Lecture Notes in Artificial Intelligence 5548

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

Sébastien Ferré   Sebastian Rudolph (Eds.)

Formal Concept Analysis

7th International Conference, ICFCA 2009
Darmstadt, Germany, May 21-24, 2009
Proceedings

Springer

Volume Editors

Sébastien Ferré
IRISA/IFSIC, Université de Rennes 1
Campus Universitaire de Beaulieu, 35042 Rennes CEDEX, France
E-mail: Sebastien.Ferre@irisa.fr

Sebastian Rudolph
Universität Karlsruhe (TH), Institut AIFB
76128 Karlsruhe, Germany
E-mail: Sebastian.Rudolph@kit.edu

Library of Congress Control Number: Applied for

CR Subject Classification (1998): I.2, G.2.1, G.2.2, F.4.1, F.4.2, D.2.4, H.3

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN	0302-9743
ISBN-10	3-642-01814-9 Springer Berlin Heidelberg New York
ISBN-13	978-3-642-01814-5 Springer Berlin Heidelberg New York

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12683055 06/3180 5 4 3 2 1 0

Preface

The discipline of formal concept analysis (FCA) is concerned with the formalization of concepts and conceptual thinking. Built on the solid foundation of lattice and order theory, FCA is first and foremost a mathematical discipline. However, its motivation and guiding principles are based on strong philosophical underpinnings. In practice, FCA provides a powerful framework for the qualitative, formal analysis of data, as demonstrated by numerous applications in diverse areas. Likewise, it emphasizes the aspect of human-centered information processing by employing visualization techniques capable of revealing inherent structure in data in an intuitively graspable way. FCA thereby contributes to structuring and navigating the ever-growing amount of information available in our evolving information society and supports the process of turning data into information and ultimately into knowledge.

In response to an expanding FCA community, the International Conference on Formal Concept Analysis (ICFCA) was established to provide an annual opportunity for the exchange of ideas. Previous ICFCA conferences were held in Darmstadt (2003), Sydney (2004), Lens (2005), Dresden (2006), Clermont-Ferrand (2007), as well as Montreal (2008) and are evidence of vivid ongoing interest and activities in FCA theory and applications.

ICFCA 2009 took place during May 21–24 at the University of Applied Sciences in Darmstadt. Beyond serving as a host of the very first ICFCA in 2003, Darmstadt can be seen as the birthplace of FCA itself, where this discipline was introduced in the early 1980s and elaborated over the subsequent decades. On this occasion, we were very delighted to include a reprint of Rudolf Wille's seminal paper on formal concept analysis to make it easily available for our community.

Out of 29 submitted papers, 15 were accepted for publication in this volume amounting to an acceptance rate of 52%. Less mature works which were still considered valuable contributions for discussion were collected in a supplementary volume, published as "Contributions to ICFCA 2009" by Verlag Allgemeine Wissenschaft (ISBN 3-935924-08-9).

Clearly, the overall process of assembling a high-quality conference program would not have been possible without the much-appreciated help of the Program Committee members, external reviewers, and members of the Editorial Board. Finally, but most notably, we wish to thank the Conference Chair, Karl Erich Wolff. In collaboration with the Local Chair, Urs Andelfinger, and the local organization team, he realized a smoothly-run conference with a pleasant and friendly atmosphere. Our warmest thanks for our hosts' hospitality.

May 2009

Sébastien Ferré
Sebastian Rudolph

Organization

Executive Committee

Conference Chair

Karl Erich Wolff	University of Applied Sciences, Darmstadt, Germany

Local Chair

Urs Andelfinger	University of Applied Sciences, Darmstadt, Germany

Conference Organization Committee

Peter Burmeister	Technische Universität Darmstadt, Germany
Markus Helmerich	Technische Universität Darmstadt, Germany
Stefan Heppenheimer	Technische Universität Darmstadt, Germany
Rudolf Wille	Technische Universität Darmstadt, Germany

