CLA'06, pages 240–255, Berlin, Heidelberg, 2008. Springer-Verlag.
20. Vaclav Snasel, Martin Polovincak, and Hussam M. Dahwa. Concept lattice Reduction by Singular Value Decomposition. *Proceedings of the Spring Young Researcher's Colloquium on Database and Information Systems*, 2007.

21. Gerd Stumme. Efficient data mining based on formal concept analysis. In Abdelkader Hameurlain, Rosine Cicchetti, and Roland Traunmüller, editors, *Database and Expert Systems Applications*, volume 2453 of *Lecture Notes in Computer Science*, pages 3–22. Springer Berlin / Heidelberg, 2002.
22. Rudolf Wille. Restructuring lattice theory: an approach based on hierarchies of concepts. In Ivan Rival, editor, *Ordered sets*, pages 445–470, Dordrecht–Boston, 1982. Reidel.
23. Rudolf Wille. Formal concept analysis as mathematical theory of concepts and concept hierarchies. In Bernhard Ganter, Gerd Stumme, and Rudolf Wille, editors, *Formal Concept Analysis*, volume 3626 of *Lecture Notes in Computer Science*, pages 1–33. Springer Berlin / Heidelberg, 2005.
24. Jian-hua Yeh and Naomi Yang. Ontology construction based on latent topic extraction in a digital library. In George Buchanan, Masood Masoodian, and Sally Cunningham, editors, *Digital Libraries: Universal and Ubiquitous Access to Information*, volume 5362 of *Lecture Notes in Computer Science*, pages 93–103. Springer Berlin / Heidelberg, 2008.

A FCA-based analysis of sequential care trajectories

Elias EGHO, Nicolas Jay, Chedy Raissi and Amedeo Napoli

Orpailleur Team, LORIA, Vandoeuvre-les-Nancy, France
elias.egho,nicolas.jay,chedy.raissi,amedeo.napoli@loria.fr

Abstract. This paper presents a research work in the domains of sequential pattern mining and formal concept analysis. Using a combined method, we show how concept lattices and interestingness measures such as stability can improve the task of discovering knowledge in symbolic sequential data. We give example of a real medical application to illustrate how this approach can be useful to discover patterns of trajectories of care in a french medico-economical database.

Keywords: Data-Mining, Formal Concept Analysis, Sequential patterns, stability

1 Introduction

Sequential pattern mining, introduced by Agrawal et al [2], is a popular approach to discover patterns in ordered data. It can be seen as an extension of the well known association rule problem, applied to data that can be modelled as sequences of itemsets, indexed for example by dates. It helps to discover rules such as: customers frequently first buy DVDs of episodes I, II and III of Stars Wars, then buy within 6 months episodes IV, V, VI of the same famous epic space opera. Sequential pattern mining has been successfully used so far in various domains : DNA sequencing, customer behavior, web mining ... [2].

Many scalable methods and algorithms have been published so far to efficiently mine sequential patterns. However few of them deal with the multidimensional aspect of databases. Multidimensionality conveys two notions:

- items can be of different intrinsic nature. While the common approach considers objects of the same dimension, for example articles bought by customers, databases can hold much more information such as article price, gender of the customer, location of the store and so on.
- a dimension can be considered at different levels of granularity. For example, apples in a basket market analysis can be either described as fruits, fresh food or food following a hierarchical taxonomy.

Plantevit et al. [13] address this problem as mining multidimensional and multilevel sequential patterns and propose a method to achieve this task. They rely on the support measure to efficiently discover relevant sequential patterns. Support

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

indicates to what extent a pattern is frequent in a database. Many (sequential and non sequential) itemset mining methods use support as measure for finding interesting correlations in databases. However, the most relevant patterns may not be the most frequent ones. Moreover, discovering interesting patterns with low support leads generally to overwhelming results that need to be further processed in order to be analyzed by human experts.

Formal Concept Analysis (FCA) is a theory of data analysis introduced in [17], that is tightly connected with data-mining and especially the search of frequent itemsets [16]. FCA organizes information into a concept lattice representing inherent structures existing in data. Recently, some authors proposed new interest measures to reduce complex concept lattices and thus find interesting patterns. In [9], Kuznetsov introduces stability, successfully used in social network and social community analysis [7, 6].

To our knowledge, there are no similar approaches to find interesting sequential patterns. In this paper, we present an original experiment based on both multilevel and multidimensional sequential patterns and lattice-based classification. This experiment may be regarded from two points of view: on the one hand, it is based on multilevel and multidimensional sequential patterns search, and on the other hand, visualization and classification of extracted sequences is based on Formal Concept Analysis (FCA) techniques, organizing them into a lattice for analysis and interpretation. It has been motivated by the problem of mining care trajectories in a regional healthcare system, using data from the PMSI, the so called French hospital information system. The remaining of the paper is organized as follows. In Section 2, we present the problem of mining care trajectories. Section 3 presents the methods proposed in domains of multilevel and multidimensional sequential patterns and Formal Concept Analysis. In Section 4, we present some of the results we achieved.

2 Mining healthcare trajectories

The PMSI (Programme de Médicalisation des Systèmes d' information) database is a national information system used in France to describe hospital activity with both an economical and medical point of view. The PMSI is based on the systematic collection of administrative and medical data. In this system, every hospitalization leads to the collection of administrative, demographical and medical data. This information is mainly used for billing and planning purposes. Its structure can be described (and voluntarily simplified) as follows:

- Entities (attributes):
  - Patients (id, gender ...)
  - Stays (id, hospital, principal diagnosis, ...)
  - Associated Diagnoses (id)
  - Procedures (id, date,...)
- Relationships
  - a patient has 1 or more stays
  - a stay may have several procedures

- a stay may have several associated diagnoses

The collection of data is done with a minimum recordset using controlled vocabularies and classifications. For example, all diagnoses are coded with the International Classification of Diseases (ICD10)[1]. Theses classifications can be used as taxonomies to feed the process of multilevel sequential pattern mining as shown in figure 1.

ICD 10 Institutions taxonomy

Fig. 1. Examples of taxonomies used in multilevel sequential pattern mining

Healthcare management and planning play a key role for improving the overall health level of the population. From a population point of view, even the best and state-of-the-art therapy is not effective if it cannot be delivered in the right conditions. Actually, many determinants affect the effective delivery of healthcare services: availability of trained personnel, availability of equipment, security constraints, costs, proximity .... All of these should meet economics, demographics, and epidemiological needs in a given area. This issue is especially acute in the field of cancer care where many institutions and professionals must cooperate to deliver high level, long term, and costly care. Therefore, it is crucial for healthcare managers and decision makers to be assisted by decision support systems that give strategic insights about the intrinsic behavior of the healthcare system.

On the one hand, healthcare systems can be considered as rich in data as they produce massive amounts of data such as electronic medical records, clinical trial data, hospital records, administrative data, and so on. On the other hand, they can be regarded as poor in knowledge as these data are rarely embedded into a strategic decision-support resource [1]. We used the PMSI system as a source of data to study patient movements between several institutions. By organizing themselves into groups of sequences representing trajectories of care, we aim at discovering patterns describing the whole course of treatments for a given population. This global approach contrasts with the usual statistical exploitations of the PMSI data that focus mainly on single hospitalizations.

In this experiment, we have worked on four years (2006 – 2009) of the PMSI data of the Burgundy region related to patient suffering from lung cancer.

[1] http://apps.who.int/classifications/apps/icd/icd10online/

3 Related work

3.1 Sequential Pattern Mining

Let $\mathtt{I}$ be a finite set of items. A subset of $\mathtt{I}$ is called an itemset. A sequence $\mathtt{s} = \langle \mathtt{s}_1\mathtt{s}_2 \dots \mathtt{s}_k \rangle$ ($\mathtt{s}_i \subseteq \mathtt{I}$) is an ordered list of itemsets. A sequence $\mathtt{s} = \langle \mathtt{s}_1\mathtt{s}_2 \dots \mathtt{s}_n \rangle$ is a subsequence of a sequence $\mathtt{s}' = \langle \mathtt{s}'_1\mathtt{s}'_2 \dots \mathtt{s}'_m \rangle$ if and only if $\exists \mathtt{i}_1, \mathtt{i}_2, \dots \mathtt{i}_n$ such that $\mathtt{i}_1 \leq \mathtt{i}_2 \leq \dots \leq \mathtt{i}_n$ and $\mathtt{s}_1 \subseteq \mathtt{s}'_{\mathtt{i}_1}, \mathtt{s}_2 \subseteq \mathtt{s}'_{\mathtt{i}_2} \dots \mathtt{a}_n \subseteq \mathtt{s}'_{\mathtt{i}_n}$. We note $\mathtt{s} \subseteq \mathtt{s}'$ and also say that $\mathtt{s}'$ contains $\mathtt{s}$. Let $\mathtt{D} = \{\mathtt{s}_1, \mathtt{s}_2 \dots \mathtt{s}_n\}$ be a database of sequences. The support of a sequence $\mathtt{s}$ in $\mathtt{D}$ is the proportion of sequences of $\mathtt{D}$ containing $\mathtt{s}$. Given a $\mathtt{minsup}$ threshold, the problem of frequent sequential pattern mining consists in finding the set $\mathtt{FS}$ of sequences whose support is not less than $\mathtt{minsup}$. Following the seminal work of Agrawal and Srikant [2] and the Apriori algorithm, many studies have contributed to the efficient mining of sequential pattern. The main approaches are PrefixSpan [11], SPADE [20], SPAM [3], PSP [10], DISC [4] and PAID [18].

Much work has been done in the area of single-dimensional sequential patterns, i.e, all the items in a sequence have the same nature like the sequence of products sold in a certain store. But in many cases, the information in a sequence can be based on several dimensions. For example: a male patient had a surgical operation in Hospital A and then received chemotherapy in Hospital B. In this case, we have 3 dimensions: gender, type of treatment (chemotherapy, surgery) and location (Hospitals A and B). Pinto et al [12] is the first work giving solutions for mining multidimensional sequential patterns. They propose to include some dimensions in the first or the last itemset in the sequence. But this works only for dimensions that remain constant over time, such as gender in our previous example. Among other proposals addressed in this area, Yu et al [19] consider multidimensional sequential pattern mining in the web domain. In their approach, dimensions are pages, sessions and days. They present two algorithms AprioriMD and PrefixMDSpan by modifying the Apriori and PrefixSpan algorithms. Zhang et al [21] propose the mining of multidimensional sequential patterns in distributed system.

Moreover, each dimension can be represented by different levels of granularity, using a taxonomy which defines the hierarchical relations between items. Figure 2 shows an example of a diseases taxonomy. Including knowledge contained in the taxonomy leads to the problem of multilevel sequential pattern mining. Its interest resides in the capacity to extract more or less general/specific sequential patterns and overcome problems of excessive granularity and low support. For example, using the diseases taxonomy in Figure 2, sequences such as $\langle \mathtt{HeartDisease}, \mathtt{BrainDisease} >$ could be extracted while $\langle \mathtt{Arryth.}, \mathtt{BrainDisease} >$ and $\langle \mathtt{Myoc.Inf.}, \mathtt{BrainDisease} >$ may have a too low support.

Although Srikant and Agrawal [14] early introduced hierarchy management in the extraction of association rules and sequential patterns, their approach was not scalable in a multidimensional context. Han et al [5] proposed a method for mining multiple level association rules in large databases. But their approach could not extract patterns containing items from different levels in the taxonomy. Plantevit et al [13] proposed M3SP, a method taking both multilevel and multidimensional aspects into account. M3SP is able to find sequential patterns with the most appropriate level of granularity.

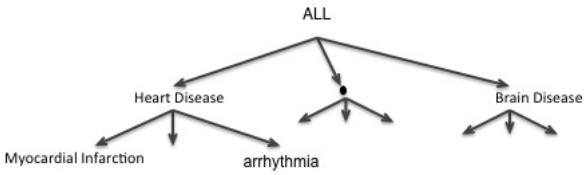


Fig. 2. disease's taxonomy

The PMSI is a multidimensional database holding information coded with controled vocabularies and taxonomies. Therefore, we relied on M3SP to extract multilevel and multidimensional sequential patterns. Nevertheless, the M3SP paradigm is still the search of frequent patterns. As our objective is to discover interesting patterns that may be infrequent, we ran M3SP iteratively until very low support thresholds. (See appendix for more details about M3SP and how we used it). This produced massive amounts of patterns requiring further processing for a practical interpretation by a domain expert. This next phase was conducted with a lattice-based classification of sequential patterns described in the following section.

3.2 Formal Concept Analysis

Introduced by Wille [17], Formal Concept Analysis is based on the mathematical order theory. FCA has successfully been applied to many fields, such as medicine and psychology, musicology, linguistic databases, information science, software engineering .... A strong feature of Formal Concept Analysis is its capability of producing graphical visualizations of the inherent structures among data.

FCA starts with a formal context $\mathbb{K} = (\mathtt{G}, \mathtt{M}, \mathtt{I})$ where $\mathtt{G}$ is a set of objects, $\mathtt{M}$ is a set of attributes, and the binary relation $\mathtt{I} = \mathtt{G} \times \mathtt{M}$ specifies which objects have which attributes. Two operators, both denoted by $'$, connect the power sets of objects $2^{\mathtt{G}}$ and attributes $2^{\mathtt{M}}$ as follows:

$$' : 2^{\mathtt{G}} \rightarrow 2^{\mathtt{M}}, \mathtt{X}' = \{\mathtt{m} \in \mathtt{M} | \forall \mathtt{g} \in \mathtt{X}, \mathtt{gIm}\}$$

$$' : 2^{\mathtt{M}} \rightarrow 2^{\mathtt{G}}, \mathtt{Y}' = \{\mathtt{g} \in \mathtt{G} | \forall \mathtt{m} \in \mathtt{Y}, \mathtt{gIm}\}$$

The operator $'$ is dually defined on attributes. The pair of $'$ operators induces a Galois connection between $2^{\mathtt{G}}$ and $2^{\mathtt{M}}$. The composition operators $''$ are closure operators: they are idempotent, extensive and monotonous. For any $\mathtt{A} \subseteq \mathtt{G}$ and $\mathtt{B} \subseteq \mathtt{M}$, A'' and B'' are closed sets whenever $\mathtt{A} = \mathtt{A}''$ and $\mathtt{B} = \mathtt{B}''$.

A formal concept of the context $\mathbb{K} = (\mathtt{G}, \mathtt{M}, \mathtt{I})$ is a pair $(\mathtt{A}, \mathtt{B}) \subseteq \mathtt{G} \times \mathtt{M}$ where $\mathtt{A}' = \mathtt{B}$ and $\mathtt{B}' = \mathtt{A}$. A is called the *extent* and $\mathtt{B}$ is called the *intent*. A concept $(\mathtt{A}_1, \mathtt{B}_1)$ is a *subconcept* of a concept $(\mathtt{A}_2, \mathtt{B}_2)$ if $\mathtt{A}_1 \subseteq \mathtt{A}_2$ (which is equivalent to $\mathtt{B}_2 \subseteq \mathtt{B}_1$) and we write $(\mathtt{A}_1, \mathtt{B}_1) \leq (\mathtt{A}_2, \mathtt{B}_2)$. The set $\mathfrak{B}$ of all concepts of a formal context $\mathbb{K}$ together with the partial order relation $\leq$ forms a lattice and is called concept lattice of $\mathbb{K}$. This lattice can be represented as a Hasse diagram providing a visual support for interpretation.

4 Classification and selection of interesting care trajectories

We use FCA to classify and filter the results of the sequential mining step. The formal context is built by taking patients as objects, and sequential patterns as attributes. A patient $\mathtt{p}$, considered as a sequence, is related to a sequential pattern $\mathtt{s}$ if $\mathtt{p}$ contains $\mathtt{s}$. Table 4 shows a formal context K_{PS} representing the binary relation between the patients and the sequences. The cross indicates that the patient has passed completely in the sequence of the health facilities. Thus, we achieve a classification of patients according to their trajectories of care.

	Seq_1	Seq_2	Seq_3	Seq_4
P_1	x	x	x	
P_2			x	
P_3		x	x	x
P_4		x		

Table 1.formal context K_{PS}

In order to choose the most important concepts, we rely on stability, a measure of interest introduced in [8] and revisited in [9].

Let $(\mathtt{A}, \mathtt{B})$ be a formal concept of $\mathfrak{B}$. Stability of $(\mathtt{A}, \mathtt{B})$ is defined as:

$$\gamma(\mathtt{A}, \mathtt{B}) = \frac{|\{\mathtt{C} \subseteq \mathtt{A} | \mathtt{C}' = \mathtt{A}' = \mathtt{B}\}|}{2^{|\mathtt{A}|}}$$

The stability index of a concept indicates how much the concept intent depends on particular objects of the extent. It indicates the probability of preserving concept intent while removing some objects of its extent. A stable concept continues to be a concept even if a few members stop being members. This means also that a stable concept is resistant to noise and will not collapse when some members are removed from its extent.

Stability offers an alternative point of view on concepts compared to the well known metric of support based on frequency, which is noticely used to build iceberg lattices [15]. Actually, combining support and stability allows a more subtle interpretation, as shown in a previous work in the same application domain [6].

5 Results

5.1 Patient healthcare trajectories

The PMSI is a relational database holding informations for any hospitalization in France. We reconstituted patient care trajectories from PMSI data considering each stay as an itemset. The sequence of stays for a same patient defines his care trajectory. In our experiment, itemsets could be made of various combinations of dimensions. Table 2 shows the trajectories of care obtained using two dimension (principal diagnosis, hospital ID). For example (C341,210780581) represents one hospitalization for a patient

Patient	Sequence
p1	$\langle(C341, 750712184)(Z452, 580780138)(D122, 030785430)\ldots\rangle$
p2	$\langle(C770, 100000017)(C770, 210780581)(Z080, 210780581)\ldots\rangle$
p3	$\langle(H259, 210780110)(H259, 210780110)(K804, 210010070)\ldots >$
p4	$\langle(R91, 210780136)(C07, 210780136)(C341, 210780136)\ldots\rangle$

Table 2. Care trajectories of 4 patients showing principal diagnoses and hospital IDs

in the University Hospital of Dijon (coded as 210780581) treated for a lung cancer (C341). Our dataset contained 486 patients suffering from lung cancer and living in the French region of Burgundy.

Table 3 shows some of the patterns generated by M3SP with the data presented in Table 2 using taxonomies of Figure 1. Pattern 3 can be interpreted as follows: 36% of patients have a hospitalization in a private institution (CL), for any kind of principal diagnosis (ALL). Then, 3 hospitalizations follow with the same principal diagnosis (Z511 coding for chemotherapy). That kind of pattern demonstrates the interest of multilevel and multidimensional sequential pattern mining: though principal diagnosis are the same in the third last stays, hospitals can be different. Mining at the lowest level of granularity, without taxonomies, would generate many different patterns with lower support.

ID	Support	Pattern
1	100%	$\langle(All, All)\rangle$
2	65%	$\langle(Z511, All)(Z511, All)(Z511, All)\rangle$
3	36%	$\langle(All, CL)(Z511, All)(Z511, All)(Z511, All)\rangle$
4	21%	$\langle(Z511, CH)(Z511, CH)(Z511, CH)(Z511, CH)(Z511, CH)\rangle$

Table 3. Example of sequential patterns generated by M3SP

However, for low support thresholds, the number of extracted patterns dramatically grows with the size of the database, depending on the number of patients, the size of the taxonomies and the number of dimensions as shown in Table 4.

Dimensions used	Number of patterns
`Institutions`	1529
`Principal Diagnosis, Institution`	4051
`All diagnoses`	50546
`Institutions, Medical Procedures`	293402

Table 4. Number of patterns generated by M3SP (`minsup`=5%)

The next step consists in building a lattice with the resulting sequential patterns in order to facilitate interpretation and selection of interesting care trajectories.

5.2 Lattice-based classification of sequential patterns

We illustrate this approach with patterns representing the sequences of institutions that are frequent in the patients set. We built a formal context relating 486 patients and 1529 sequential patterns. These sequences are generated in the first experimental by considering only one dimension (healthcare institutions). It is characterized with a taxonomy with two levels of granularity. We iteratively applied M3SP, decreasing threshold by one patient at each step. The resulting lattice has 10145 concepts organized on 48 different levels. Figure 3 shows the upper part of the lattice. Concepts intents are sets of one or more sequential patterns. From the lowest right concept, we can see that 37 patients support 3 sequential patterns:

- at least one hospitalization in the hospital 690781810
- they were hospitalized at least once in a University Hospital (CHU/CHR)
- they had at least 2 hospitalization, for simplicity, 2*(ALL) is the contraction of (ALL)(ALL).

The intent of top concept is $\langle(ALL)\rangle$, because all patients have at least one hospitalization during their treatment. The intent of co-atoms (i.e. immediate descendant of top) is always a sequence of length one, holding items of high level of granularity.

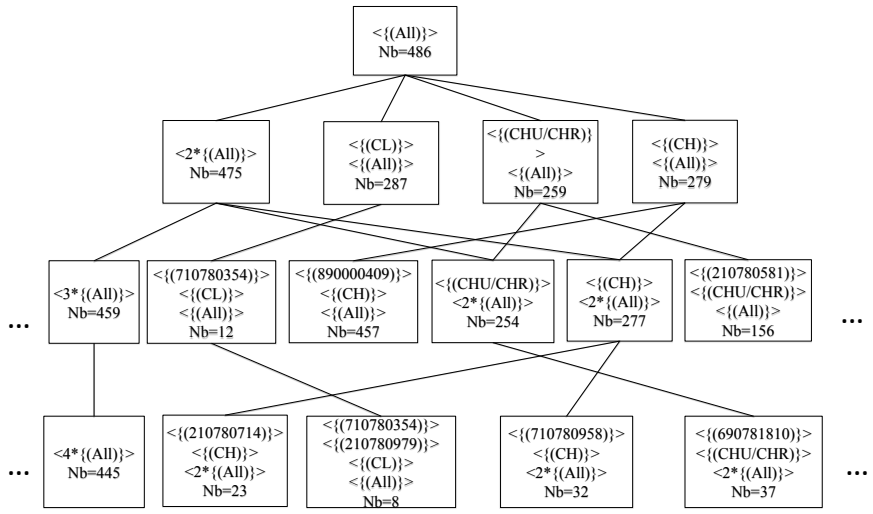


Fig. 3. Lattice of sequences of healthcare institutions

Filtering concepts can be achieved using both support and stability. In order to highlight the interesting properties of stability, we try to answer the question "is there a number of hospitalizations that characterizes care trajectories for lung cancer?". A basic scheme in lung cancer treatment consists generally in a sequence of 4 chemotherapy sessions possibly following a surgical operation. Due to noise in data or variability in

practices, we may observe sequences of 4, 5, 6 or more stays in the PMSI database. Mining such data with an *a priori* fixed support threshold may not discover the most interesting patterns. If the threshold is too high, we simply miss the good pattern. If it is too low, similar patterns, differing only in length, with close values of support can be extracted. Figure 4 shows the power of stability in discriminating such patterns. The concept with intent $\langle (CL)\rangle\langle 2 * (ALL)\rangle$ is the most frequent. It represents patients with at least a stay in a private organization, and at least 2 stays in hospital. Similar concepts have a relatively close support, and differ only in the total number of stays. The concept with 5 stays has the highest stability. This probably matches the basic treatment scheme of lung cancer. Our interpretation relies on the power of stability to point out noisy concepts. Actually, only a few patients in concept $\langle (CL)\rangle\langle 2 * (ALL)\rangle$ have only 2 stays.

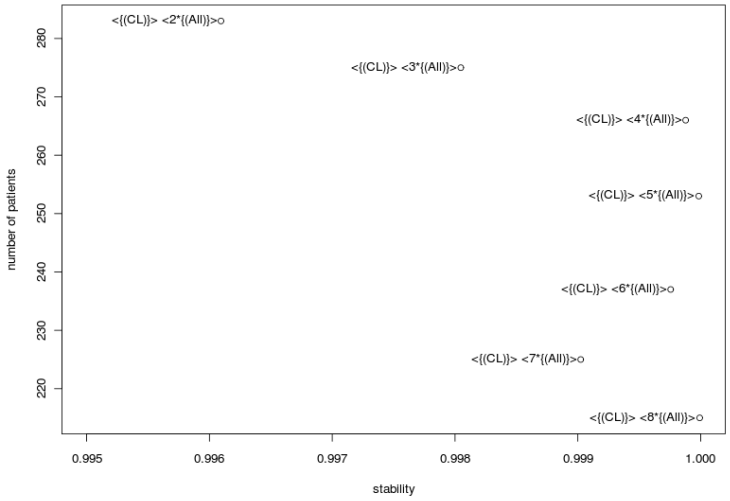


Fig. 4. Discriminating power of stability: scatter plot of support and stability of concepts (represented by their intent)

Another interesting feature of lattice-based classification of sequential patterns lies in its ability to characterize objects by several patterns. Let consider the minimal database of sequences $\mathtt{D} = \{\mathtt{s}_1 = \langle(\mathtt{a})(\mathtt{b})(\mathtt{c})\rangle; \mathtt{s}_2 = \langle(\mathtt{a})(\mathtt{c})(\mathtt{b})\rangle : \mathtt{s}_3 = \langle(\mathtt{d})\rangle\}$. With a 2/3 threshold, $\langle(\mathtt{a})(\mathtt{b})\rangle$ and $\langle(\mathtt{a})(\mathtt{c})\rangle$ are considered as frequent sequential patterns, but sequential pattern mining will give no information about the fact that all sequences containing the pattern $\langle(\mathtt{a})(\mathtt{b})\rangle$ contain also the pattern $\langle(\mathtt{a})(\mathtt{c})\rangle$. However this information can be obtained by classifying sequential patterns with FCA.

6 Conclusion

In this paper we propose an original combination of sequential pattern mining and FCA to explore a database of multidimensional sequences. We show some interesting properties of concept lattices and stability index to classify and select interesting sequential patterns. This work is in a early step. Further developments can be made in several axes. First, other measures of interest could be investigated to qualify sequential patterns. Furthermore, connexions between FCA and the sequential mining problem could be explored in a more integrative approach, especially by studying closure operators on sequences.

7 Acknowledgments

The authors wish to thank the TRAJCAN project for its financial support and Mrs. Catherine QUANTIN, the responsible of TRAJCAN project at university hospital of Dijon.

References

1. Abidi, S.S.: Knowledge management in healthcare: towards 'knowledge-driven' decision-support services. Int J Med Inform 63(1-2), 5–18 (Sep 2001)
2. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Yu, P.S., Chen, A.S.P. (eds.) Eleventh International Conference on Data Engineering. pp. 3–14. IEEE Computer Society Press, Taipei, Taiwan (1995), citeseer.ist.psu.edu/agrawal95mining.html
3. Ayres, J., Gehrke, J., Yiu, T., Flannick, J.: Sequential pattern mining using a bitmap representation. pp. 429–435. ACM Press (2002)
4. ying Chiu, D., hung Wu, Y., Chen, A.L.P.: An efficient algorithm for mining frequent sequences by a new strategy without support counting. In: In Proceedings of the 20th International Conference on Data Engineering (ICDE'04. pp. 375–386. IEEE Computer Society (2004)
5. Han, J., Fu, Y.: Mining multiple-level association rules in large databases. Knowledge and Data Engineering, IEEE Transactions on 11(5), 798 –805 (sep/oct 1999)
6. Jay, N., Kohler, F., Napoli, A.: Analysis of social communities with iceberg and stability-based concept lattices. In: Medina, R., Obiedkov, S.A. (eds.) International Conference on Formal Concept Analysis (ICFCA'08). LNAI, vol. 4923, pp. 258–272. Springer (2008)
7. Kuznetsov, S., Obiedkov, S., Roth, C.: Reducing the representation complexity of lattice-based taxonomies. In: Priss, U., Polovina, S., Hill, R. (eds.) Proc. of ICCS 15th Intl Conf Conceptual Structures. LNCS/LNAI, vol. 4604, pp. 241–254. Springer (2007)
8. Kuznetsov, S.O.: Stability as an estimate of the degree of substantiation of hypotheses derived on the basis of operational similarity. Nauchn. Tekh. Inf., Ser.2 (Automat. Document. Math. Linguist.) 12, 21–29 (1990)
9. Kuznetsov, S.O.: On stability of a formal concept. Annals of Mathematics and Artificial Intelligence 49, 101–115 (2007), http://www.springerlink.com/content/fk1414v361277475/

10. Masseglia, F., Cathala, F., Poncelet, P.: The psp approach for mining sequential patterns. pp. 176–184 (1998)
11. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.: Prefixspan: Mining sequential pattern by prefix-projected growth. In: ICDE. pp. 215–224 (2001)
12. Pinto, H., Han, J., Pei, J., Wang, K., Chen, Q., Dayal, U.: Multi-dimensional sequential pattern mining. In: CIKM '01: Proceedings of the tenth international conference on Information and knowledge management. pp. 81–88. ACM Press, New York, NY, USA (2001)
13. Plantevit, M., Laurent, A., Laurent, D., Teisseire, M., Choong, Y.W.: Mining multidimensional and multilevel sequential patterns. ACM Trans. Knowl. Discov. Data 4(1), 1–37 (2010)
14. Srikant, R., Agrawal, R.: Mining sequential patterns: Generalizations and performance improvements. In: Apers, P.M.G., Bouzeghoub, M., Gardarin, G. (eds.) Proc. 5th Int. Conf. Extending Database Technology, EDBT. vol. 1057, pp. 3–17. Springer-Verlag (25–29 1996), http://citeseer.ist.psu.edu/article/srikant96mining.html
15. Stumme, G.: Efficient data mining based on formal concept analysis. In: Lecture Notes in Computer Science, vol. 2453, p. 534. Springer (Jan 2002)
16. Valtchev, P., Missaoui, R., Godin, R.: Formal concept analysis for knowledge discovery and data mining: The new challenges. In: Eklund, P.W. (ed.) ICFCA. Lecture Notes in Computer Science, vol. 2961, pp. 352–371. Springer (2004)
17. Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Rival, I. (ed.) Ordered Sets. Reidel (1982)
18. Yang, Z., Kitsuregawa, M., Wang, Y.: Paid: Mining sequential patterns by passed item deduction in large databases. In: IDEAS'06. pp. 113–120 (2006)
19. Yu, C.C., Chen, Y.L.: Mining sequential patterns from multidimensional sequence data. Knowledge and Data Engineering, IEEE Transactions on 17(1), 136 – 140 (jan 2005)
20. Zaki, M.J.: Spade: An efficient algorithm for mining frequent sequences. Machine Learning 42(1-2), 31–60 (January 2001), http://www.springerlink.com/link.asp?id=n3t642725v615427
21. Zhang, C., Hu, K., Chen, Z., Chen, L., Dong, Y.: Approxmgmsp: A scalable method of mining approximate multidimensional sequential patterns on distributed system. In: Fuzzy Systems and Knowledge Discovery, 2007. FSKD 2007. Fourth International Conference on. vol. 2, pp. 730 –734 (aug 2007)

Appendix

The M3SP algorithm is able to extract sequential patterns characterized by several dimensions with different levels of granularity for each dimension [13]. Each dimension has a taxonomy which defines the hierarchical relations between items. M3SP runs in three steps: data pre-processing , MAF-item generation and sequence mining.

In Figure 5, we present an example to illustrate the mechanism of M3SP. Table 5b shows a dataset of hospitalizations relating patients (P) with attributes from three dimensions

- T, the date of stay,
- H, the healthcare setting in which the hospitalization takes place,
- D, the disease of the patient.

For instance, the first tuple means that, at date 1, the patient 1 has been treated for the disease $\mathtt{D}_{11}$ in hospital $\mathtt{H}_{11}$. Let us now assume that we want to extract all multidimensional sequences that deal with hospitals and diseases that are frequent in the patients set. Figure 5a displays a taxonomy for dimensions H and D.

Pre-processing step

M3SP considers three types of dimensions: a temporal dimension $\mathtt{D}_\mathtt{t}$, a set of analysis dimensions $\mathtt{D}_\mathtt{A}$, and a set of reference dimensions $\mathtt{D}_\mathtt{R}$. M3SP orders the dataset according to $\mathtt{D}_\mathtt{t}$. The tuples appearing in a sequence are defined over the dimensions of $\mathtt{D}_\mathtt{A}$. The support of the sequences is computed according to dimensions of $\mathtt{D}_\mathtt{R}$. M3SP splits the dataset into blocks according to distinct tuple values over reference dimensions. The support of a given multidimensional sequence is the ratio of the number of blocks supporting the sequence over the total number of blocks. In our example, H (hospitals) and D (diseases) are the analysis dimensions, T is the temporal dimension and P (patients) is the only reference dimension. We obtain two blocks defined by $\mathtt{Patient}_1$ and $\mathtt{Patient}_2$. as shown in table 5c.

MAF-item generation step

In this step, M3SP generates all the Maximal Atomic Frequent items or MAF-items. In order to define MAF-items, we fisrt define the specificity relation between items.

Specificity relation. Given two multidimensional items $\mathtt{a} = (\mathtt{d}_1, ..., \mathtt{d}_\mathtt{m})$ and $\mathtt{a}' = (\mathtt{d}'_1, ..., \mathtt{d}'_\mathtt{m})$, $\mathtt{a}'$ is said to be more specific than $\mathtt{a}$, denoted by $\mathtt{a} \preccurlyeq_\mathtt{I} \mathtt{a}'$, if for every $\mathtt{i} = 1, ..., \mathtt{m}$, $\mathtt{d}'_\mathtt{i} \in \mathtt{d}_\mathtt{i} \downarrow$. Where $\mathtt{d}_\mathtt{i} \downarrow$ is the set of all direct specializations of $\mathtt{d}_\mathtt{i}$ according to the dimension taxonomy of d_i. In our example, we have $(\mathtt{H}_1, \mathtt{D}_1) \preccurlyeq_\mathtt{I} (\mathtt{H}_1, \mathtt{D}_{11})$, because $\mathtt{H}_1 \in \mathtt{H}_1 \downarrow$ and $\mathtt{D}_1 \in \mathtt{D}_{11} \downarrow$.

MAF-item. An atomic item $\mathtt{a}$ is said to be a Maximal Atomic Frequent item, or a MAF-item, if $\mathtt{a}$ is frequent and if for every $\mathtt{a}'$ such that $\mathtt{a} \preccurlyeq_\mathtt{I} \mathtt{a}'$, the item $\mathtt{a}'$ is not frequent. In our example, if we consider $\mathtt{minsup} = 100\%$, $b = (H_1, D_1)$ is a MAF-item, because it is frequent and there is not another item as frequent and more specific than b.

The computation of MAF-items is represented by a tree in which the nodes are of the form $(\mathtt{d}_1, \mathtt{d}_2)_\mathtt{s}$, meaning that $(\mathtt{d}_1, \mathtt{d}_2)_s$ is an atomic item with support s as we

show in Figure 5d. In this tree, MAF-items are displayed as boxed nodes. We note that all leaves are not necessarily MAF-items. For example, $(\mathtt{H}_2, \mathtt{D}_{21})_{100\%}$ is a leaf, but not a MAF-item. This is because $(\mathtt{H}_2, \mathtt{D}_{21})_{100\%} \preccurlyeq_{\mathtt{I}} (\mathtt{H}_{21}, \mathtt{D}_{21})_{100\%}$ and $(\mathtt{H}_{21}, \mathtt{D}_{21})$ has been identified as being an MAF-item.

Sequence mining step

Frequent sequences can be mined using any standard sequential pattern-mining algorithm (PrefixSpan in this work). Since in such algorithms, the dataset to be mined is a set-pairs of the form (id, seq), where id is a sequence identifier and seq is a sequence of itemsets, our example dataset is transformed as follows :

- every MAF-item is associated with a unique identifier denoted by $ID(a)$ (table 5e), playing the role of the items in standard algorithms.
- every block b is assigned a patient identifier $ID(p)$, playing the role of the sequence identifiers in standard algorithms,
- every block b transformed into a pair $(ID(b), \zeta(b))$, where $\zeta(b)$ is a sequence. (table 5f)

PrefixSpan is run over table 5f. By considering a support threshold `minsup` =50%, table 5g displays all the frequent sequences in their transformed format as well in their multidimensional format in which identifiers are replaced with their actual values.

The basic step in M3SP method is MAF-item generation, because it provides all multidimensional items that occur in sequences to be mined. If the set of MAF-items is changed, the sequence will be changed. M3SP always extracts the most specific multidimensional items.

For example $(\mathtt{H}_1, \mathtt{D}_1)$ is frequent according to `minsup`=50%, but another item, $(\mathtt{H}_{11}, \mathtt{D}_{11})$ is more specific and still frequent. As a result, $(\mathtt{H}_1, \mathtt{D}_1)$ is not a MAF-item and consenquently not used to build squences. Finaly the frequent sequence $\langle\{(\mathtt{H}_1, \mathtt{D}_1), (\mathtt{H}_{21}, \mathtt{D}_{21})\}\rangle$ does not appear in the results of M3SP. However, tables 5 and 6 show the MAF-items set and the frequent sequences extracted by M3SP at a 100% threshold. It can be noticed that (H_1, D_1) is a MAF-item and that the sequence $\langle\{(\mathtt{H}_1, \mathtt{D}_1), (\mathtt{H}_{21}, \mathtt{D}_{21})\}\rangle$ is generated.