Program and Conference Proceedings

Program Chairs

Sébastien Ferré	Université de Rennes 1, France
Sebastian Rudolph	Universität Karlsruhe, Germany

Editorial Board

Peter Eklund	University of Wollongong, Australia
Bernhard Ganter	Technische Universität Dresden, Germany
Robert Godin	LATECE, Université du Québec à Montréal (UQAM), Montréal, Canada
Sergei O. Kuznetsov	Higher School of Economics, Moscow, Russia
Raoul Medina	LIMOS, Université Clermont-Ferrand 2, France
Rokia Missaoui	Université du Québec en Outaouais (UQO), Gatineau, Canada
Sergei Obiedkov	Higher School of Economics, Moscow, Russia
Uta Priss	Napier University, Edinburgh, UK
Stefan E. Schmidt	Technische Universität Dresden, Germany
Gerd Stumme	University of Kassel, Germany
Rudolf Wille	Technische Universität Darmstadt, Germany
Karl Erich Wolff	University of Applied Sciences, Darmstadt, Germany

Program Committee

Mike Bain	University of New South Wales, Sydney, Australia
Peter Becker	Concepta Consulting Pty Ltd., Australia
Radim Belohlavek	Binghamton University - State University of New York, USA
Sadok Ben Yahia	Faculty of Sciences of Tunis, Tunisia
Claudio Carpineto	Fondazione Ugo Bordoni, Italy
Frithjof Dau	SAP Research CEC Dresden, Germany
Vincent Duquenne	ECP6-CNRS, Université Paris 6, France
Linton Freeman	UCI, California, USA
Alain Gély	LITA, Université Paul Verlaine, Metz, France
Joachim Hereth	DMC GmbH, Germany
Wolfgang Hesse	Philipps-Universität Marburg, Germany
Bjoern Koester	Technische Universität Dresden, Germany
Derrick G. Kourie	University of Pretoria, South Africa
Markus Krötzsch	Universität Karlsruhe, Germany
Marzena Kryszkiewicz	Warsaw University of Technology, Poland
Leonard Kwuida	Dresden, Germany
Wilfried Lex	Universität Clausthal, Germany
Christian Lindig	Schloss Dagstuhl - Leibniz Center for Informatics, Germany
Engelbert Mephu Nguifo	IUT de Lens - Université d'Artois, France
Lhouari Nourine	LIMOS, Université Clermont Ferrand 2, France
Jean-Marc Petit	LIRIS, INSA Lyon, France
Alex Pogel	New Mexico State University, Las Cruces, USA
Sandor Radeleczki	University of Miskolc, Hungary
Camille Roth	CNRS/EHESS, Paris, France
Jurg Schmid	Universität Bern, Switzerland
Selma Strahringer	Cologne University of Applied Sciences, Germany
Petko Valtchev	Université du Québec à Montréal (UQAM), Montréal, Canada
Serhyi Yevtushenko	Luxoft, Ukraine

External Reviewers

Robert Jäschke	University of Kassel, Germany
Peggy Cellier	Université de Rennes 1, France
Denny Vrandečić	Universität Karlsruhe, Germany

Sponsoring Institutions

Hochschule Darmstadt, University of Applied Sciences
Fachbereich Mathematik der Hochschule Darmstadt
Fachbereich Informatik der Hochschule Darmstadt
Zentrum für Forschung und Entwicklung der Hochschule Darmstadt
Fachbereich Mathematik der Technischen Universität Darmstadt
Ernst-Schröder-Zentrum für Begriffliche Wissensverarbeitung

Table of Contents

Invited Talks

Theory

Algorithms

Applications

History

Usability Issues in Description Logic Knowledge Base Completion

Franz Baader and Barış Sertkaya[*]

TU Dresden, Germany
{baader,sertkaya}@tcs.inf.tu-dresden.de

Abstract. In a previous paper, we have introduced an approach for extending both the terminological and the assertional part of a Description Logic knowledge base by using information provided by the assertional part and by a domain expert. This approach, called knowledge base completion, was based on an extension of attribute exploration to the case of partial contexts. The present paper recalls this approach, and then addresses usability issues that came up during first experiments with a preliminary implementation of the completion algorithm. It turns out that these issues can be addressed by extending the exploration algorithm for partial contexts such that it can deal with implicational background knowledge.