Thus, given two `minsup` thresholds $\sigma^{'} < \sigma$. The set of frequent sequences obtained for σ' may not always contain the set of sequences obtained for σ.

Considering this as a limit in our approach as we wanted to extract both general and specific sequences, we iteratively applied M3SP, decreasing threshold by one patient at each step. This allowed us to extract more potentially interesting sequences than by using a single low `minsup` threshold.

MAF-item
$(\mathtt{H}_1, \mathtt{D}_1)$
$(\mathtt{H}_{21}, \mathtt{D}_{21})$

Table 5. maf-item,`minsup` =100%

Frequent Multidimensional Sequences
$\langle\{(\mathtt{H}_1, \mathtt{D}_1)\}\rangle$
$\langle\{(\mathtt{H}_{21}, \mathtt{D}_{21})\}\rangle$
$\langle\{(\mathtt{H}_1, \mathtt{D}_1), (\mathtt{H}_{21}, \mathtt{D}_{21})\}\rangle$

Table 6. Sequences for `minsup` =100%

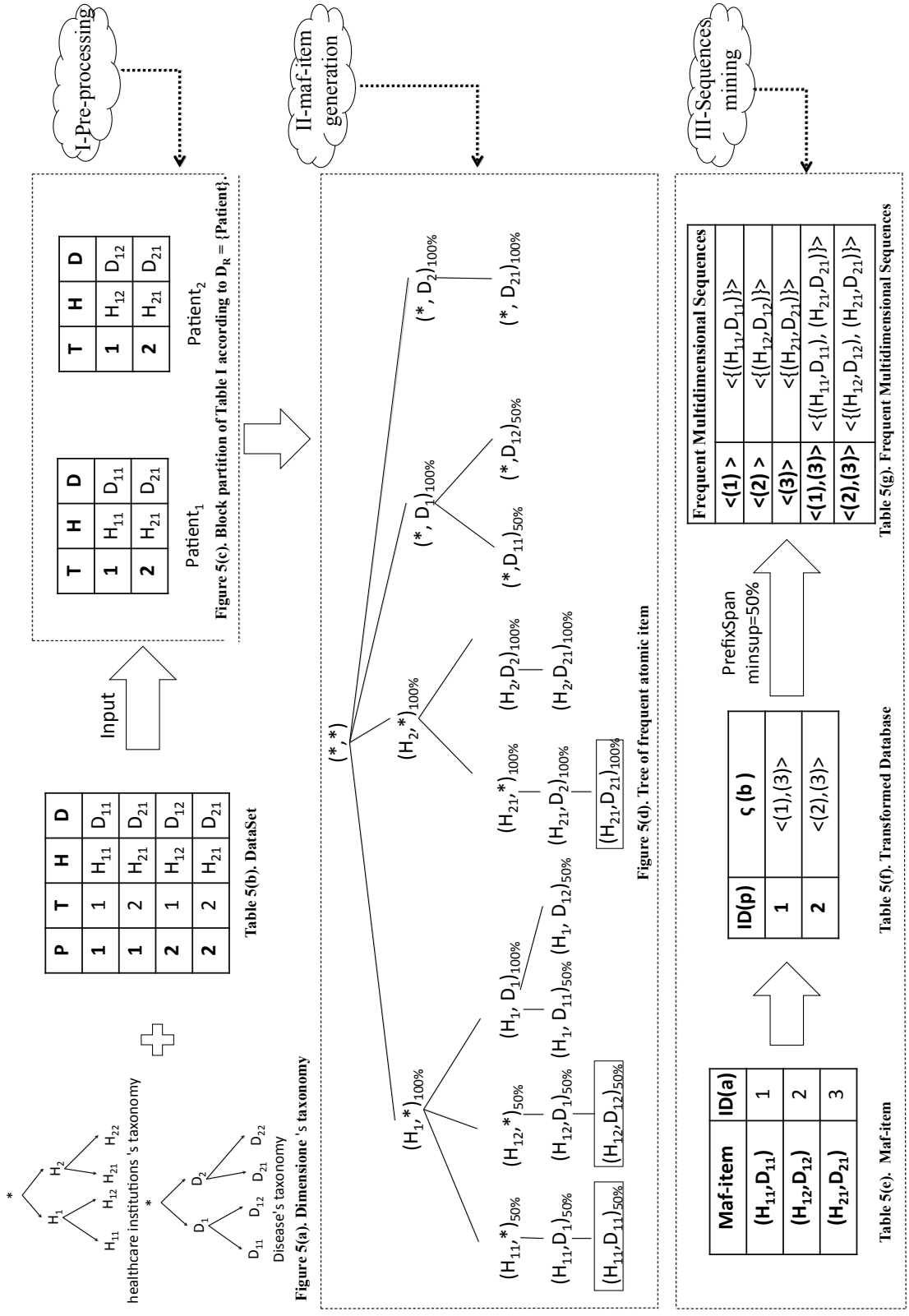


Figure 5(a). Dimensione 's taxonomy

P	T	H	D
1	1	H_{11}	D_{11}
1	2	H_{21}	D_{21}
2	1	H_{12}	D_{12}
2	2	H_{21}	D_{21}

Table 5(b). DataSet

T	H	D
1	H_{11}	D_{11}
2	H_{21}	D_{21}

Figure 5(c). Block partition of Table I according to D_R = {Patient}.

T	H	D
1	H_{12}	D_{12}
2	H_{21}	D_{21}

Figure 5(d). Tree of frequent atomic item

Maf-item	ID(a)
(H_{11},D_{11})	1
(H_{12},D_{12})	2
(H_{21},D_{21})	3

Table 5(e). Maf-item

ID(p)	ς (b)
1	<(1),(3)>
2	<(2),(3)>

Table 5(f). Transformed Database

Frequent Multidimensional Sequences	
<(1) >	<{$(H_{11},D_{11}$)}>
<(2) >	<{$(H_{12},D_{12}$)}>
<(3)>	<{$(H_{21},D_{21}$)}>
<(1),(3)>	<{$(H_{11},D_{11}$), $(H_{21},D_{21}$)}>
<(2),(3)>	<{$(H_{12},D_{12}$), $(H_{21},D_{21}$)}>

Table 5(g). Frequent Multidimensional Sequences

Fig. 5. example for M3SP method, `minsup` =50%

Querying Relational Concept Lattices

Z. Azmeh[1], M. Huchard[1],
A. Napoli[2], M. Rouane-Hacene[3], and P. Valtchev[3]

[1] LIRMM, 161, rue Ada, F-34392 Montpellier Cedex 5
[2] LORIA, B.P. 239, F-54506 Vandœuvre-lès-Nancy
[3] Dépt. d'informatique, UQÀM, C.P. 8888, Succ. Centre-Ville Montréal, Canada

Abstract. Relational Concept Analysis (RCA) constructs conceptual abstractions from objects described by both own properties and inter-object links, while dealing with several sorts of objects. RCA produces lattices for each category of objects and those lattices are connected via relational attributes that are abstractions of the initial links. Navigating such interrelated lattice family in order to find concepts of interest is not a trivial task due to the potentially large size of the lattices and the need to move the expert's focus from one lattice to another. In this paper, we investigate the navigation of a concept lattice family based on a query expressed by an expert. The query is defined in the terms of RCA. Thus it is either included in the contexts (modifying the lattices when feasible), or directly classified in the concept lattices. Then a navigation schema can be followed to discover solutions. Different navigation possibilities are discussed.

Keywords: Formal Concept Analysis, Relational Concept Analysis, Relational Queries.

1 Introduction

Recently [1], we worked on the problem of selecting suitable Web services for instantiating an abstract calculation workflow. This workflow can be seen as a DAG whose nodes are abstract tasks (like *book a hotel room*) and directed edges are connections between the tasks, which often correspond to a data flow (like connecting *reserve a train ticket* and *book a hotel room*: train dates and timetable are transmitted from *reserve a train ticket* to *book a hotel room*). The selection is based on quality-of-service (QoS) properties like response time or availability and on the composability quality between services chosen for neighbor tasks in the workflow. Besides, we aim at identifying and storing a set of backup services adapted to each task. To be efficient in the replacement of a failing Web service by another, we want to organize each set of backup Web services by a partial order that expresses the quality criteria and helps to choose a good trade-off for instantiating the abstract workflow. Analyzing such multi-relational data is a complex problem, which can be approached by various methods including querying, visualization, statistics, or rule extraction (data mining).

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

We proposed an approach based on Relational Concept Analysis (an iterative version of Formal Concept Analysis) to solve this problem, because of its multi-relational nature. Web services are filtered and grouped by tasks they may satisfy (*e. g.* the Web services for booking a hotel room). In formal contexts (one for each task), we associate the Web services and their QoS criteria. For example, the service *HotelsService* by *lastminutetravel.com* would be described by *low response time*, *medium availability* (classical scaling is applied to the QoS values). In relational contexts we encode the composability levels in each directed edge of the workflow. Given an edge of the workflow, the composition quality depends on the way output data of the source task cover input data of the ending task, and the need for data adaptation. A relational context encodes for example the relation *Adaptable-Fully-Composable* between services for *reserve a train ticket* and services for *book a hotel room*. In this relation *TravelService* by *puturist.com* is connected to *HotelsService* by *lastminutetravel.com* if output data of *TravelService* can be used, with a slight adaptation, to fill input data of *HotelsService*.

The concept lattice family we obtain (one Web service lattice for each task of the workflow) makes it possible: (1) to select a Web service for each task based on QoS and composability criteria, (2) to memorize classified alternatives for each task.

Due to the nature of our problem, we are interested in classifying independently the Web services corresponding to the tasks and not classifying the solutions. By solution, we mean a set of Web services, each of which can instantiate a task of the workflow. If a particular service fails or is no more available, the goal is to constitute a new working combination out of the old one, with the smallest number of service replacements. To the best of our knowledge, this problem area has not been investigated in-depth prior to our study, especially in the context of Relational Context Analysis [7, 6]. Therefore, we believe that it would be useful to generalize and report what we learned in our experience. In more general terms, we have multi-relational data and a question which contains variables we want to instantiate, and we aim at:

- Finding a specific set of objects that satisfy the query. An answer is composed of objects, each object instantiates one variable;
- Classifying, for each variable, the objects depending on the way they satisfy (or not) the query, to find alternative answers.

In this paper, we put the problem in a more general framework, which assumes an unrestricted relational context family and a query given by an expert. The query can be seen as a DAG, where some nodes are labelled by variables and some others are labelled by objects. The nodes roughly correspond to the formal (object-attribute) contexts and the edges correspond to the relational (object-object) contexts. A set of lattices is built using Relational Concept Analysis and the existential scaling operator. We assume that an expert gives a total ordering of the edges of the DAG. Then an algorithm navigates the lattices following this ordering. This navigation allows us to determine objects that answer the query.

These objects with their position in the lattices are what the expert wants to explore, to extract a solution and store the alternatives.

In the following, Section 2 reminds the main principles of Relational Concept Analysis (RCA). Section 3 defines the model of queries in the RCA framework that we consider in this paper. Section 4 presents and discusses an algorithm that navigates the concept lattice family using a query. Related work is presented in Section 5 and we conclude in Section 6.

2 Background on Relational Concept Analysis

For FCA, we use the notations of [4]. In RCA [5], the objects are classified not only according to the attributes they share, but also according to the links between them. Let us take the following case study. We consider a list of countries, a list of restaurants, a list of Mexican dishes, a list of ingredients, and finally a list of salsas. We impose some relations between these entities {Country, Restaurant, MexicanDish, Ingredient, Salsa}, such that: a Country "has" a Restaurant; a Restaurant "serves" a MexicanDish; a MexicanDish "contains" an Ingredient; an Ingredient is "made-in" a Country; and finally a Salsa is "suitable-with" a MexicanDish. We express these entities and their relations by the DAG in Fig. 1. We capture an instantiation of this entity-relationship diagram in a relational context family.

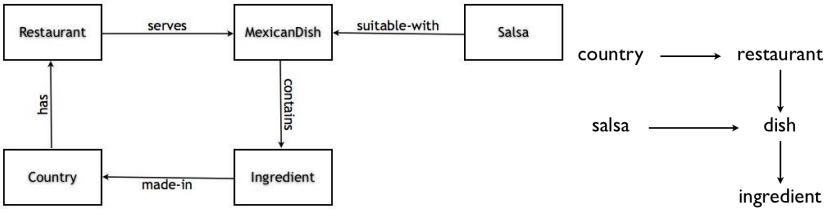


Fig. 1. The entities of the Mexican food example (left). The query schema (right)

Definition 1. *A relational context family* ***RCF*** *is a pair* $(\mathbb{K}, R)$ *where* $\mathbb{K}$ *is a set of formal (object-attribute) contexts* $K_i = (O_i, A_i, I_i)$ *and* R *is a set of relational (object-object) contexts* $r_{ij} \subseteq O_i \times O_j$*, where* O_i *(domain of* r_{ij}*) and* O_j *(range of* r_{ij}*) are the object sets of the contexts* K_i *and* K_j*, respectively.*

The RCF corresponding to our example contains five formal contexts and five relational contexts, illustrated in Table 1 (except the *made-in* relational context, which is not used in this paper for sake of simplicity). An RCF is used in an iterative process to generate at each step a set of concept lattices. First concept lattices are built using the formal contexts only. Then, in the following steps, a scaling mechanism translates the links between objects into

Table 1. Relational Context Family for mexican dishes

Country	ca	en	fr	lb	mx	es	us	America	Asia	Europe
Canada	×							×		
England		×								×
France			×							×
Lebanon				×					×	
Mexico					×			×		
Spain						×				×
USA							×	×		

Restaurant	r1	r2	r3	r4	r5	r6	r7
Chili's	×						
Chipotle		×					
El Sombrero			×				
Hard Rock				×			
Mi Casa					×		
Taco Bell						×	
Old el Paso							×

MexicanDish	d1	d2	d3	d4	d5	d6
Burritos	×					
Enchiladas		×				
Fajitas			×			
Nachos				×		
Quesadillas					×	
Tacos						×

Ingredient	i1	i2	i3	i4	i5	i6	i7	i8	i9	i10	i11	i12
chicken	×											
beef		×										
pork			×									
vegetables				×								
beans					×							
rice						×						
cheese							×					
guacamole								×				
sour-cream									×			
lettuce										×		
corn-tortilla											×	
flour-tortilla												×

Salsa	s1	s2	s3	s4	mild	medium-hot	hot
Fresh Tomato	×				×		
Roasted Chili-Corn		×				×	
Tomatillo-Green Chili			×			×	
Tomatillo-Red Chili				×			×

contains	chicken	beef	pork	vegetables	beans	rice	cheese	guacamole	sour-cream	lettuce	corn-tortilla	flour-tortilla
Burritos	×	×	×	×	×	×	×	×	×	×		×
Enchiladas	×						×		×		×	
Fajitas	×	×		×			×	×	×	×		×
Nachos				×	×		×	×				
Quesadillas	×	×					×				×	×
Tacos	×	×			×		×			×	×	×

has	Chili's	Chipotle	El Sombrero	Hard Rock	Mi Casa	Taco Bell	Old el Paso
Canada	×	×	×	×		×	
England		×		×		×	
France			×	×			×
Lebanon	×			×		×	
Mexico	×				×	×	
Spain				×		×	
USA	×	×	×	×	×	×	

serves	Burritos	Enchiladas	Fajitas	Nachos	Quesadillas	Tacos
Chili's			×		×	×
Chipotle	×					×
El Sombrero	×	×	×	×	×	×
Hard Rock			×	×		
Mi Casa	×	×		×	×	×
Taco Bell	×			×	×	×
Old el Paso						×

suitable-with	Burritos	Enchiladas	Fajitas	Nachos	Quesadillas	Tacos
Fresh Tomato	×	×	×	×	×	×
Roasted Chili-Corn	×			×		
Tomatillo-Green Chili	×			×		
Tomatillo-Red Chili	×	×	×	×	×	×

conventional FCA attributes and derives a collection of lattices whose concepts are linked by relations. For example, the existential scaled relation (that we will use in this paper) captures the following information: if an object o_s is linked to another object o_t, then in the scaled relation, this link is encoded in a relational attribute assigned to o_s. This relational attribute states that o_s is linked to a concept, which clusters o_t with other objects. This is used to form new groups, for example the group (See $Concept\_84$) of restaurants, which serve at least one dish containing sour cream (such dishes are grouped in $Concept\_75$). The steps are repeated until reaching the stability of lattices (when no more new concepts are generated). For mexican dishes, four lattices of the concept lattice family are represented in Figures 3 and 4. The ingredient lattice is presented in Fig. 2.

Definition 2. *Let $r_{ij} \subseteq O_i \times O_j$ be a relational context. The **exists** scaled relation $r_{ij}^{\exists}$ is defined as $r_{ij}^{\exists} \subseteq O_i \times \mathfrak{B}(O_j, A, I)$, such that for an object o_i and a concept c: $(o_i, c) \in r_{ij}^{\exists} \iff \exists\, x, x \in o_i' \cap Extent(c)$.*

In this definition, A is any set of attributes maybe including relational attributes, which are defined below.

Definition 3. *A relational attribute $(s\ r\ c)$ is composed of a scaling operator s (for example exists), a relation $r \in R$, and a concept c. It results from scaling a relation $r_{ij} \in R$ where $r_{ij} \subseteq O_i \times O_j$. It expresses a relation between the objects $o \in O_i$ with the concepts of $\mathfrak{B}(O_j, A, I)$. An existential relational attribute is denoted by $\exists r_{ij}\ c$ where $c \in \mathfrak{B}(O_j, A, I)$.*

For example: the *Concept_50* in the Country lattice owns the relational attribute $\exists$*has Concept_60*. This expresses that each country in *Concept_50* (Canada and USA) *has* at least a restaurant in *Concept_60* extent (El Sombrero or Mi Casa).

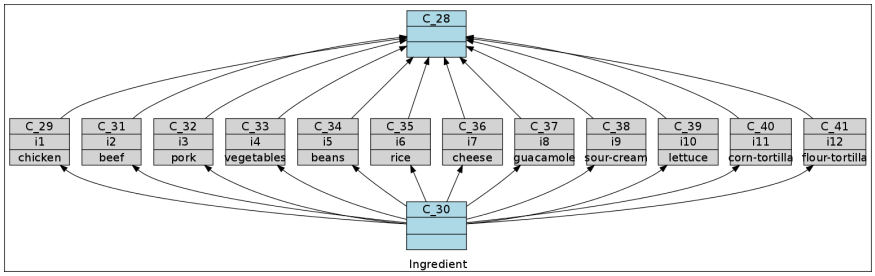


Fig. 2. The concept lattice for ingredients of the RCF in Table 1 (concepts names are reduced to C_n).