1 Introduction

Description Logics (DLs) [1] are a successful family of logic-based knowledge representation formalisms, which can be used to represent the conceptual knowledge of an application domain in a structured and formally well-understood way. They are employed in various application domains, such as natural language processing, configuration, databases, and bio-medical ontologies, but their most notable success so far is due to the fact that DLs provide the logical underpinning of OWL, the standard ontology language for the semantic web [20]. As a consequence of this standardization, several ontology editors support OWL [19,22,23,27], and ontologies written in OWL are employed in more and more applications. As the size of these ontologies grows, tools that support improving their quality become more important. The tools available until now use DL reasoning to detect inconsistencies and to infer consequences, i.e., implicit knowledge that can be deduced from the explicitly represented knowledge. There are also promising approaches that allow to pinpoint the reasons for inconsistencies and for certain consequences, and that help the ontology engineer to resolve inconsistencies and to remove unwanted consequences [6,7,21,24,32]. These approaches address the quality dimension of *soundness* of an ontology, both within itself (consistency) and w.r.t. the intended application domain (no unwanted consequences). Here, we are concerned with a different quality dimension, namely *completeness* of the ontology w.r.t. to the intended application domain. In [5],

[*] Supported by the German Research Foundation (DFG) under grant BA 1122/12-1.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 1–21, 2009.

we have provided a basis for formally well-founded techniques and tools that support the ontology engineer in checking whether an ontology contains all the relevant information about the application domain, and in extending the ontology appropriately if this is not the case. In the present paper, we give an overview over this approach, and then describe how the general framework must be extended such that it becomes easier to use for a domain expert. But first, let us introduce the problem of knowledge base completion, and our approach for solving it, in a bit more detail.

A DL knowledge base (nowadays often called ontology) usually consists of two parts, the terminological part (TBox), which defines concepts and also states additional constraints (so-called general concept inclusions, GCIs) on the interpretation of these concepts, and the assertional part (ABox), which describes individuals and their relationship to each other and to concepts. Given an application domain and a DL knowledge base (KB) describing it, we can ask whether the KB contains all the "relevant" information[1] about the domain:

- Are all the relevant constraints that hold between concepts in the domain captured by the TBox?
- Are all the relevant individuals existing in the domain represented in the ABox?

As an example, consider the OWL ontology for human protein phosphatases that has been described and used in [37]. This ontology was developed based on information from peer-reviewed publications. The human protein phosphatase family has been well characterized experimentally, and detailed knowledge about different classes of such proteins is available. This knowledge is represented in the terminological part of the ontology. Moreover, a large set of human phosphatases has been identified and documented by expert biologists. These are described as individuals in the assertional part of the ontology. One can now ask whether the information about protein phosphatases contained in this ontology is complete. Are all the relationships that hold among the introduced classes of phosphatases captured by the constraints in the TBox, or are there relationships that hold in the domain, but do not follow from the TBox? Are all possible kinds of human protein phosphatases represented by individuals in the ABox, or are there phosphatases that have not yet been included in the ontology or even not yet been identified?

Such questions cannot be answered by an automated tool alone. Clearly, to check whether a given relationship between concepts—which does not follow from the TBox—holds in the domain, one needs to ask a domain expert, and the same is true for questions regarding the existence of individuals not described in the ABox. The rôle of the automated tool is to ensure that the expert is asked as few questions as possible; in particular, she should not be asked trivial questions, i.e., questions that could actually be answered based on the represented knowledge. In the above example, answering a non-trivial question regarding

[1] The notion of "relevant information" must, of course, be formalized appropriately for this problem to be addressed algorithmically.

human protein phosphatases may require the biologist to study the relevant literature, query existing protein databases, or even to carry out new experiments. Thus, the expert may be prompted to acquire new biological knowledge.