3 Introducing Relational Queries

In this section, we define the notion of *query* and *answer to a query*. First (section 3.1) we recall simple queries that help navigating concept lattices [7]. Then (section 3.2), we generalize to relational queries that lead the navigation across a concept lattice family.

3.1 Simple queries

Definition 4. *A query (including its answer) on a context $K = (O, A, I)$, denoted by $q|_K$ (or q when it is not ambiguous), is a pair $q = (o_q, a_q)$, such that o_q is the query object(s) i.e. the set of objects satisfying the query (or the answer set), and a_q is the set of attributes defining the constraint of the query. By definition, we have: $o_q' \supseteq a_q$, where $a_q \subseteq A$.*

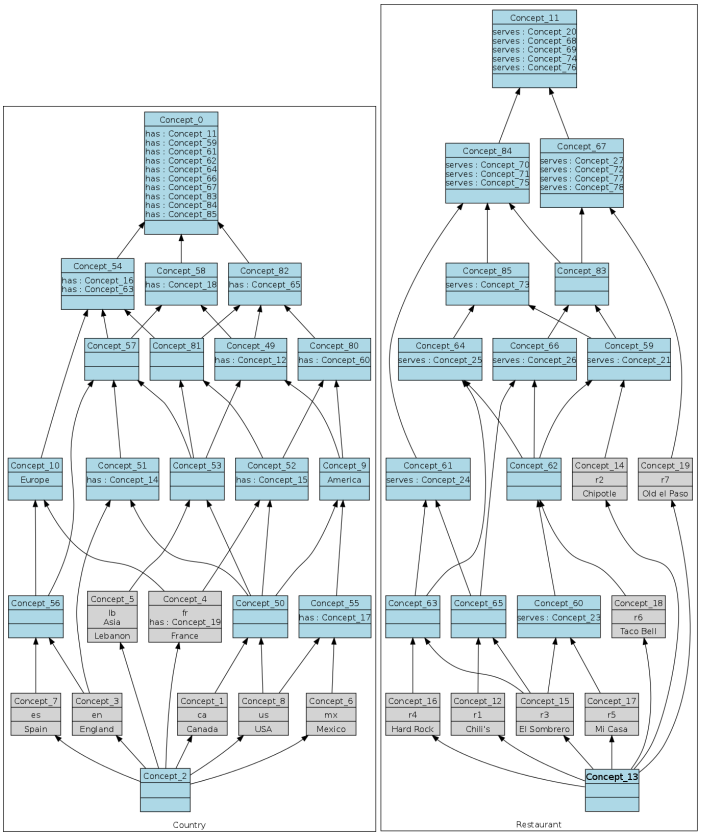


Fig. 3. Country and restaurant lattices for `exists` and the RCF in Table 1.

For example $q|_{K_{country}} = (\{England, France, Spain\}, \{Europe\})$ is a query on the country context (in Table 1), asking for countries in Europe. Another example $q|_{K_{MexicanDish}} = (\{\}, \{rice, corn\text{-}tortilla\})$

When a_q is closed, solving the query consists in finding the concept $C = (a'_q, a_q)$. To ensure that such a concept exists, a virtual query object ov_q that satisfies $ov'_q = a_q$ can be added to the context (as an additional line). Then, three types of answers can be interesting: the more precise answers are in a'_q, less constrained (with less attributes) answers are in extents of super-concepts of C, more constrained (with more attributes) answers are in extents of sub-concepts of C. When a_q is not closed, and we don't use the virtual query object, searching for answers needs to find first the more general concept C whose intent contains a_q. Now we will define more generally what we mean by relational queries.

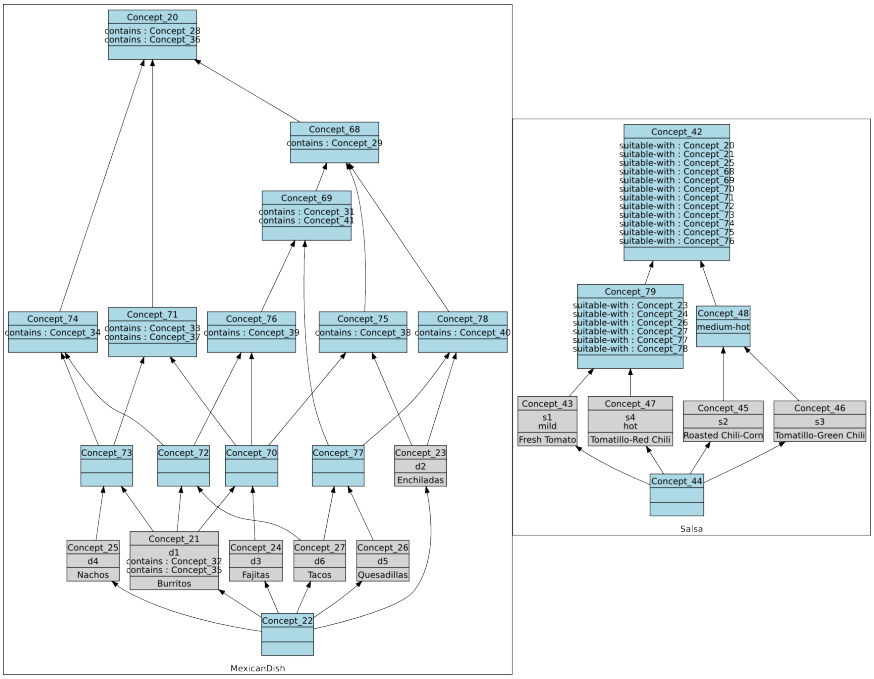


Fig. 4. Dishes and salsa lattices for `exists` and the RCF in Table 1.

3.2 Relational queries

In this study, a relational query is composed of several simple queries, to which we add relational constraints. The relational constraints are expressed via virtual query objects (variables), one for each formal context, where we want to find an object. A virtual query object may have relations (according to the relational contexts) with objects of other contexts, as well as with other virtual query objects.

Definition 5. *A relational query Q on a relational context family $(\mathbb{K}, R)$ is a pair $Q = (A_q, O_{vq}, R_q)$, where:*

1. *A_q is a set of simple queries*
   $A_q = \{q|_{K_i} = (o_q|_{K_i}, a_q|_{K_i}) \mid q|_{K_i}$ is a query on $K_i \in \mathbb{K}\}$
2. *There is a one-to-one mapping between A_q and O_{vq}, where O_{vq} is the set of virtual query objects.*
3. *R_q is a set of relational constraints $R_q = \{(o_{vq|_{K_i}}, r_{ij}, O_q)\}$, where $o_{vq|_{K_i}}$ is the virtual object associated with $q|_{K_i}$, $O_q \subseteq O_j \cup \{o_{vq|_{K_j}}\}$, with $o_{vq|_{K_j}}$ is the virtual object associated with K_j.*

For example, let us consider the following query: I am searching for a country with the attribute "fr", a restaurant in this country serving Mexican dish containing (chicken, cheese, and corn-tortilla), and a salsa which is "hot" and suitable with the dish. This query can be translated into a relational query $Q_{example} = (A_q, O_{vq}, R_q)$ as follows: $A_q = \{q_{country}, q_{rest.}, q_{dish}, q_{salsa}\}$, $a_{q_{country}} = \{fr\}$, $a_{q_{rest.}} = a_{q_{dish}} = \emptyset$, $a_{q_{salsa}} = \{hot\}$.
$O_{vq} = \{o_{vq_{dish}}, o_{vq_{country}}, o_{vq_{rest.}}, o_{vq_{salsa}}\}$
$R_q = \{(o_{vq_{dish}}, \textit{contains}, \{\textit{chicken}, \textit{cheese}, \textit{corn-tortilla}\}), (o_{vq_{country}}, \textit{has}, \{o_{vq_{rest.}}\}), (o_{vq_{rest.}}, \textit{serves}, \{o_{vq_{dish}}\}), (o_{vq_{salsa}}, \textit{suitable-with}, \{o_{vq_{dish}}\})\}$.
By definition, a query corresponds to the data model, and must respect the schema of the RCF (see in Fig. 1).

An answer to the relational query is included in the answers of the simple queries. For our example, the answers of the simple queries would be $o_{q_{country}} = \{France\}$, $o_{q_{rest.}}$ contains all the restaurants, $o_{q_{dish}}$ contains all the dishes, $o_{q_{salsa}} = \{Tomatillo\text{-}Red\ Chili\}$. If we consider these objects connected with the relations, this forms what we call the maximal answer graph. In this graph, we are interested in the subgraphs that cover the query (they have at least one object per element of A_q). These subgraphs are included in the graph of Fig. 5. There are various interesting forms of answer: having exactly one object per element of A_q, or having several objects per element of A_q.

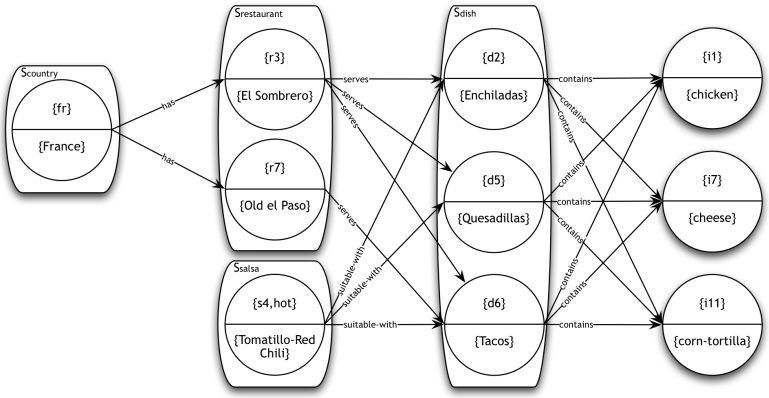


Fig. 5. The subgraph containing all the answers with the relations between the objects corresponding to the relational query example.

Definition 6. *An answer to a relational query $Q = (A_q, O_{vq}, R_q)$ is a set of objects X having a unique object per each context that is involved in the query:*

$$X =< o_i \mid o_i \in O_i \textit{ with } 1 \leq i \leq n >$$

These objects satisfy the query $Q = (A_q, O_{vq}, R_q)$, when they have the requested attributes: $\forall\ q|_{K_i} \in A_q,\ \exists\ o_i \in X\ :\ o_i' \supseteq a_{q|_{K_i}}$

and they are connected as expected:
$\forall\ (o_{vq|_{K_i}}, r, O_q) \in R_q$ *with* $r \subseteq O_i \times O_j$, (*and thus* : $O_q \subseteq O_j \cup \{o_{vq|_{K_j}}\}$) *and* $\forall\ o \in O_q$, *we have* :

1. *if* $o \in O_j$, *we have* $(o_i, o) \in r$
2. *if* $o = o_{v_{q|_{K_j}}}$, *we have* $(o_i, o_j) \in r$ *with* $o_j \in X \cap O_j$

For our example, the set of answers to the relational query, is:
$\{\{France,\ El\ Sombrero,\ Enchiladas,\ Tomatillo\text{-}\ Red\ Chili\},\ \{France,\ El\ Sombrero,\ Quesadillas,\ Tomatillo\text{-}Red\ Chili\},\ \{France, El\ Sombrero,\ Tacos,\ Tomatillo\text{-}Red\ Chili\},\ \{France,\ Old\ el\ Paso,\ Tacos,\ Tomatillo\text{-}Red\ Chili\}\}$.

Answers can be provided with an aggregated form which can be found in lattices, as we explain below. They allow us to discover sets of equivalent objects relatively to the answer. E.g. $\{Enchiladas, Quesadillas, Tacos\}$ are equivalent objects if we choose $France$ and $ElSombrero$.

Definition 7. *An aggregated answer to a query* $Q = (A_q, O_{vq}, R_q)$ *is the set AR containing the sets* S_i*, such that:*

- *there is a one-to-one mapping between AR and* A_q *which maps each* $q|_{K_i}$ *to a set* S_i
- $\forall\ q|_{K_i} \in A_q,\ \forall\ o_i \in S_i,\ o_i' \supseteq q|_{K_i}$ *(objects of* S_i *have the requested attributes)*
- *when* $(o_{vq|_{K_i}}, r, O_q) \in R_q$
  - *if* $o_{vq|_{K_j}} \in O_q,\ r \subseteq O_i \times O_j$, *thus* : $\forall\ o_i \in S_i,\ \forall\ o_j \in S_j,\ S_j \in AR$, *we have* $(o_i, o_j) \in r$ *(virtual objects are connected if requested)*
  - *for each* $o_j \in O_q \cap O_j$ *we have* : $(o_i, o_j) \in r$ *(connections with particular objects are satisfied).*

For example, an aggregated answer for our query is $\{S_{country}, S_{rest.}, S_{dish}, S_{salsa}\}$ $= \{\{France\}, \{ElSombrero\},\ \{Enchiladas, Quesadillas,\ Tacos\}, \{Tomatillo\text{-}RedChili\}\}$

4 Navigating a Concept Lattice Family w.r.t. a Query

In this section, we explain how the navigation between the concept lattices can be guided by a relational query. Following relational attributes that lead us from one lattice to another, we navigate a graph whose nodes are the concept lattices. In a first subsection, we propose an algorithm which gives a general navigation schema that applies to concept lattices built with the existential scaling. Then we present several variations of this navigation algorithm.

4.1 A query-based navigation algorithm

Our approach for navigating the concept lattices along the relational attributes is based on the observations made during an experimental use of RCA, for finding the appropriate Web services to implement an abstract calculation workflow [1]. We consider an RCF and a query that respects the RCF relations. From our experience, we observed that an expert often expresses his query by a phrase, where the chronology of the principal verbs (relations) gives a natural path for the query flow. This will be our hypothesis. Let us consider the query previously specified: *I am searching for a country, with the basic attribute "fr", that has a restaurant which serves dishes containing chicken, cheese and corn-tortilla; I am searching for a hot salsa suitable with this dish.* In order to simplify the notation, we use the same notation for queries $q|_{k_i}$ and the virtual objects $o_{vq|_{K_i}}$.

The query path is a total ordering of the arcs of the query (the query itself is a DAG in general). For our example, the path is the total ordering for R_q given by $\{(q_{country}, has, \{q_{restaurant}\}), (q_{restaurant}, serves, \{q_{dish}\}), (q_{dish}, contains, \{chicken, cheese, corn\text{-}tortilla\}), (q_{salsa}, suitable\text{-}with, \{q_{dish}\})\}$. Each arc corresponds to a relation used in the query. All the relations involved inside a query are covered by this path. This translation of the expert query determines a composition on the relations. The query path does not always correspond to a directed chain in the object graph (*e.g.* dishes are the end of two of the considered relations (serves and suitable-with)).

We propose the algorithms 1 to 3 (an additional procedure is needed which combines two others) for navigating through a concept lattice family using queries. During the exploration, we fill a set X by objects that will constitute an answer at the end (at most one object for each formal context). In this section, the algorithm is presented as an automatic procedure. Its use to guide an expert in its manual exploration of the data is discussed afterwards.

Algorithm 1 identifies three main cases:

- line 2, the arc connects two query objects, *e.g.* $(q_{country}, has, \{q_{restaurant}\})$;
- line 5, the arc connects a query object to original objects *e.g.* $(q_{dish}, contains, \{chicken, cheese, corn\text{-}tortilla\})$;
- line 8, the arc connects a query object to another query object and to original objects *e.g.* $(q_{dish}, contains, \{q_{ingredient}, chicken, cheese, corn\text{-}tortilla\})$.

Each of these cases considers, for a given arc a, whether the partial answer X already contains a source object or (inclusively) a target object.
When the arc connects a query object to another query object $a = (q|_{K_s}, r_{st}, q|_{K_t})$, (Algorithm 2), four cases are possible.

- X does not contain any object for K_s and any o_t for K_t: we identify the highest concept that introduces the attributes of $q|_{K_s}$ and we select an object in its extent (lines 3-5). Then the algorithm continues on the next conditional statement (to find a target).

- X contains an object o_s for K_s and an object o_t for K_t selected in previous steps: we just check if o_s owns the relational attribute pointing at the object concept introducing o_t, that is γo_t (line 8)[1].
- X contains only an object o_s for K_s. We should find a target. We identify, under the meet of the concepts that introduce the attributes of $q|_{K_t}$, one of the lowest concepts to which o_s points (lines 12-14). We select a target in its extent.
- X contains only an object o_t for K_t. We should find a source. We identify the meet of the concepts that introduce the attributes of $q|_{K_s}$ and the relational attribute that points to o_t (lines 20-23). We select a source in its extent.

When the arc connects a query object to original objects $a = (q|_{K_s}, r_{st}, O_q)$ (Algorithm 3):

- Either X contains an object for K_s and we need to check if the relational attributes confirm that this object is connected to all the original objects in O_q) (line 4);
- Or we have to select an object for K_s, owning the attributes of the query $q|_{K_s}$ and owning the relational attributes ending in the concepts introducing the original objects (line 9-11).

The algorithm for the last case is a combination of the algorithms for the two other cases. Note that whenever a condition is not verified, we have to backtrack, this is not specified in the algorithm for sake of simplicity. If the query path forms also a directed chain in the entity-relationship diagram, the main algorithm is a depth-first search. But in the general case, in some steps, when we consider an arc, we assigned to X an object for the end of the arc, and we need to find a source object.

For example, we start with the arc $(q_{country}, has, \{q_{restaurant}\})$ where the query path begins. We have to identify a source object o_s satisfying the query $\{fr\}$ (Definition 4). For example, we choose the object *France* appearing the extent of $Concept_4$, whose intent contains *fr*.

We extract the relational attributes of $o_s = France$, having the form $\exists r_{st}\ C$). They are in practice in the lattices denoted by $r : C$. For example, we obtain has:*Concept_19*, *has:Concept_15*, *has:Concept_60*, *has:Concept_16*, etc. We keep the relational attributes with the concepts satisfying the target query in the corresponding lattice and discard the rest. In our example, the $q_{restaurant}$ is empty. A relational attribute with the smallest concept (C_t) is the one to consider that leads us to find a solution. We choose *Concept_15* among the available smallest concepts. Let $\exists\ r_{st}\ C_t$ be the selected relational attribute (if it exists). The object o_t must be in the extent of C_t. In our example, we select *El Sombrero*.

Then we consider the query-to-query arc $(q_{restaurant}, serves, \{q_{dish}\})$. Given that an object is selected for $K_{restaurant}$, we look for a possible target object, led by the query $q_{dish} = \emptyset$ and the relational attributes owned by the object

[1] We remind that γo is the object concept introduced by o.

concept $Concept\_15$ which introduces *El Sombrero*. Suppose we choose (line 13) a relational attribute that targets one of the minimum concepts, namely $serves : Concept\_23$ (but $serves : Concept\_26$ or $serves : Concept\_25$ are also possible). This leads us to $Concept\_23$, in the extent of which we select *Enchiladas*.

Dealing with the next arc $(q_{dish}, contains, \{chicken, cheese, corn\text{-}tortilla\})$ involves, since we have already selected a dish, to verify (Algorithm 3, line 4) that, the object concept $\gamma Enchiladas$ owns all the relational attributes that go to object concepts introducing chicken, cheese, and corn-tortilla. These are $contains : \gamma\ chicken = Concept\_29$, $contains : \gamma\ cheese = Concept\_36$ and $contains : \gamma\ corn - tortilla = Concept\_40$ and they are indeed inherited by $\gamma Enchiladas = Concept\_23$.

When the arc $(q_{salsa}, suitable\text{-}with, \{q_{dish}\})$ is considered, the target (Enchiladas) is in X. Thus we identify a source in the extent of the $Concept\_47$, which satisfies the target query $\{hot\}$. Its intent contains $suitable - with : Concept\_23$ which is *Enchiladas*. A target object (Tomatillo-Red Chili) is selected in the extent of $Concept\_47$. The answer is now complete.

Algorithm 1: Navigate(RCF, Q, P_Q) $//P_Q = (a_k) \mid a_k = r_{ij}\ and\ r_{ij} \in R_Q$

```
Data: (𝕂, R): an RCF; Q = (A_q, O_vq, R_q): a query on (𝕂, R); and a query path
Result: X: an object set (answer for Q) or fail
foreach arc a ∈ P_Q do                                                    1
    if a = (q|_Ks, r_st, q|_Kt) then                                      2
        Case_pure_query                                                   3
    else                                                                  4
        if a = (q|_Ks, r_st, O_q) with O_q ⊆ O_t then                     5
            Case_pure_objects                                             6
        else                                                              7
            if a = (q|_Ks, r_st, q|_Kt) with q|_Kt ∈ O_q then             8
                Case_query_and_objects                                    9
```

4.2 Variations about the algorithm

Integrating queries into the contexts. One approach that was investigated in the case of simple queries consists of integrating the virtual query object in the context, then building the concept lattice. This can also be done for relational queries. A relational query $Q = (A_q, O_{vq}, R_q)$ can be integrated into an RCF by adding the virtual query objects $o_{v_{q|_{K_i}}}$ into the context K_i. Each virtual query object $o_{vq|_{K_i}}$ owns the attributes of the query $a_{q|_{K_i}}$ and for each arc $(o_{v_{q|_{K_i}}}, r_{ij}, o_{v_{q|_{K_j}}})$, the relational context of r_{ij} is enriched by a line for $o_{v_{q|_{K_i}}}$,

Algorithm 2: Case_pure_query

Let $a = (q|_{K_s}, r_{st}, q|_{K_t})$
if *// X does not contain a source and a target for the current arc a* $X \cap O_s = \emptyset$ and $X \cap O_t = \emptyset$ **then**
  *// **select** a source in the extent of a concept that verifies the source query*
  Let C_s be the highest concept having Intent $(C_s) \supseteq q|_{K_s}$
  select $o_s \in Extent(C_s)$
  $X \leftarrow X \cup \{o_s\}$
if *// X contains a source and a target for the current arc a* $X \cap O_s = \{o_s\}$ and $X \cap O_t = \{o_t\}$ **then**
  // verify that the source is connected to the target
  check $\exists r_{st}\ \gamma_{o_t} \in Intent(\gamma_{o_s})$
else
  if *// X contains a source for the current arc a* $X \cap O_s = \{o_s\}$ **then**
    // select a target in the extent of a concept that verifies the target query and is connected to the source
    Let C_t be the highest concept having Intent $(C_t) \supseteq q|_{K_t}$
    and $C_t \in \min(C \mid \exists\ (\exists r_{st}\ C) \in Intent(\gamma o_s))$
    select $o_t \in Extent(C_t)$
    $X \leftarrow X \cup \{o_t\}$
  else
    // X contains a target for the current arc a
    // select a source in the extent of a concept that verifies the source query and is connected to the target
    Let $o_t \in X \cap O_t$
    Let C_s be the highest concept having Intent $(C_s) \supseteq q|_{K_s}$
    and $\exists r_{st}\ \gamma o_t\ \in Intent(C_s)$
    select $o_s \in Extent(C_s)$
    $X \leftarrow X \cup \{o_s\}$

Algorithm 3: Case_pure_objects

Let $a = (q|_{K_s}, r_{st}, O_q)$ with $O_q \subseteq O_t$
if *// X contains a source for the current arc a* $X \cap O_s = \{o_s\}$ **then**
  // verify that the source is connected to the objects in O_q
  check $\forall o \in O_q, \exists r_{st}\ \gamma_o \in Intent(\gamma_{o_s})$
else
  // X does not contain a possible source
  // select a source in the extent of a concept that verifies the source query
  // and is connected to the target objects
  Let C_s be the highest concept having Intent $(C_s) \supseteq q|_{K_s}$
  and $\forall o \in O_q, \exists r_{st}\ \gamma_o \in Intent(C_s)$
  select $o_s \in Extent(C_s)$
  $X \leftarrow X \cup \{o_s\}$

a column for $o_{v_{q|_{K_j}}}$ and the relation $(o_{v_{q|_{K_i}}}, o_{v_{q|_{K_j}}})$[2]. We generate the corresponding concept lattice family, considering the existential scaling [3]. Locating the highest concept that introduces all the attributes of each query of each concerned context, now is much more easy because it introduces the virtual query object. Then, we can navigate in a similar way as before.

Opportunities of browsing offered by the exploration. As we explained before, the algorithm described in the previous section can be understood as an automatic procedure to determine a solution to a query. Nevertheless, it is more interesting to use it as a guiding method for the exploration of data by a human expert. Each object selection is a departure point for inspecting the objects of the selected concept, and, explore the neighborhood, going up by relaxing constraints or going down by adding constraints.

A point in favor of the lattices is that they do not only give us a solution, but they also classify the objects of the solutions and provide a navigation structure. They also carry other information about the objects which can be useful for the expert: attributes that objects of the answer set have necessarily, attributes that appear simultaneously as attributes of the answer, etc.

In our Web service application, we preferred the solution which integrates the query in RCF because it was easier to identify the answers. The lattices show how the existing objects match and differ from the query, thanks to the factorization of attributes between the query and the existing objects. Nevertheless, having several queries at the same time would not be efficient. Thus, the solution has been used only for specific problems. An incremental algorithm can be used to introduce the query, which enlightens the process of modifying the lattice and highlights the structure of the data. We can keep the original lattice (before query integration), and save the query objects together with the resulting concepts in an auxiliary structure. This way, we can always easily go back to the original lattices.

5 Related Work

ER-compatible data, e.g., relational databases, and concept lattices have a long history of collaboration. First attempts to apply FCA to that sort of data go back to the introduction of concept graphs by R. Wille in the mid-90s [8]. The standard approach is rooted in the translation of an ER model into a power-context family (PCF) where basically everything is represented within a formal context [9]. Thus, inter-object links of various arities (i.e., tuples of different sizes) are reified and hence become formal objects of a dedicated context (one per arity). The overall reasoning is therefore uniformly based on the formal concepts.

[2] See our example in Table `http://www.lirmm.fr/~huchard/RCA_queries/mexicoExistsWithQuery.rcft.html`)

[3] It is represented in Figure `http://www.lirmm.fr/~huchard/RCA_queries/mexicoExistsWithQuery.rcft.svg`

While this brings an undeniable mathematical strength in the formalization of the data processing and, in particular, querying, there are some issues with the expressiveness. Indeed, while formal concepts are typically based on a doubly universal quantification, the relational query languages mostly apply existential one.

Alternatives to the PCF in the interpretation of concept graphs have been proposed that involve the notions of nested graphs and cuts [2]. It was shown that the resulting formalism, called Nested Query Graphs, have the same expressive power over relational data as first order predicate logic and hence can be used as a visual representation of most mainstream SQL queries.

Existing approaches outside the concept graphs-based paradigm (see [3, 6]) follow a more conventional coding schema. Here inter-object links are modeled either through a particular sort of formal attributes or they reside in a different binary tables that match two sorts of individuals among them (instead of matching a set of individuals against a set of properties). Our own relational concept analysis framework is akin to this second category of approaches, hence our querying mechanisms are closer in spirit to those presented in the aforementioned papers.

For instance, in [3], the author proposes a language modeled w.r.t. SPARQL (the query language associated with the RDF language) to query relational data within the logical concept analysis (LCA) framework. The idea is to explore the relation structure of the data, starting from a single object and following its links to other objects. The language admits advanced constructs such as negation and disjunction and therefore qualifies as a fully-fledged relational query language.

Recently, a less expressive language has been proposed in [6] for the browsing of a relational database content while taking advantage of the underlying conceptual structure. As the author himself admits, the underlying data format used to ground the language semantics, the linked context family, is only slightly different from our own relational context family construct. The queries are limited here to conjunctions and existential quantifiers, yet variables are admitted. Consequently, query topologies are akin to general graphs: In actuality, the browsing engine comprises a factorization mechanism enabling the discovery of identical extensions in the query graph which are subsequently merged.

The downside of remaining free of the extensive commitments made by the concept graphs formalism both in terms of syntax and of semantics is the lack of unified methodological and mathematical framework beneath this second group of approaches. As a result, these diverge on a wide range of aspects which makes their in-depth comparison a hard task.

First, there is an obvious query language expressiveness gap: On that axis, the two extremes are occupied by the LCA- and the RCA-based approaches, respectively, the former being the most expressive and the latter, the less expressive one. Then, the role played by the concept lattices vs. the query resolution is specific in each case. While in the LCA-based approach the concepts seem to be formed on the fly, in [6] the author seems to imply that they are constructed beforehand. Despite this distinction, in both cases the concept lattice is a sec-

ondary structure that supports query resolution. In our own approach however, lattices are not only constructed prior to querying, but they also incorporate relational information in the intents of their concepts. In this sense, they are the primary structures whereas the queries are intended as navigational support.

6 Conclusion

In this paper, we have presented a query-based navigation approach that helps an expert to explore a concept lattice family. The approach was based on an application of Relational Concept Analysis to the selection of suitable Web services for instantiating an abstract service composition. There are many perspectives of this work. In our Web service experience, we tested other scaling operators (like the *covers* operator) that offers other results, and helps to find more easily the aggregate answers. The query language can be made more expressive (including quantifiers). For example, we can request dishes containing *only* {chicken, cheese, ...}, which means that the universal scaling operator shall be used in the RCA process for this particular relation. Besides, the query path can be calculated, rather than being defined by the expert, suggesting more efficient exploration paths.

References

1. Azmeh, Z., Driss, M., Hamoui, F., Huchard, M., Moha, N., Tibermacine, C.: Selection of composable web services driven by user requirements. In: ICWS. pp. 395–402. IEEE Computer Society (2011)
2. Dau, F., Correia, J.H.: Nested concept graphs: Applications for databases and mathematical foundations. In: Contribution to ICCS 2003. Skaker Verlag (2003)
3. Ferré, S.: Conceptual navigation in RDF graphs with SPARQL-Like Queries. In: Kwuida, L., Sertkaya, B. (eds.) ICFCA. LNCS, vol. 5986, pp. 193–208. Springer (2010)
4. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical Foundations. Springer-Verlag (1999)
5. Huchard, M., Rouane-Hacene, M., Roume, C., Valtchev, P.: Relational concept discovery in structured datasets. Ann. Math. Artif. Intell. 49(1-4), 39–76 (2007)
6. Kötters, J.: Object configuration browsing in relational databases. In: Valtchev, P., Jäschke, R. (eds.) ICFCA. Lecture Notes in Computer Science, vol. 6628, pp. 151–166. Springer (2011)
7. Messai, N., Devignes, M.D., Napoli, A., Smaïl-Tabbone, M.: Querying a bioinformatic data sources registry with concept lattices. In: Dau, F., Mugnier, M.L., Stumme, G. (eds.) ICCS. LNCS, vol. 3596, pp. 323–336. Springer (2005)
8. Wille, R.: Conceptual graphs and formal concept analysis. In: Lukose, D., Delugach, H.S., Keeler, M., Searle, L., Sowa, J.F. (eds.) ICCS. Lecture Notes in Computer Science, vol. 1257, pp. 290–303. Springer (1997)
9. Wille, R.: Formal concept analysis and contextual logic. In: Hitzler, P., Scharfe, H. (eds.) Conceptual Structures in Practice. pp. 137–173. Chapman and Hall/CRC (2009)

Links between modular decomposition of concept lattice and bimodular decomposition of a context

Alain Gély

LITA, Ile du Saulcy, 57045 Metz Cedex 1
Université de Metz, France
gely@univ-metz.fr

Abstract. This paper is a preliminary attempt to study how modular and bimodular decomposition, used in graph theory, can be used on contexts and concept lattices in formal concept analysis (FCA).
In a graph, a module is a set of vertices defined in term of behaviour with respect to the outside of the module: All vertices in the module act with no distinction and can be replaced by a unique vertex, which is a representation of the module. This definition may be applied to concepts of lattices, with slighty modifications (using order relation instead of adjacency).
One can note that modular decomposition is not well suited for bipartite graphs. For example, every bipartite graph corresponding to a clarified context is trivially prime (not decomposable w.r.t modules). In [4], authors have introduced a decomposition dedicaced to bipartite graph, called the *bimodular decomposition*. In this paper, we show how modular decomposition of lattices and bimodular decomposition of contexts interact. These results may be used to improve readability of a Hasse diagram.

1 Introduction

Concept lattices are well suited to deal with knowledge representation and classification, but when the number of concepts grows, it is not very convenient to visualize the Hasse diagram. To avoid this problem, some approaches keep only part of the concepts (Iceberg lattice [9] , usage of Galois sub-hierarchy [3], concepts with high stability score [7, 8] or any combinaison of these techniques); Others approaches try to obtain a more readable lattice by usage of a threeshold, as for α-galois lattices [10]. Another solution is to use decompositions to improve readability (see all chapter 4 of [5] and particularly nested diagrams).

There is a lot of works in graph theory about decomposition of a graph. A classical and well studied decomposition is the modular decomposition (see for example [6]). This decomposition has great properties: possibility of replacing a set of vertices by a single representant, so that visualization of the graph is better understandable; recursive approach, so that one can go from generalities to finer

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

detail levels (useful for knowledge representation); nice theoretical properties, as the existence of a decomposition tree or closure properties for the family of modules.

Modular decomposition of graphs may be adapted to lattices with only little changes: In a graph, this is *the adjacency relation* which is fundamental, but it is *the order relation* for lattices.

Moreover, concept lattices are usually computed from a context, which can be considerated as a bipartite graph. So, there are two structures which can be decomposed: the concept lattice and the bipartite graph.

Unfortunately, bipartite graphs are not good candidates for modular decomposition: except for twin vertices (vertices with the same neighbourghood) or connected components, there are no modules (except trivial one's) in such graphs. To improve the decomposition of bipartite graph, the notion of *bimodule* is introduced in [4].

Goal of this paper is to study how bimodules of a bipartite graph interact with modules of the concept lattice of this context, and to see how it can be used to help the visualisation of information contained in lattices.

The next section is dedicaced to definitions. Section 2.2 introduces modules of a graph and transposes the definition to lattice (modules of a lattice). Section 3 is about bimodules of a bipartite graph (context) and the links that exist with the corresponding concept lattice. After some discussion in section 4, we conclude the paper in section 5.

2 Preliminaries

2.1 Definitions

In this paper, all discrete structures are finites and all graphs are simples (no loops neither multi-edges). Since this paper is about usages of graph theory results, a formal context will be considerated as a bipartite graph $B = (O, A, I)$ with O (objects) and A (attributes) being two stable sets of vertices, and I (incidence relation between objects and attributes) the set of edges of B.

For a vertex v, v' denotes the neighbourghood of v (vertices adjacents to v). For a subset V of vertices, V' denotes the common neighbourhood (vertices which are adjacent to every vertices of V). With this notation, the classical definition of galois connections follows immediately.

Definition 1 (Galois connections). *For a set $X \subset O$, $Y \subseteq A$ we define*

$$X' = \{y \in O \mid xIy \text{ for all } x \in X\},$$
$$Y' = \{x \in A \mid xIy \text{ for all } y \in Y\}.$$

A clarified context is a context such that $x' = y'$ implie $x = y$ for any vertices of $O \cup A$. A clarified context is reduced if no vertex v is such that $v' = V'$ with $V \subseteq O \cup A$, $v \notin V$.

A complete lattice $L = (P, \leq, \vee, \wedge)$ is a poset such that for all $X \subseteq P$, there exist a supremum and an infimum in P. $j \in P$ is $\vee$-irreducible element if $x \vee y = j$ implies $x = j$ or $y = j$. $m \in P$ is a $\wedge$-irreducible element if $x \wedge y = m$ implies $x = m$ or $y = m$. j covers a unique element j_* ($j_* \prec j$), m is covered by a unique element m^* ($m \prec m^*$). We denote J the set of $\vee$-irreducible elements and M the set of $\wedge$-irreducible elements.

For a formal context $C = (O, A, I)$ a formal concept is a pair (X, Y), $X \subseteq O$, $X \subseteq A$ and $X' = Y$ and $Y' = X$. X is called the extent of the concept and Y is called the intent. The set of formal concepts ordered by inclusion on the intents is the concept lattice of C.

For every finite lattice $L = (P, \leq, \vee, \wedge)$ there is, up to isomorphism, a unique reduced context $C = (J, M, \leq)$. In the following of this paper, we will consider only reduced contexts, *i.e.* contexts such that $O = J$, the set of $\vee$-irreducible elements and $A = M$ the set of $\wedge$-irreducible elements of L.

2.2 Modules of graphs and lattices

We denote a graph G with $G = (V, E)$. V is the set of vertices and E a set of edges. Let $X \subset V$ and $s \in V \backslash X$. Then s *distinguishes* X if $s' \cap X \neq \emptyset$ and $s' \cap X \neq X$. That is, s is adjacent with some vertices of X and not adjacent with some others vertices of X. So, if no vertex distinguishes a set X, then for the outside of X and relation of adjacency, every vertex is similar and X can be viewed as a unique vertex.

Definition 2 (Module, graph theory). *A module in a graph is a subset of vertices that no vertex distinguishes.*

The graph which is obtained by the replacement of a module by a single vertex is called a quotient graph. It is a simplification of the original one (see Fig. 1). As no vertex distinguishes X (elements in dashed line), there exist only two possibilities for a vertex v not in X: either v is adjacent to every vertex of X (then there exists an edge between v and the representant of X) or v is adjacent with no vertex of X (then, there is no edge between v and the representant of X).