Attribute exploration [11] is an approach developed in Formal Concept Analysis (FCA) [14] that can be used to acquire knowledge about an application domain by querying an expert. One of the earliest applications of this approach is described in [29,36], where the domain is lattice theory, and the goal of the exploration process is to find, on the one hand, all valid relationships between properties of lattices (like being distributive), and, on the other hand, to find counterexamples to all the relationships that do not hold. To answer a query whether a certain relationship holds, the lattice theory expert must either confirm the relationship (by using results from the literature or carrying out a new proof for this fact), or give a counterexample (again, by either finding one in the literature or constructing a new one).

Although this sounds very similar to what is needed in our context, we could not directly use this approach in [5]. The main reason for this is the open-world semantics of description logic knowledge bases. Consider an individual i from an ABox $\mathcal{A}$ and a concept C occurring in a TBox $\mathcal{T}$. If we cannot deduce from the TBox $\mathcal{T}$ and $\mathcal{A}$ that i is an instance of C, then we do not assume that i does not belong to C. Instead, we only accept this as a consequence if $\mathcal{T}$ and $\mathcal{A}$ imply that i is an instance of $\neg C$. Thus, our knowledge about the relationships between individuals and concepts is incomplete: if $\mathcal{T}$ and $\mathcal{A}$ imply neither $C(i)$ nor $\neg C(i)$, then we do not know the relationship between i and C. In contrast, classical FCA and attribute exploration assume that the knowledge about individuals is complete: the basic datastructure is that of a formal context, i.e., a crosstable between individuals and properties. A cross says that the property holds, and the absence of a cross is interpreted as saying that the property does not hold. In contrast, in a *partial context*, a property may be known to hold or not to hold for an individual, but there is also the third possibility that nothing is known about the relationship between the individual and this property.

There has been some work on how to extend FCA and attribute exploration from complete knowledge to the case of such partial knowledge [9,10,17,18,28]. However, this work is based on assumptions that are different from ours. In particular, it assumes that the expert cannot answer all queries and, as a consequence, the knowledge obtained after the exploration process may still be incomplete and the relationships between concepts that are produced in the end fall into two categories: relationships that are valid no matter how the incomplete part of the knowledge is completed, and relationships that are valid only in some completions of the incomplete part of the knowledge. In contrast, our intention is to complete the KB, i.e., in the end we want to have complete knowledge about these relationships. What may be incomplete is the description of individuals used during the exploration process. From the FCA point of view, the main new result in [5] is the extension of attribute exploration to the case of partial contexts. This approach is then used to derive an algorithm for completing DL knowledge bases.

Before publishing [5], we had implemented a first experimental version of a tool for completing DL knowledge bases as an extension of the ontology editor Swoop [22], using the system Pellet as underlying reasoner [33]. A first evaluation of this tool on the OWL ontology for human protein phosphatases mentioned in the introduction, with biologists as experts, was quite promising, but also showed that the tool must be improved in order to be useful in practice. In particular, we have observed that the experts sometimes make errors when answering queries. Thus, the tool should support the expert in detecting such errors, and also make it possible to correct errors without having to restart the exploration process from scratch. Another usability issue on the wish list of our experts was to allow the postponement of answering certain queries, while continuing the exploration process with other queries.

The present paper is a follow-up work to [5], which addresses these usability issues. In the next section, we give a brief introduction to DLs, and then recall, in Section 3, the completion approach developed in [5]. For more details, we refer the reader to that paper as well as to the technical report [4] accompanying it. In Section 4, we address the usability issues mentioned above. From the FCA point of view, our solution to these problems requires an extension of the results in [5] to the case of attribute exploration *w.r.t. background knowledge and partial contexts.* In Section 5, we describe our implementation of the improved approach, which is now realized as an OWL plugin to PROTÉGÉ 4 [19], and uses the incremental reasoning facilities of Pellet [16].