For a graph $G = (V, E)$, the set V and singletons $x \in V$ are trivial modules. A graph without non trivial module is called a prime graph (for the modular decomposition). Two modules A and B overlap if no one is a subset of the other and $A \cap B \neq \emptyset$. A module which does not overlap another module is a *strong module*.

Modules and strong modules are central in several decomposition processes and their properties have been well studied. In the first definitions, modules where defined with respect to the adjacency relation, but decompositions have been generalized (for example in [1]) for others properties of graphs.

For a lattice, it is more natural to consider the order relation than an adjacency relation, so a natural definition follows immediately:

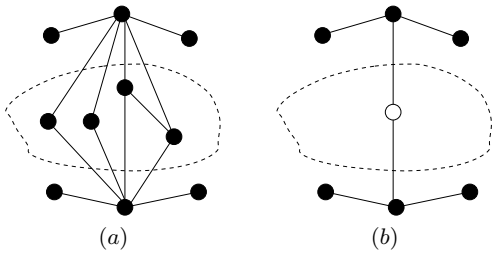


Fig. 1. (*a*) A module in a graph and (*b*) the quotient graph

Definition 3. *For a lattice $L = (P, \leq, \vee, \wedge)$, a lattice module is a set of elements $X \subseteq P$ such that, for every $y \in P \backslash X$, one of the three following statements is true:*

- $\forall x \in X,\ x < y$;
- *or* $\forall x \in X,\ x > y$;
- *or* $\forall x \in X,\ x || y$.

It is clear with this definition that a module in a lattice L is equivalent to a module (with respect to adjacency) in the graph obtained by transitive closure of the Hasse Diagram of L.

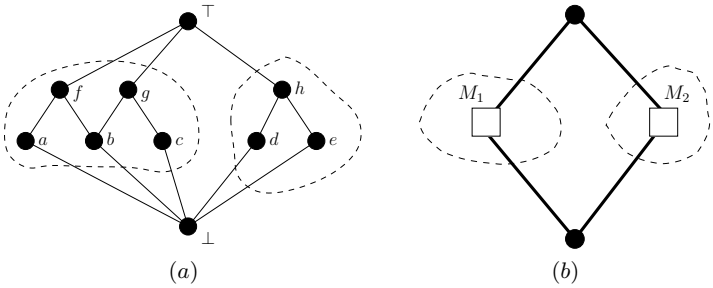


Fig. 2. Two strong modules of lattice (*a*) and the quotient lattice (*b*). Since no vertex outside the module distinguishes vertices inside the module, it can be collapsed to a single vertex which is the representant of the module. Note that M_2 can be recursively decomposed in two other modules $\{h\}$ (trivial) and $\{d, e\}$.

Let $X \subseteq P$ be a subset of elements of a lattice L, with $A = min(X)$ and $B = max(X)$ the sets of minimal (resp. maximal) elements of X. X is a convex set iff for all $y \in P$ such that $a < y < b$, $a \in A$, $b \in B$, then $y \in X$. If A and B are reduced to singletons, X is an interval. $[A, B]$ denotes the convex set defined by the two sets A and B.

Lemma 1. *Modules in lattices are convex sets.*

Proof. Suppose it is not, then there exists $y \in P \backslash X$ with $a < y < b$ and so, y distinguishes a and b. It follows that X is not a module.

From now, since lattices modules are convex sets, we will use the notation $X = [A, B]$ to speak of a module X.

Lemma 2. *For a lattice module* $[A, B]$*:*

1. *if* $|A| > 1$ *then* $A \subseteq J$,
2. *if* $|B| > 1$ *then* $B \subseteq M$.

Proof. Suppose $|A| > 1$, and let $A = a_1, a_2, \ldots, a_n$.
Suppose $a_i \notin J$, then, since $a_i || a_j$ there exists at least one $\vee$-irreducible element j such that $j < a_i$ and $j \not< a_j$, which is a contradiction with the fact that $[A, B]$ is a module.

Dually proof applies for elements of B.

Note that when $|A| = 1$ (dually for B) the maximal element of the module is not necessary an irreducible one (See Fig. 3).

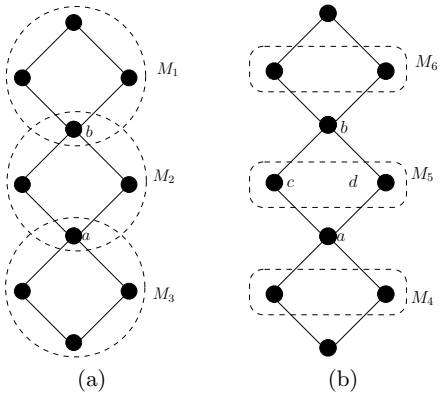


Fig. 3. (*a*) Module M_2 is a convex set $[A, B]$, with $A = \{a\} \not\subset J$ and $B = \{b\} \not\subset M$. M_1, M_2 and M_3 overlap: there are not strong modules. (*b*) M_4, M_5 and M_6 are strong modules but are not intervals. $M_5 = [A, B]$, with $A = \{b, c\} \subset J$ and $B = \{b, c\} \subset M$.

Lemma 3. *For a lattice module* $[A, B]$*:*

1. *if* $|A| > 2$, $\bigwedge A = a_i \wedge a_j$ *for all* $a_i, a_j \in A$.
2. *if* $|B| > 2$, $\bigvee B = b_i \vee b_j$ *for all* $b_i, b_j \in B$.

Proof. Clearly, suppose $|A| > 2$ and there exist $a_i, a_j, a_k \in A$ such that $x_1 = a_i \wedge a_j \neq a_j \wedge a_k = x_2$. w.l.o.g suppose $x_1 \not< x_2$. Then $x_1 < a_j$ and $x_1 \not< a_k$. It follows that x_1 distinguishes $[A, B]$.

3 Modules of lattices and bimodules of contexts

As a preliminary remark, we recall that all considered contexts are reduced, and so, clarified. The clarification of a context is the fact to keep only one object o for all objects o_i such that $o'_i = o'_j$ (dually for attribute). It is clear that the set $\{o_1, \ldots, o_n\}$ is a module in the bipartite graph and this process is equivalent to replace twin vertices by a representant.

Modules are not well situed for bipartite graphs. Twin vertices and connected components are the only modules for these graphs which are poorly decomposable. In goal to improve the decomposition, Fouquet and *all* have introduced *bimodule*, an analog of module for bipartite graphs.

Definition 4 (Bimodule). *Let $C = (O, A, I)$ be a bipartite graph, and $(X, Y) \subset (O, A)$, then (X, Y) is a bimodule if no $x \in O \backslash X$ distinguishes A and no $y \in A \backslash Y$ distinguishes O.*

Example of bimodule is given in Fig. 4: b and c are not distinguished with respect to vertices 4 (none of them are adjacent) or 3 (each of them is adjacent). Similarly, 1 and 2 are not distinguished by a (each of them is adjacent) and d (none of them is adjacent).

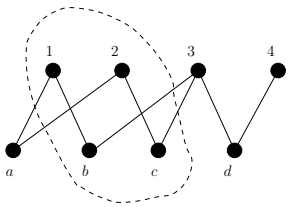


Fig. 4. Example of bimodules

The whole bipartite (O, A), all vertices and pairs (j, m), $j \in J$, $m \in M$ are trivial modules. In the following, we consider only non trivial bimodules, *i.e* bimodules with at least 3 elements.

Proposition 1. *To any non trivial module X of lattice corresponds a bimodule of reduced context.*

Proof. By definition of a lattice module, no elements inside the modules are distinguished by elements outside. It follows directly that no $\vee$-irreducible element outside the module distinguishes $\wedge$-irreducible elements inside, and conversely.

Now, we want to know how a bimodule on the context may be interpretated in the concept lattice. First, we define a set in the lattice L from a bimodule X.

From a bimodule $X = (J_1, M_1) \subseteq (J, M)$, we build a subset C of concepts in L such that:

- attributes concepts of X are in C,
- objects concepts of X are in C,
- $C = [A, B]$ is a convex set, with A being maximal elements of C and B being minimal elements of C.

As previously seen, a lattice module corresponds to a context bimodule but, with the previous construction, the converse may be false: There exist bimodules of a context such that $[A, B]$ does not correspond to a module in the lattice. As an example, Fig. 5 shows the lattice for the bipartite graph in Fig. 4. The set $[A, B]$ is bounded by a dashed line.

Nevertheless, we can observe that, even if $[A, B]$ is not a module, there exists a possibility of simplification, replacing a set of elements by two vertices and an edge.

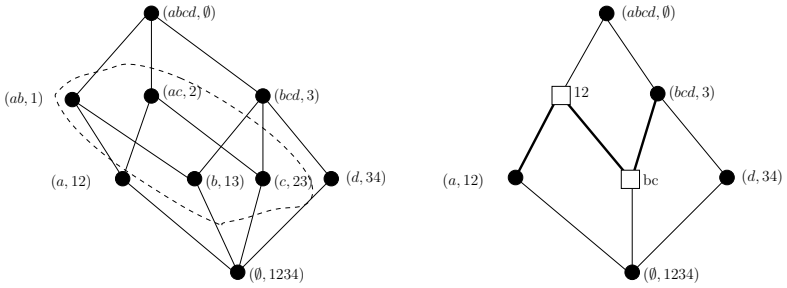


Fig. 5. (a) lattice for bipartite graph in Fig. 4.a and (b) simplified lattice

In the following, we show that when $[A, B]$ is not a module, the shape of the set $[A, B]$ is very constrained.

Proposition 2. *Let $[A, B]$ be a convex set built from a non trivial bimodule X. If $[A, B]$ is not a module, then*

1. $|\ |[A, B] \cap M| - |[A, B] \cap J|\ | \leq 1$
2. *if* $|[A, B] \cap M| = |[A, B] \cap J|$, *then* $A \subseteq J$ *and* $B \subseteq M$
3. *if* $|max([A, B] \cap J)| > 1$, $a_+ = \bigvee a_i$, $a_i \in max([A, B] \cap J)$ *is such that* $a_i \prec a_+$
4. *if* $|min([A, B] \cap M)| > 1$, $b_- = \bigwedge b_i$, $b_i \in min([A, B] \cap M)$ *is such that* $b_- \prec b_i$

Proof. First, we show that $|\ |[A, B] \cap M| - |[A, B] \cap J|\ | \leq 1$:

Suppose that $[A, B]$ is not a module of the concept lattice, then there exists an element $x \notin [A, B]$ which distinguishes $[A, B]$. Without lost of generality, we can consider that $x \in J$ and there exist $y \in [A, B]$ such that $x < y$. We denote P_1

the set of elements of $[A, B]$ which are greater than x and $P_2 = [A, B] \backslash P_1$. Since x is a $\vee$-irreducible element, it does not distinguish any $\wedge$-irreducible elements in $[A, B]$. It follows that all $\wedge$-irreducible elements in $[A, B]$ are in P_1 and none of them in P_2 (or conversely).

In a finite lattice, every element e is $\wedge$-dense, *i.e.* equal to the infimum of $\wedge$-irreducible elements greater than e.

All $\vee$-irreducible elements in P_2 cannot be distinguished by $\wedge$-irreducible elements outside $[A, B]$. One unique $\vee$-irreducible element j_{max} of P_2 may be defined by $j_{max} = \bigwedge m_i, \ldots, m_j$, with $m_i, \ldots, m_j \notin P_1$. All other $\vee$-irreducible elements in P_2 are distinguished by $\wedge$-irreducible elements in P_1 (and only by these elements). So $P_2 \cap J = X \cup \{j_{max}\}$ (j_{max} may not exist).

Suppose $|X| < |[A, B] \cap M|$, then there exist $m_1, m_2 \in [A, B] \cap M$, $j_1 \in X$ such that $j_1 < m_1$, $j_1 < m_2$, $m_1 || m_2$. It follows that $j_1 < m_1 \wedge m_2$, which is impossible since $j_1 \in P_2$ and elements in P_2 are not comparable to x.

Similarly, suppose $|X| > |[A, B] \cap M|$, At least one $\vee$-irreducible element j of X is smaller than two $\wedge$-irreducible elements m_1 and m_2 of P_1, with $m_1 || m_2$. This is impossible, so $|X| = |[A, B] \cap M|$ and $| \, |[A, B] \cap M| - |[A, B] \cap J| \, | \leq 1$.

$A \subseteq J$ and $B \subseteq M$ follow directly of the fact that, by construction A and B contain irreducible elements and for each $\wedge$-irreducible element $m \in [A, B]$, there exists a $\vee$-irreducible element $j \in [A, B]$ such that $j < m$.

It remains to prove that, when $max([A, B] \cap J)$ contains at least two elements, $a_+ = \bigvee a_i$, $a_i \in max([A, B] \cap J)$ is such that $a_i \prec a_+$ (and dually for b_-). Suppose it is not the case, then exist at least two elements x_1 and x_2 smaller than a_+ and such that x_1 and x_2 distinguish elements in A. It follows that one can find a $\wedge$-irreducible element which distinguishes $\vee$-irreducible elements in A and that is a contradiction.

It follows from this proposition that even if a set $[A, B]$ is not a module, it can be collapsed into two vertices j and m such that $j < m$ (but maybe not $j \prec m$). j is a representant for the set $[A, B] \cap J$ and m a representant for the set $[A, B] \cap M$. Moreover, $j \prec a_+$ and $b_+ \prec m$.

4 Discussion

4.1 Algorithmic Aspects

It is known that the family of modules of a graph (and so, of a lattice) and the family of bimodules of a bipartite graph are closed by intersection. Since the whole graph is a (trivial) module, it defines a lattice. So, for any set S of vertices, it is possible to use a closure operator to compute the smallest module which contains S. Algorithm 1 adds all vertices which distinguish respectively X and Y and the same process is repeated until no more vertex can be added.

Usually, bimodules decomposition does not produce all possible modules, but an inclusion tree such that all possible bimodules can be deduced from this tree. The root represents the whole graph and the leaves are vertices (trivial

```
Input: (O, A, I) a bipartite graph, (X, Y) ⊂ (O, A)
Output: (X_c, Y_c), smallest bimodule containing (X, Y)
begin
    continue ← true;
    (X_c, Y_c) ← (X, Y);
    while continue do
        continue ← false;
        forall the x ∈ J\X_c do
            if x distinguishes Y_c then
                X_c ← X_c ∪ x;
                continue ← true;
            end
        end
        forall the y ∈ M\Y_c do
            if y distinguishes X_c then
                Y_c ← Y_c ∪ y;
                continue ← true;
            end
        end
    end
    return (X_c, Y_c)
end
```

Algorithm 1: Computation of the smallest bimodule which contains (X, Y)

bimodules). It follows that the size of the tree is $O(n)$, with $n = |O| + |A|$. In [1], authors propose a $O(n^3)$ algorithm to compute a such tree.

4.2 Decomposition and Real Data

In Fig. 6, an example of bimodule is shown on the "Living Beings and Water" concept lattice [5]. g is the attribute for "can move around" and h is the one for "has limbs". These two attributes are equivalent (cannot be distinguished) from the outside of the bimodule. So, on the lattice in Fig. 6.c these two attributes are collapsed, as well as objects 1 (Leech) and 2 (Bream). Further work must be done on real data to see what bimodules can enlight for practical cases.

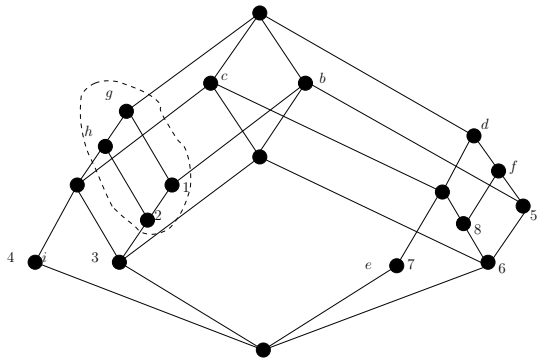
5 Conclusion

First, we have seen that modules defined on a lattice have natural links with bimodules of the bipartite graph (context) of this lattice. Modules of a lattice can be used the same way as modules of a graph are used: to produce a quotient lattice, which is a simplification of the original one. Recursive definition of modules allows to consider several details levels in the lattice.

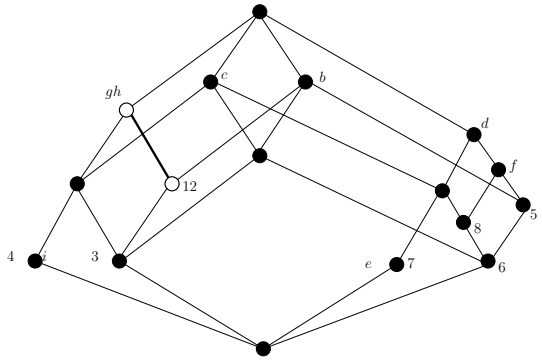
All results in modular decomposition may be transposed immediatly to concept lattice and associated context to improve the readibility of the lattice.

	b	c	d	e	f	g	h	i
1	×					×		
2	×					×	×	
3	×	×				×	×	
4		×				×	×	×
5	×		×		×			
6	×	×	×		×			
7		×	×	×				
8		×	×		×			

(a)



(b)



(c)

Fig. 6. (a) "Living Beings and Water " Context [5], (b) Concept lattice for "living Beings and Water" and (c) the same concept lattice with a bimodule collapsed.

Second, investigation of bimodules properties shows that a bimodule may not correspond to a module of the lattice. Nevertheless, it remains possible to use it to produce a simplification of the original lattice. In such a case, the bimodule is collapsed in two elements a and b which represent $\vee$-irreducible elements and $\wedge$-irreducible elements of the bimodule.

This last case is a particular case of another decomposition proposed for inheritance hierarchies [2], called the block decomposition (with a different definition of block that the one in [5]): a block is an interval $[a, b]$ such that only a and b can be distinguished of other vertices from the outside of the block. As a perspective behaviour of this decomposition for lattices and associated properties on the context can be investigated.