2 Description Logics

In order to represent knowledge about an application domain using Description Logics (DLs) one usually first defines the relevant concepts of this domain, and then describes relationships between concepts and relationships between individuals and concepts in the knowledge base. To construct concepts, one starts with a set N_C of *concept names* (unary predicates) and a set N_R of *role names* (binary predicates), and builds complex *concept descriptions* out of them by using the *concept constructors* provided by the particular *description language* being used. In addition, a set N_I of *individual names* is used to refer to domain elements. Table 1 displays commonly used concept constructors. In this table C and D stand for concept descriptions, r for a role name, and $a, b, a_1, \ldots, a_n$ for individual names. In the current paper, we do not fix a specific set of constructors since our *results apply to arbitrary DLs* as long as they allow for the constructors conjunction and negation (see the upper part of Table 1). For our purposes, a *TBox* is a finite set of general concept inclusions (GCIs), and an *ABox* is a finite set of concept and role assertions (see the lower part of Table 1). A *knowledge base (KB)* consists of a TBox together with an ABox. As usual, we use $C \equiv D$ as an abbreviation for the two GCIs $C \sqsubseteq D$ and $D \sqsubseteq C$.

The semantics of concept descriptions, TBoxes, and ABoxes is given in terms of an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ (the *domain*) is a non-empty set, and $\cdot^{\mathcal{I}}$ (the *interpretation function*) maps each concept name $A \in N_C$ to a set

Table 1. Syntax and semantics of commonly used constructors

Constructor name	Syntax	Semantics
negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
existential restriction	$\exists r.C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y : (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
one-of	$\{a_1, \ldots, a_n\}$	$\{a_1^{\mathcal{I}}, \ldots, a_n^{\mathcal{I}}\}$
general concept inclusion	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
concept assertion	$C(a)$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
role assertion	$r(a,b)$	$(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$

$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, each role name $r \in N_R$ to a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and each individual name $a \in N_I$ to an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. Concept descriptions C are also interpreted as sets $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, which are defined inductively, as seen in the semantics column of Table 1. An interpretation $\mathcal{I}$ is a *model* of the TBox $\mathcal{T}$ (the ABox $\mathcal{A}$) if it satisfies all its GCIs (assertions) in the sense shown in the semantics column of the table. In case $\mathcal{I}$ is a model of both $\mathcal{T}$ and $\mathcal{A}$, it is called a model of the knowledge base $(\mathcal{T}, \mathcal{A})$.

Given a KB $(\mathcal{T}, \mathcal{A})$, concept descriptions C, D, and an individual name a, the inference problems *subsumption*, *instance*, and *consistency* are defined as follows:

- *Subsumption:* C is *subsumed* by D w.r.t. $\mathcal{T}$ (written $C \sqsubseteq_{\mathcal{T}} D$) if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for all models $\mathcal{I}$ of $\mathcal{T}$
- *Instance:* a is an *instance* of C w.r.t. $\mathcal{T}$ and $\mathcal{A}$ (written $\mathcal{T}, \mathcal{A} \models C(a)$) if $a^{\mathcal{I}} \in C^{\mathcal{I}}$ holds for all models of $(\mathcal{T}, \mathcal{A})$.
- *Consistency:* the knowledge base $(\mathcal{T}, \mathcal{A})$ is *consistent* if it has a model.

For most DLs, these problems are decidable, and there exist highly optimized DL reasoners such as FaCT++ [35], RacerPro [15], Pellet [33], KAON2 [25], and HermiT [26], which can solve these problems for very expressive DLs on large knowledge bases from practical applications.

The following example demonstrates how a DL that has the constructors conjunction, disjunction, existential restriction, and one-of can be used to model some simple properties of countries.

Example 1. Assume that our set of concept names N_C contains the concepts Country, Ocean, and Sea; the set of role names N_R contains the roles hasBorderTo, isMemberOf, and hasOfficialLanguage; and the set of individual names N_I contains the individuals German, EU, MediterraneanSea, Italy, and Germany. Using these names, the following TBox defines a coastal country as a country that has a border to a sea or an ocean; a Mediterranean country as a country that has border to the MediterraneanSea; a German-speaking country as a country that has the language German as an official language; and an EU member as a country that is a member of the EU.

$$\begin{aligned}\mathcal{T}_{countries} := \{\ \mathsf{Coastal} &\equiv \mathsf{Country} \sqcap \exists \mathsf{hasBorderTo}.(\mathsf{Ocean} \sqcup \mathsf{Sea})\\ \mathsf{EUmember} &\equiv \mathsf{Country} \sqcap \exists \mathsf{isMemberOf}.\{\mathsf{EU}\}\\ \mathsf{Mediterranean} &\equiv \mathsf{Country} \sqcap \exists \mathsf{hasBorderTo}.\{\mathsf{MediterraneanSea}\}\\ \mathsf{GermanSpeaking} &\equiv \mathsf{Country} \sqcap \exists \mathsf{hasOfficialLanguage}.\{\mathsf{German}\}\ \}\end{aligned}$$

The following ABox states facts about the countries Italy and Germany, and about the Mediterranean Sea:

$$\begin{aligned}\mathcal{A}_{countries} := \{&\mathsf{GermanSpeaking(Germany)},\ \mathsf{EUmember(Germany)},\\ &\mathsf{Coastal(Germany)},\ \mathsf{Mediterranean(Italy)},\ \mathsf{Sea(MediterraneanSea)}\}\end{aligned}$$

3 Partial Contexts, Attribute Exploration, and Completion of DL Knowledge Bases

In [5], we have extended the classical approach to FCA to the case of objects that have only a partial description in the sense that, for some attributes, it is not known whether they are satisfied by the object or not. This was needed due to the open-world semantics of DL knowledge bases. If an assertion $C(a)$ does not follow from a knowledge base, then one does not assume that its negation holds. Thus, if neither $C(a)$ nor $\neg C(a)$ follows, then we do not know whether a has the property C or not. In this section, we first give the basic definitions for extending FCA to the case of partially described objects, and then introduce a version of attribute exploration that works in this setting. More details can be found in [5,4]. In the following, we assume that we have a finite set M of attributes and a (possibly infinite) set of objects.

Definition 1. *A* partial object description (pod) *is a tuple* (A, S) *where* $A, S \subseteq M$ *are such that* $A \cap S = \emptyset$. *We call such a pod a* full object description (fod) *if* $A \cup S = M$. *A set of pods is called a* partial context *and a set of fods a* full context.

Intuitively, the pod (A, S) says that the object it describes satisfies all attributes from A and does not satisfy any attribute from S. For the attributes not contained in $A \cup S$, nothing is known w.r.t. this object. A partial context can be extended by either adding new pods or by extending existing pods.

Definition 2. *We say that the pod* (A', S') extends *the pod* (A, S), *and write this as* $(A, S) \leq (A', S')$, *if* $A \subseteq A'$ *and* $S \subseteq S'$. *Similarly, we say that the partial context* $\mathcal{K}'$ extends *the partial context* $\mathcal{K}$, *and write this as* $\mathcal{K} \leq \mathcal{K}'$, *if every pod in* $\mathcal{K}$ *is extended by some pod in* $\mathcal{K}'$. *If* $\overline{\mathcal{K}}$ *is a full context and* $\mathcal{K} \leq \overline{\mathcal{K}}$, *then* $\overline{\mathcal{K}}$ *is called a* realizer *of* $\mathcal{K}$. *If* $(\overline{A}, \overline{S})$ *is a fod and* $(A, S) \leq (\overline{A}, \overline{S})$, *then we also say that* $(\overline{A}, \overline{S})$ realizes (A, S).

Next, we introduce the notion of an implication between attributes, which formalizes the informal notion “relationship between properties” used in the introduction.