References

1. m. Bui Xuan, B., Habib, M., Limouzy, V., Montgolfier, F.D.: Homogeneity vs. adjacency: generalising some graph decomposition algorithms. In: In 32nd International Workshop on Graph-Theoretic Concepts in Computer Science (WG), volume 4271 of LNCS (2006)
2. Capelle, C.: Block decomposition of inheritance hierarchies. In: WG. pp. 118–131 (1997)
3. Encheva, S.: Galois sub-hierarchy and orderings. In: Proceedings of the 10th WSEAS international conference on Artificial intelligence, knowledge engineering and data bases. pp. 168–171. AIKED'11, World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA (2011), http://portal.acm.org/citation.cfm?id=1959485.1959517
4. Fouquet, J., Habib, M., de Montgolfier, F., Vanherpe, J.: Bimodular decomposition of bipartite graphs. In: Graph-Theoretic Concepts in Computer Science 30th International Workshop, WG 2004, Bad Honnef, Germany, June 21-23 (2004)
5. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical Foundations. Springer-Verlag Berlin (1996)
6. Habib, M., Paul, C.: A survey of the algorithmic aspects of modular decomposition. Computer Science Review 4, 41–59 (2010)
7. Jay, N., Kohler, F., Napoli, A.: Analysis of social communities with iceberg and stability-based concept lattices. In: Proceedings of the 6th international conference on Formal concept analysis. pp. 258–272. ICFCA'08, Springer-Verlag, Berlin, Heidelberg (2008), http://portal.acm.org/citation.cfm?id=1787746.1787765
8. Klimushkin, M., Obiedkov, S., Roth, C.: Approaches to the selection of relevant concepts in the case of noisy data. In: Kwuida, L., Sertkaya, B. (eds.) Proc. 8th Intl. Conf. Formal Concept Analysis. LNCS/LNAI, vol. 5986, pp. 255–266. Springer (2010)
9. Stumme, G., Taouil, R., Bastide, Y., Lakhal, L.: Conceptual clustering with iceberg concept lattices. In: In: Proc. of GI-Fachgruppentreffen Maschinelles Lernen'01, Universität Dortmund (2001)
10. Ventos, V., Soldano, H.: Alpha galois lattices: an overview. In: In: International Conference in Formal Concept Analysis (ICFCA05), LNCS. pp. 298–313. Springer (2005)

Abduction in Description Logics using Formal Concept Analysis and Mathematical Morphology: Application to Image Interpretation

Jamal Atif[1], Céline Hudelot[2], and Isabelle Bloch[3]

1. Université Paris Sud, LRI - TAO, Orsay, France jamal.atif@lri.fr
2. Ecole Centrale de Paris, France, celine.hudelot@ecp.fr
3. Telecom ParisTech - CNRS LTCI, Paris, France
isabelle.bloch@telecom-paristech.fr

Abstract. We propose an original way of enriching Description Logics with abduction reasoning services by computing the best explanations of an observation through mathematical morphology (using erosions) over the Concept Lattice of a background theory. The intended application is scene understanding and spatial reasoning.
Keywords: Abduction, Description Logics, FCA, Mathematical Morphology, Scene Understanding.

1 Introduction and notations

Scene interpretation can benefit from prior knowledge expressed as ontologies and from description logics (DL) endowed with spatial reasoning tools as illustrated in our previous work [5, 6]. The challenge in this work was to derive reasoning tools that are able to handle in a unified way quantitative information supplied by the image domain and qualitative pieces of knowledge supplied by the ontology level. Object recognition and interpretation are seen as the satisfiability of a current situation (spatial configuration) encoded in the ABox of the DL and its TBox part. However, when the expert knowledge is not crisply consistent with the observations, which is common in image interpretation, then this approach does not apply or leads to inconsistent results. Adapting DL reasoning tools to such situations can be performed using abduction. Our aim is thus to compute the "best explanation" to the observed phenomena in such situations. Formally, given a background theory $\mathcal{K}$ representing the expert knowledge and a formula C representing an observation on the problem domain, abductive reasoning searches for an explanation formula D such that D is satisfiable w.r.t. $\mathcal{K}$ and it holds that $\mathcal{K} \models D \rightarrow C$ ($\mathcal{K} \cup D \models C$). We propose to add abductive reasoning tools to DL by associating ingredients from mathematical morphology, DL and Formal Concept Analysis (FCA), and by computing the best explanations of an observation through algebraic erosion over the concept lattice of a background theory which is efficiently constructed using tools from FCA. We show that the defined operators satisfy important rationality postulates of abductive reasoning.

Based on the TBox $\mathcal{T}$ and the ABox $\mathcal{A}$ parts of a knowledge base $\mathcal{K}$, we consider ABox abduction [3]: if for every $a \in A$ it holds that $\mathcal{K} \not\models \neg a$, an ABox Abduction Problem, denoted as $\langle \mathcal{K}, A \rangle$, consists in finding a set of assertions γ such that $\mathcal{K} \cup \gamma \models A$.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

The set γ (consistent with $\mathcal{K}$) is said to be an explanation of A. Explanatory reasoning is concerned with preferred explanations rather than just plain explanations. So, explaining an observation requires that some formulas must be "selected" as preferred explanations.

We also rely on classical notions of (FCA), and denote a formal context by $\mathbb{K} = (G, M, I)$, where G is the set of objects, M the set of attributes and $I \subseteq G \times M$ a relation between the objects and attributes. For $X \subseteq G$ and $Y \subseteq M$, the derivation operators are denoted by α and β, with $\alpha(X) = \{m \in M \mid \forall g \in X, (g, m) \in I\}$, and $\beta(Y) = \{g \in G \mid \forall m \in Y, (g, m) \in I\}$. The concept lattice is defined from the classical partial ordering $(X_1, Y_1) \leq (X_2, Y_2) \Leftrightarrow X_1 \subseteq X_2$ ($\Leftrightarrow Y_2 \subseteq Y_1$).

Links between FCA and DL can be formalized via the notion of semantic context $\mathbb{K}_{\mathcal{T}} := (G, M, I)$ defined as [1]: $G := \{(\mathcal{I}, d) \mid \mathcal{I}$ is a model of $\mathcal{T}$ and $d \in \Delta^{\mathcal{I}}\}$, $M := \{m_1, \ldots, m_n\}$, and $I := \{((\mathcal{I}, d), m) \mid d \in m^{\mathcal{I}}\}$, where $\mathcal{I} = (\Delta^{\mathcal{I}}, .^{\mathcal{I}})$ denotes an interpretation. The lattice can be constructed using the distributive concept exploration algorithm [9].

2 Abduction Operators from Mathematical Morphology on Complete Lattices

Let $(L, \preceq)$ and $(L', \preceq')$ be two complete lattices (which do not need to be equal). An operator $\delta : L \rightarrow L'$ is a dilation if it commutes with the supremum. An operator $\varepsilon : L' \rightarrow L$ is an erosion if it commutes with the infimum. Classical properties of mathematical morphology operators on complete lattices can be found in [4, 8].

Here, with the aim of performing ABox abduction, we would like to reason on subsets of G in order to find their best explanations (in G). Hence we consider the complete lattice $(\mathcal{P}(G), \subseteq)$ and operations from $\mathcal{P}(G)$ into $\mathcal{P}(G)$, where $\mathcal{P}(G)$ is the set of subsets of G. Since the ordering on G is equivalent to the one of M, reasoning on G will directly lead to results on M. In order to define explicit operations on $\mathcal{P}(G)$, we will make use of particular erosions and dilations, called morphological ones [8], which involve the notion of structuring element, i.e. a binary relation b between elements of G. For $g \in G$, we denote by $b(g)$ the set of elements of G in relation with g. It can be typically derived from a distance d: $b(g) = \{g' \in G \mid \exists X \in \mathcal{P}(G), g' \in X, d(\{g\}, X) \leq 1\}$. The morphological erosion of X is then expressed as $\varepsilon_b(X) = \{g \in G \mid b(g) \subseteq X\}$. Defining b from a distance is particularly interesting in the context of abduction, where the "most central" parts of models will have to be defined. Erosion is then expressed as $\varepsilon^n(X) = \{g \in G \mid d(g, X^C) > n\}$, where X^C denotes the complement of X in G. Here G is a discrete finite space, and therefore only integer values of n are considered. All classical properties of mathematical morphology hold in this framework.

Last Non-empty Erosion. As shown in [2] in the framework of propositional logic, erosions can be used to find explanations. In this context, the idea was to find the *most central part* of a formula as the best explanation. This approach was shown to have good properties with respect to rationality postulates of abductive reasoning [7]. In this paper, we propose similar ideas, but adapted to the context of concept lattices, using erosions as defined above. For any $X \subseteq G$, we define its last erosion as $\varepsilon_\ell(X) = \varepsilon^n(X) \Leftrightarrow$

$\varepsilon^n(X) \neq \emptyset$, and $\forall m > n,\ \varepsilon^m(X) = \emptyset$. This last non-empty erosion defines the subset of models in G that are the furthest ones from the complement of X (according to the distance d), i.e. the most central in X.

Definition 1 *Let A be a set of ABox assertions. A preferred explanation γ of A is defined from the last non-empty erosion as $A \rhd^{\ell ne} \gamma \stackrel{def}{\Leftrightarrow} \gamma^{\mathcal{I}} \subseteq \varepsilon_\ell(A^{\mathcal{I}})$. In this equation, $A^{\mathcal{I}}$ should be understood as the extent of the semantic concept associated with the DL concept A. When a constraint (e.g. a set of hypotheses belonging to the backgroud theory) $\mathcal{H}$ has to be introduced, then this definition is modified as $A \rhd^{\ell ne} \gamma \stackrel{def}{\Leftrightarrow} \gamma^{\mathcal{I}} \subseteq \varepsilon_\ell(\mathcal{H}^{\mathcal{I}} \cap A^{\mathcal{I}})$.*

Starting from the subset to be explained, performing successive erosions amounts to "go down" in the lattice as much as possible, in order to find a non-empty set of interpretations.

Last Consistent Erosion. Another idea to introduce the constraint $\mathcal{H}$ is to erode it, as soon as it remains consistent with A. This leads to a second explanatory relation.

Definition 2 *A preferred explanation γ of A is defined from the last consistent erosion as: $A \rhd^{\ell c} \gamma \stackrel{def}{\Leftrightarrow} \gamma^{\mathcal{I}} \subseteq \varepsilon_{\ell c}(\mathcal{H}^{\mathcal{I}}, A^{\mathcal{I}}) \cap A^{\mathcal{I}}$, where $A^{\mathcal{I}}$ corresponds to the extent of the semantic context and $\varepsilon_{\ell c}$ is the last consistent erosion defined as $\varepsilon_{\ell c}(\mathcal{H}^{\mathcal{I}}, A^{\mathcal{I}}) = \varepsilon^n(\mathcal{H}^{\mathcal{I}})$ where $n = \max\{k \mid \varepsilon^k(\mathcal{H}^{\mathcal{I}}) \cap A^{\mathcal{I}} \neq \emptyset\}$.*

Here we consider erosion of $\mathcal{H}$ (i.e. $\mathcal{H}^{\mathcal{I}}$) alone, which means that we are looking at the subsets (submodels) of the models of A while being the most in the constraint.

Properties and interpretations. A first important property is that reasoning on G actually amounts to reason on the whole formal context. Here, explanations where defined from ABox reasoning, leading to erosions of subsets of G (models). Let (X, Y) be a formal concept, with $X \subseteq G$ and $Y \subseteq M$. From the definitions of explanations of X, we can derive directly the corresponding concepts for Y, using the derivation operator, i.e. $\alpha(\gamma) = \{m \in M \mid \forall g \in \gamma, (g, m) \in I\}$. Note that eroding X amounts to dilate Y, which is in accordance with the correspondence between the Galois connection property between derivation operators and the adjunction properties of dilation and erosion. Let us now consider the rationality postulates introduced in [7] for explanation relations. It has been proved that most of them hold for explanations derived from last non-empty erosion and from last consistent erosion [2]. These results extend to the DL context as follows:

- Both $\rhd^{\ell ne}$ and $\rhd^{\ell c}$ are independent of the syntax (since they are computed on models).
- Definitions are consistent in the sense that $\mathcal{K} \not\models \neg A$ iff $\exists \gamma, A \rhd \gamma$.
- A reflexivity property holds for both definitions: if $A \rhd \gamma$, then $\gamma \rhd \gamma$.
- Disjunctions of explanations: if $A \rhd \gamma$ and $A \rhd \delta$, then $A \rhd (\gamma \sqcup \delta)$, for both definitions. This means that if there are several possible explanations, their disjunction is an explanation as well, which is an expected result.

- Disjunction on the left: if $C \rhd^{\ell c} \gamma$ and $D \rhd^{\ell c} \gamma$, then $(C \sqcup D) \rhd^{\ell c} \gamma$ (since the erosion is always performed on $\mathcal{H}^{\mathcal{I}}$). However this property does not hold for $\rhd^{\ell ne}$ since erosion does not commute with the supremum.
- For the same reasons, we have the following property for $\rhd^{\ell c}$: if $C \rhd^{\ell c} \gamma$ and $D \rhd^{\ell c} \delta$, then $(C \sqcup D) \rhd^{\ell c} \gamma$ or $(C \sqcup D) \rhd^{\ell c} \delta$, but it does not hold for $\rhd^{\ell ne}$.
- For conjunctions, we have a monotony property for $\rhd^{\ell c}$: if $C \rhd^{\ell c} \gamma$ and $\gamma^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ (i.e. $D \models \gamma$), then $(C \sqcap D) \rhd^{\ell c} \gamma$. For $\rhd^{\ell ne}$, only a weaker form holds: if $C \rhd^{\ell ne} \gamma$ and $D \rhd^{\ell ne} \gamma$, then $(C \sqcap D) \rhd^{\ell ne} \gamma$. Note that this weaker form is also very natural and interesting.

Since both $\rhd^{\ell ne}$ and $\rhd^{\ell c}$ operators perform erosion in the interpretation set $\Delta^{\mathcal{I}}$, any solution belongs then to this set and $\mathcal{K}$ is a model of the obtained solution. Hence we have the following theorems:

- Soundness: If $\exists \gamma \mid A \rhd \gamma$ then $\mathcal{K} \models \gamma$.
- Completeness: $\mathcal{K} \models \gamma \Rightarrow \exists A \mid \mathcal{K} \models A : A \rhd \gamma$.

3 Conclusion

With the aim of image interpretation, we have proposed abductive inference services in DL based on mathematical morphology over concept lattices, whose construction is based on exploiting the advances of using FCA in DL. The properties and interpretations of the introduced explanatory operators were analyzed, and the rational postulates of abductive reasoning were stated and extended to our context. Future work will concern the complexity analysis of these operators and associated algorithms, and a deeper investigation of their applications to image interpretation.

References

1. F. Baader. Computing a minimal representation of the subsumption lattice of all conjunctions of concepts defined in a terminology. In *Knowledge Retrieval, Use and Storage for Efficiency: 1st International KRUSE Symposium*, pages 168–178, 1995.
2. I. Bloch, R. Pino-Pérez, and C. Uzcátegui. Explanatory Relations based on Mathematical Morphology. In *ECSQARU 2001*, pages 736–747, Toulouse, France, sep 2001.
3. C. Elsenbroich, O. Kutz, and U. Sattler. A case for abductive reasoning over ontologies. In *OWL: Experiences and Directions*, Athens, Georgia, USA, 2006.
4. H. J. A. M. Heijmans and C. Ronse. The Algebraic Basis of Mathematical Morphology – Part I: Dilations and Erosions. *Computer Vision, Graphics and Image Processing*, 50:245–295, 1990.
5. C. Hudelot, J. Atif, and I. Bloch. Fuzzy spatial relation ontology for image interpretation. *Fuzzy Sets and Systems*, 159(15):1929–1951, 2008.
6. C. Hudelot, J. Atif, and I. Bloch. Integrating bipolar fuzzy mathematical morphology in description logics for spatial reasoning. In *European Conference on Artificial Intelligence ECAI 2010*, pages 497–502, Lisbon, Portugal, August 2010.
7. R. Pino-Pérez and C. Uzcátegui. Jumping to Explanations versus jumping to Conclusions. *Artificial Intelligence*, 111:131–169, 1999.
8. J. Serra. *Image Analysis and Mathematical Morphology*. Academic Press, New-York, 1982.
9. G. Stumme. Distributive concept exploration–a knowledge acquisition tool in formal concept analysis. In *KI-98: Advances in Artificial Intelligence*, pages 117–128. Springer, 1998.

A local discretization of continuous data for lattices: *Technical aspects*

Nathalie Girard, Karell Bertet and Muriel Visani

Laboratory L3i - University of La Rochelle - FRANCE
ngirar02, kbertet, mvisani@univ-lr.fr

Abstract. Since few years, Galois lattices (GLs) are used in data mining and defining a GL from complex data (*i.e.* non binary) is a recent challenge [1,2]. Indeed GL is classically defined from a binary table (called context), and therefore in the presence of continuous data a *discretization* step is generally needed to convert continuous data into discrete data. Discretization is classically performed before the GL construction in a **global** way. However, **local** discretization is reported to give better classification rates than global discretization when used jointly with other symbolic classification methods such as decision trees (DTs). Using a result of lattice theory bringing together set of objects and specific nodes of the lattice, we identify subsets of data to perform a **local discretization for GLs**. Experiments are performed to assess the efficiency and the effectiveness of the proposed algorithm compared to global discretization.

1 Discretization process

The discretization process consists in converting continuous attributes into discrete attributes [3]. This conversion can induce scaling attributes or **disjoint intervals**. We focus on the latter. Such a transformation is necessary for some classification models like symbolic models, which cannot handle continuous attributes [4]. Consider a continuous data set $D = (O, F)$, where each object in O is described by p continuous attributes in F. The discretization process is performed by iteration of attribute splitting step, according to a **splitting criterion** (Entropy [3], Gini [5], χ^2 [6], ...) until a **stopping criterion** S is satisfied (a maximal number of intervals to create, a purity measure,...).
More formally for one discretization step, for selecting the best attribute to be cut, let $(v_1, \ldots, v_N)$ be the sorted values of a continuous attribute $V \in F$. Each v_i corresponds to a value verified by one object of the data set D. The set of possible cut-points is $C_V = (c_V^1, \ldots, c_V^{N-1})$ where $c_V^i = \frac{v_i + v_{i+1}}{2}$ $\forall i \leq N-1$.
The best cut-point, denoted c_V^*, is defined by:

$$c_V^* = argmax_{c_V^i \in C_V}(gain(V, c_V^i, D)) \quad (1)$$

where $gain(V, c, D)$ denotes in a **generic** manner the **splitting criterion** computed for the attribute V, the cut-point $c \in C_V$ and the data set D.
The best attribute, denoted V^*, is the $V \in F$ maximizing the **splitting criterion** computed for its best cut-point (*i.e.* c_V^*):

$$V^*(D) = argmax_{V \in F}(gain(V, c_V^*, D)) \quad (2)$$

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

Finally for one discretization step, the attribute V^* is divided into two intervals: $[v_1, c^*_{V^*}]$ and $]c^*_{V^*}, v_n]$ and the process is repeated.
This process can be run using, at each step, all the objects in the training set. This is **global discretization**. It can also be run during model construction considering, at each step, only a part of the training set. This is **local discretization**. In [7], Quinlan shows that **local discretization improves supervised classification** with decision trees (DTs) as compared with global discretization. In DT construction, the growing process is iterated until S is satisfied. Local discretization is performed on the subset of objects in the current node to select its best attribute ($V^*(node)$), according to the splitting criterion. Given the structural links between DTs and Galois lattices (GLs) [8], we propose a local discretization algorithm for GL and compare its performances with a global discretization.