Definition 3. *An* implication *is of the form $L \to R$ where $L, R \subseteq M$. This implication is* refuted *by the pod (A, S) if $L \subseteq A$ and $R \cap S \neq \emptyset$. It is* refuted *by the partial context $\mathcal{K}$ if it is refuted by at least one element of $\mathcal{K}$. The set of implications that are not refuted by a given partial context $\mathcal{K}$ is denoted by $Imp(\mathcal{K})$. The set of all fods that do not refute a given set of implications $\mathcal{L}$ is denoted by $Mod(\mathcal{L})$.*

Obviously, $\mathcal{K} \leq \mathcal{K}'$ implies that every implication refuted by $\mathcal{K}$ is also refuted by $\mathcal{K}'$. For a set of implications $\mathcal{L}$ and a set $P \subseteq M$, the *implicational closure* of P with respect to $\mathcal{L}$, denoted by $\mathcal{L}(P)$, is the smallest subset Q of M such that

- $P \subseteq Q$, and
- $L \to R \in \mathcal{L}$ and $L \subseteq Q$ imply $R \subseteq Q$.

A set $P \subseteq M$ is called *$\mathcal{L}$-closed* if $\mathcal{L}(P) = P$.

Definition 4. *The implication $L \to R$ is said to* follow *from a set $\mathcal{J}$ of implications if $R \subseteq \mathcal{J}(L)$. The set of implications $\mathcal{J}$ is called* complete *for a set of implications $\mathcal{L}$ if every implication in $\mathcal{L}$ follows from $\mathcal{J}$. It is called* sound *for $\mathcal{L}$ if every implication that follows from $\mathcal{J}$ is contained in $\mathcal{L}$. A set of implications $\mathcal{J}$ is called a* base *for a set of implications $\mathcal{L}$ if it is both sound and complete for $\mathcal{L}$, and no strict subset of $\mathcal{J}$ satisfies this property.*

The following fact is trivial, but turns out to be crucial for our attribute exploration algorithm.

Proposition 1. *For a given set $P \subseteq M$ and a partial context $\mathcal{K}$, $\mathcal{K}(P) := M \setminus \bigcup\{S \mid (A, S) \in \mathcal{K}, P \subseteq A\}$ is the largest subset of M such that $P \to \mathcal{K}(P)$ is not refuted by $\mathcal{K}$.*

Attribute Exploration with Partial Contexts

The classical attribute exploration algorithm of FCA [11,14] assumes that there is a domain expert that can answer questions regarding the validity of implications in the application domain. Accordingly, our approach requires an expert that can decide whether an implication is refuted in the application domain or not. In contrast to existing work on extending FCA to the case of partial knowledge [9,17,18,10], we do *not* assume that the expert has only partial knowledge and thus cannot answer all implication questions.

To be more precise, we consider the following setting. We are given an initial (possibly empty) partial context $\mathcal{K}$, an initially empty set of implications $\mathcal{L}$, and a full context $\overline{\mathcal{K}}$ that is a realizer of $\mathcal{K}$. The expert answers implication questions "$L \to R$?" w.r.t. the full context $\overline{\mathcal{K}}$. More precisely, if the answer is "yes," then $\overline{\mathcal{K}}$ does not refute $L \to R$. The implication $L \to R$ is then added to $\mathcal{L}$. Otherwise, the expert extends the current context $\mathcal{K}$ such that the extended context refutes $L \to R$ and still has $\overline{\mathcal{K}}$ as a realizer. Consequently, the following invariant will be satisfied by $\mathcal{K}, \overline{\mathcal{K}}, \mathcal{L}$: $\quad \mathcal{K} \leq \overline{\mathcal{K}} \subseteq Mod(\mathcal{L})$.

Since $\overline{\mathcal{K}} \subseteq Mod(\mathcal{L})$ implies $\mathcal{L} \subseteq Imp(\overline{\mathcal{K}})$, this invariant ensures that $\mathcal{L}$ is sound for $Imp(\overline{\mathcal{K}})$. Our aim is to enrich $\mathcal{K}$ and $\mathcal{L}$ such that eventually $\mathcal{L}$ is also complete for $\mathit{Imp}(\overline{\mathcal{K}})$, and $\mathcal{K}$ refutes all other implications (i.e., all the implications refuted by $\overline{\mathcal{K}}$). As in the classical case, we want to do this by asking as few as possible questions to the expert.