2 Local discretization for Galois lattices

A GL is generally defined from a binary relation R between objects O and binary attributes I - *i.e.* a binary data set also called a **formal context** - denoted as a triplet $T = (O, I, R)$. A GL is composed of a set of **concepts** - a concept (A, B) is a maximal objects-attributes subset in relation - ordered by a generalization/specialization relation. For more details on GL theory, notation and their use in classification tasks, please refer to [9,10]. To define a local discretization for GL, we have to identify at each discretization step the subset of concepts to be processed. Given a subset of objects $A \in P(O)$, there always exists a smallest concept M containing this subset and identified in lattice theory as a **meet-irreducible concept** of the GL [11]. Moreover, it is possible to compute the set of meet-irreducibles directly from the context, thus the generation of the lattice is useless [12]. Consequently, local discretization is performed on the set of meet-irreducible concepts MI which does not satisfy S. Attributes in MI are locally discretized: the best attribute $V^*(M)$ for each $M \in MI$ is computed according to eq. (3); then the best one $V^*(MI)$ (eq. (4),(5)) for the whole set MI is split into two intervals as explain before. The context T is then updated with these new intervals; and its MI are computed. The process is iterated until all $M \in MI$ verify the stopping criterion S. The context T is initialized with, for each continuous attribute, an interval -*i.e.* a binary attribute- containing all continuous values observed in D; thus each object is in relation with every binary attributes of T. The GL of the inital context T contains only one concept (O, I) being a meet-irreducible concept, which is used to initialize MI. See [13] for more details on the algorithm.
The main difference with DT is that splitting an attribute in a GL impacts all the other concepts of the GL that contain this attribute, and due to the order relation between concepts $\leq$, the structure of the GL is also modified. Whereas, when an attribute is split in a DT node, predecessors and others branches are not impacted. In order to select the best $V^*(MI)$ over all the concepts sharing this attribute, we introduce different computing of $V^*(MI)$.

Let $MI = \{D_q = (A_q, B_q); q \leq Q\}\}$ be the set of meet-irreducible concepts not satisfying S. The best attribute $V^*(D_q)$ associated to its best cut-point is first computed for each concept $D_q \in MI$:

$$V^*(D_q) = argmax_{V \in B_q}(gain(V, c^*_V, D_q)) \quad (3)$$

where c^*_V is defined by (1) for D_q instead of D.
Let us define $I^*_{MI} = \{V^*(D_1), \ldots, V^*(D_Q)\}$ the set of best attributes associated to each concept in MI. The best attribute $V^*(MI)$ among I^*_{MI} can be defined in two different ways:
By local discretization: Local discretization selects the best attribute $V \in I^*_{MI}$ as the one having the best gain for MI:

$$V^*(MI) = argmax_{V^*(D_q) \in I^*_{MI}}(gain(V^*(D_q), c^*_{V^*(D_q)}, D_q)) \quad (4)$$

By linear local discretization: Linear local discretization takes into account that the split of one attribute $V \in I^*_{MI}$ in a concept D_q can impact the other concepts. So we compute a linear combination of the criterion as the sum of the gain for each concept $D_{q'} \in MI$ containing this attribute V. The selected attribute is the one that gives the best linear combination:

$$V^*(MI) = argmax_{V \in I^*_{MI}}(\sum_{D_{q'} \in MI | V \in B_{q'}} \frac{|A_{q'}|}{\sum_{D_q \in MI} |A_q|} * gain(V, c^*_V, D_{q'})) \quad (5)$$

3 Experimental comparison

The study is performed on three supervised databases of the UCI Machine Learning Repository[1]: the Image Segmentation database (Image1), the Glass Identification Database (GLASS) and the Breast Cancer Database (BREAST Cancer). We also use one supervised data set stemming from GREC 2003 database[2] described by the statistical Radon signature (GREC Radon). Table 1 presents the **complexity of each lattice structure** associated to each discretization algorithm and the **classification performance** using each GL by navigation [14] and using CHAID as DT classifier [6]. Discretization is performed in each case with χ^2 as a splitting and stopping supervised criterion.

4 Conclusion

The study [3] shows that for DTs, local discretization induces more complex structures compared to global discretization; Table 1 shows that **for GL, on the contrary, local discretization allows to reduce the structures' complexity**. In [7], Quinlan proves that local discretization improves classification performance of DTs compared to global discretization; as in DTs, Table 1 shows that **local discretization improves GLs classification performances**.

[1] http://archive:ics:uci:edu/ml [2] www.cvc.uab.es/grec2003/symreccontest/index.htm

Table 1. Structures complexity and Classification performance

	Nb concepts			Recognition rates			
	Local	Linear Local	Global	Local	Linear Local	Global	CHAID
Image1	**527**	649	12172	90.33	**91.57**	82.23	90.95
GLASS	**1950**	2128	2074	71.11	72.60	**73.18**	63.72
BREAST Cancer	3608	**2613**	7784	91.66	91.23	90.05	**93,47**
GREC Radon	**69**	92	2192	90.43	90.17	81.42	**92.94**

References

1. Ganter, B., Kuznetsov, S.: Pattern structures and their projections. In Delugach, H., Stumme, G., eds.: Conceptual Structures: Broadening the Base. Volume 2120 of LNCS. (2001) 129–142
2. Kaytoue, M., Kuznetsov, S.O., Napoli, A., Duplessis, S.: Mining gene expression data with pattern structures in formal concept analysis. Inf. Sci. **181** (2011) 1989–2001
3. Dougherty, J., Kohavi, R., Sahami, M.: Supervised and unsupervised discretization of continuous features. In: Machine Learning: Proc. of the Twelfth International Conference, Morgan Kaufmann (1995) 194–202
4. Muhlenbach, F., Rakotomalala, R.: Discretization of continuous attributes. In Reference, I.G., ed.: Encyclopedia of Data Warehousing and Mining. J. Wang (2005) 397–402
5. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and regression trees. Wadsworth Inc., 358 pp (1984)
6. Kass, G.: An exploratory technique for investigating large quantities of categorical data. Applied Statistics **29(2)** (1980) 119–127
7. Quinlan, J.: Improved use of continuous attributes in C4.5. Journal of Artificial Intelligence Research **4** (1996) 77–90
8. Guillas, S., Bertet, K., Visani, M., Ogier, J.M., Girard, N.: Some links between decision tree and dichotomic lattice. In: Proc. of the Sixth International Conference on Concept Lattices and Their Applications, CLA 2008 (2008) 193–205
9. Ganter, B., Wille, R.: Formal concept analysis, Mathematical foundations. Springer Verlag, Berlin, 284 pp (1999)
10. Fu, H., Fu, H., Njiwoua, P., Nguifo, E.M.: A comparative study of fca-based supervised classification algorithms. In: Concept Lattices. Volume LNCS 2961. (2004) 219–220
11. Birkhoff, G.: Lattice theory. Third edn. Volume 25. American Mathematical Society, 418 pp (1967)
12. Wille, R.: Restructuring lattice theory : an approach based on hierarchies of concepts. Ordered sets (1982) 445–470 I. Rival (ed.), Dordrecht-Boston, Reidel.
13. Girard, N., Bertet, K., Visani, M.: Local discretization of numerical data for galois lattices. In: Proceedings of the 23rd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2011 (2011) to appear.
14. Visani, M., Bertet, K., Ogier, J.M.: Navigala: an original symbol classifier based on navigation through a galois lattice. International Journal of Pattern Recognition and Artificial Intelligence, IJPRAI **25** (2011) 449–473

Formal Concept Analysis on Graphics Hardware

W. B. Langdon, Shin Yoo, and Mark Harman

CREST centre, Department of Computer Science,
University College London Gower Street, London WC1E 6BT, UK

Abstract. We document a parallel non-recursive beam search GPGPU FCA CbO like algorithm written in nVidia CUDA C and test it on software module dependency graphs. Despite removing repeated calculations and optimising data structures and kernels, we do not yet see major speed ups. Instead GeForce 295 GTX and Tesla C2050 report 141 072 concepts (maximal rectangles, clusters) in about one second. Future improvements in graphics hardware may make GPU implementations of Galois lattices competitive.

Keywords: software module clustering, MDG, close-by-one, arithmetic intensity

1 Introduction

Formal Concept Analysis [7] is a well known technique for grouping objects by the attributes they have in common. It can be thought of as discrete data clustering. In general the number of conceptual clusters grows exponentially. However there are a few specialised algorithms which render FCA manageable, even on quite large problems, provided the object-attribute table is sparse [10]. Krajca, Outrata and Vychodil [10] report considerable improvement in FCA algorithms in the last two decades. All these successful algorithms use depth first tree search to find all the conceptual clusters in an object-attribute table.

Computer graphics gaming cards (GPUs) are relatively cheap and yet offer far more computing power than the computer's CPU alone. (E.g. a 295 GTX contains 480 fully functioning processors and yet costs only a few hundred pounds.) Also microprocessor trends suggest faster computing will require parallel computing in future. There are already hundreds of millions of computers fitted with graphics hardware which might be used for general purpose computing [3].

Krajca *et al.* [10] report using a distributed computer to overcome the "major drawback [of FCA's] computational complexity". They report their parallel algorithm PCbO gives near linear speed increase with number of computing nodes in a network of up to 15 PCs. In other work [11] they conclude that there is no universal best FCA data structure. Instead they suggest that the optimum performance will depend upon the application. In earlier work, Huaiguo Fu had created a parallel implementation of NextClosure but it was limited to 50 attributes [5] but this was subsequently greatly extended [6]. However, like Krajca *et al.* [10], both Fu's [5] and [6] approaches use conventional distributed computers composed of a few CPUs rather than hundreds of GPU processing elements.

Local Proceedings in ISBN 978–2–905267–78–8, Inria NGE/LORIA, Nancy, France.

Similarly Djoufak Kengue *et al.* [4]'s ParCIM implementation used a conventional network of 8 computers connected in a star fashion with MPI. Ours is the first FCA implementation to run in parallel on computer graphics cards (GPUs).

2 CUDA FCA Implementation

Although In-close [1] claims to be faster we easily obtained FCbO [9] from Source Forge. We initially implemented the Krajca sequential algorithm [9] in Python. This was followed by a version in CUDA C, where `ComputeClosure` is implemented in parallel on the GPU. (For details see our technical report [13].)

Krajca's routines `ComputeClosure` and `GenerateFrom` essentially form a depth first search algorithm which builds and navigates a tree of formal concepts from a binary 0/1 matrix describing which object has which property. Since the search is recursive and operates on one point in the tree at one time, it is unsuitable for parallel operation on graphics cards. Our graphics card parallel version retains the tree but uses beam search rather than depth first search.

Instead of proceeding to the first leaf of the tree, recursively backing up and then going forward to the next leaf and so on, in beam search, we also start from the top of the tree and then proceed along every branch to the next level. This requires saving information on the beam for every node at that level. Beam search next expands the search again to cover everything at the next level and so on until all the leafs of the tree have been reached. Notice instead of working on a single point in the tree the beam covers many points which can be worked on in parallel. Indeed within a couple of levels we can get a beam containing tens of thousands of individual search points which can be processed independently. This suits the GPU architecture which needs literally thousands of independent processing threads for it to deliver its best performance [12]. You will have spotted that in an exponential problem, like FCA, beam search quickly runs out of memory.

Even for quite modest tree depths the beam width is limited by the available space in the GPU card. (We have a configuration limit of 1.8 million simultaneous parallel operations.) When a beam search exceeds this limit, only the first 1.8 million searches are loaded onto the GPU and the rest of the beam is queued on the host PC. (Although we have not done this, in multi-GPU systems it would be possible to split the beam between the GPUs, allocating up to 1.8 million to each GPU.) The GPU only searches to the next level. It returns the concepts found by the searches and the newly discovered branches which remain to be searched. The concepts are printed by the host PC and the new branches are added to the end of the beam to await their turn. Effectively the beam becomes a queue of points in the tree waiting to be searched. The number of parallel searches is mostly limited by the need to have space on the GPU for all the potential new branches. This depends upon the tree's fan out which is problem dependent. Nonetheless the GPU can manage modest real software engineering examples (e.g. dependence clustering of the Linux kernel). Notice the beam will contain a mixture of pending search points at different depths in the tree.

Table 1. Performance on, FCA benchmarks, random module dependency graphs, and Software Engineering datasets [8]. Time given in seconds, except longest Python run which is hours:mins:secs. (For $\frac{1}{2}$ 295 GTX and Tesla C2050 the total time on the GPU is given.)

Dataset	Size	Density	Concepts	FCbO	Python	295 GTX	C2050
krajca	5×7	54%	16	0.00	0.11	0.01	0.01
wiki	10×5	44%	14	0.00	0.03	0.00	0.00
random	10×10	20%	16	0.00	0.04	0.00	0.00
random	100×100	2%	137	0.00	0.40	0.02	0.01
random	200×200	2%	420	0.00	4.33	0.00	0.01
random	500×500	2%	2861	0.01	162.60	0.02	0.02
bison	37×37	24%	692	0.00	0.32	0.00	0.01
compiler	33×33	6%	24	0.00	0.05	0.00	0.00
dot	42×42	28%	1302	0.00	0.71	0.00	0.01
grappa	86×86	7%	850	0.00	2.54	0.01	0.01
incl	172×172	2%	238	0.00	1.84	0.00	0.01
ispell	24×24	34%	432	0.00	0.15	0.01	0.01
linuxConverted	955×955	2%	141072	0.73	15:42:51	1.79	0.93
mtunis	20×20	29%	110	0.00	0.05	0.00	0.01
rcs	29×29	37%	1074	0.00	0.46	0.01	0.02
swing	413×413	2%	3654	0.01	208.71	0.03	0.02

3 Results

FCbO (version 2010/10/05) was downloaded and compiled without changes on a 2.66 GHz PC with 3 Gigabytes of RAM running 64 bit CentOS 5.0. The performance of FCbO, our Python code and our CUDA code on two types of GPU are given in Table 1. They show performance on: two bench mark problems, a selection of randomly generated symmetric object-attribute pairings and software module dependency graphs of real world example programs.

4 Discussion

It is unclear why our code does not do better.

We would expect a linear speed advantage for FCbO from both using 64 bit operations and from using compiled rather than interpreted code. However on sizable examples, the ratio between the speed of FCbO and that of our Python code is huge. This hints that FCbO has some algorithmic advantage.

GPUs are often limited by the time taken to move data rather than to perform calculations. "Arithmetic intensity" is the ratio of calculations per data item. Typically this is in the range 4–64 FLOP/TDE [2, p206], we estimate the arithmetic intensity of Krajca *et al.*'s algorithm [9] is less than 1. Thus a potential problem might be there is simply is not enough computation required by FCA compared to the volume of data.

Newer versions of CUDA have make it easier to overlap GPU operations. However our implementation does not do this. Since the work is spread across the multi-processors, we suspect that idle time is not a major problem.

5 Conclusions

There are many problems which are traditionally solved by depth first search. However this may not suit low cost computer graphics GPU hardware. We have implemented a form of beam search and demonstrated it on several existing FCA benchmarks and ten software engineering dependence clustering problems [8]. GPU beam search may also be more widely applicable.

Acknowledgements

I am grateful for the assistance of Gernot Ziegler of nVidia. Steve Worley, Sarnath Kannan, Stephen Swift, Stan Seibert and Yuanyuan Zhang. Software engineering MDGs were supplied by Spiros Mancoridis. Tesla donated by nVidia. Funded by EPSRC grant EP/G060525/2.

References

1. S. J. Andrews. In-close, a fast algorithm for computing formal concepts. In *Conceptual Structures Tools Interoperability Workshop at the 17th International Conference on Conceptual Structures*, Moscow, 26-31 July 2009.
2. M. Christen, O. Schenk, and H. Burkhart. Automatic code generation and tuning for stencil kernels on modern shared memory architectures. *CSRD*, 26(3):205–210.
3. B. Del Rizzo. Dice puts faith in nvidia PhysX technology for Mirror's Edge. NVIDIA Corporation press release, Nov 19 2008.
4. J. Djoufak Kengue, P. Valtchev, and C. Tayou Djamegni. Parallel computation of closed itemsets and implication rule bases. In I. Stojmenovic, *et al.*, eds., *ISPA 2007*, *LNCS 4742*, pp359–370. Springer.
5. Huaiguo Fu and E. Nguifo. A parallel algorithm to generate formal concepts for large data. In P. Eklund, ed., *ICFCA*, *LNAI 2961*, pp141–142. Springer, 2004.
6. Huaiguo Fu and M. O'Foghlu. A distributed algorithm of density-based subspace frequent closed itemset mining. In *HPCC*, pp750–755. IEEE, 2008.
7. B. Ganter and R. Wille. *Formal Concept Analysis*. Springer, 1999.
8. M. Harman, S. Swift, and K. Mahdavi. An empirical study of the robustness of two module clustering fitness functions. In H.-G. Beyer, *et al.*, eds., *GECCO 2005*.
9. P. Krajca, J. Outrata, and V. Vychodil. Parallel recursive algorithm for FCA. In R. Belohlavek and S. O. Kuznetsov, eds., *CLA 2008*, Olomouc, Czech Republic.
10. P. Krajca, J. Outrata, and V. Vychodil. Parallel algorithm for computing fixpoints of Galois connections. *Ann Math Artif Intel*, 59:257–272, 2010.
11. P. Krajca and V. Vychodil. Comparison of data structures for computing formal concepts. In V. Torra, *et al.*, eds., *MDAI 2009*, *LNCS 5861*, pp114–125. Springer.
12. W. B. Langdon. Graphics processing units and genetic programming: An overview. *Soft Computing*, 15:1657–1669, Aug. 2011.
13. W. B. Langdon, S. Yoo, and M. Harman. Non-recursive beam search on GPU for formal concept analysis. RN/11/18, Computer Science, UCL, London, UK, 2011.

Author Index

Editors:	Amedeo Napoli, Vilem Vychodil
Publisher & Print:	INRIA Nancy – Grand Est and LORIA France
Title:	CLA 2011, Proceedings of the Eighth International Conference on Concept Lattices and Their Applications
Place, year, edition:	Nancy, 2011, 1st
Page count:	xii+419
Impression:	100
Archived at:	`http://cla.inf.upol.cz`

Not for sale

ISBN 978–2–905267–78–8