Definition 5. *Let $\mathcal{L}$ be a set of implications and $\mathcal{K}$ a partial context. An implication is called* undecided *w.r.t. $\mathcal{K}$ and $\mathcal{L}$ if it neither follows from $\mathcal{L}$ nor is refuted by $\mathcal{K}$. It is* decided *w.r.t. $\mathcal{K}$ and $\mathcal{L}$ if it is not undecided w.r.t. $\mathcal{K}$ and $\mathcal{L}$.*

In principle, our attribute exploration algorithm tries to decide each undecided implication by either adding it to $\mathcal{L}$ or extending $\mathcal{K}$ such that it refutes the implication. If all implications are decided, then our goal is achieved [4].

Proposition 2. *Assume that $\mathcal{K} \leq \overline{\mathcal{K}} \subseteq \mathit{Mod}(\mathcal{L})$ and that all implications are decided w.r.t. $\mathcal{K}$ and $\mathcal{L}$. Then $\mathcal{L}$ is complete for $\mathit{Imp}(\overline{\mathcal{K}})$ and $\mathcal{K}$ refutes all implications not belonging to $\mathit{Imp}(\overline{\mathcal{K}})$.*

How can we find—and let the expert decide—all undecided implications without naively considering all implications? The following proposition motivates why it is sufficient to consider implications whose left-hand sides are $\mathcal{L}$-closed. It is an immediate consequence of the fact that $\mathcal{L}(\cdot)$ is a closure operator, and thus idempotent.

Proposition 3. *Let $\mathcal{L}$ be a set of implications and $L \rightarrow R$ an implication. Then, $L \rightarrow R$ follows from $\mathcal{L}$ iff $\mathcal{L}(L) \rightarrow R$ follows from $\mathcal{L}$.*

Concerning right-hand sides, Proposition 1 says that the largest right-hand side R such that $L \rightarrow R$ is not refuted by $\mathcal{K}$ is $R = \mathcal{K}(L)$. Putting these two observations together, we only need to consider implications of the form $L \rightarrow \mathcal{K}(L)$ where L is $\mathcal{L}$-closed. In order to enumerate all left-hand sides, we can thus use the well-known approach from FCA for enumerating closed sets in the lectic order [14]. In this approach, the elements of M are assumed to have a fixed order that imposes a linear order on the power set of M, called the *lectic order*. An algorithm that, given $\mathcal{L}$ and an $\mathcal{L}$-closed set P, computes in polynomial time the lectically next $\mathcal{L}$-closed set that comes after P, is described in [11].

If an implication is added because the expert has stated that it holds in $\overline{\mathcal{K}}$, then we can extend the current context $\mathcal{K}$ by closing the first component of every pod in $\mathcal{K}$ w.r.t. the new set of implications $\mathcal{L}$. In fact, $\mathcal{L} \subseteq \mathit{Imp}(\overline{\mathcal{K}})$ makes sure that the extended context is still realized by $\overline{\mathcal{K}}$. To allow for this and possible other ways of extending the partial context, the formulation of the algorithm just says that, in case an implication is added, the partial context can also be extended. Whenever an implication is not accepted by the expert, $\mathcal{K}$ will be extended to a context that refutes the implication and still has $\overline{\mathcal{K}}$ as a realizer.

Based on these considerations, an attribute exploration algorithm for partial contexts was introduced in [5], and is here recalled as Algorithm 1.

The following theorem states that this algorithm always terminates, and in which sense it is correct.

Theorem 1. *Let M be a finite set of attributes, and $\overline{\mathcal{K}}$ and $\mathcal{K}_0$ respectively a full and a partial context over the attributes in M such that $\mathcal{K}_0 \leq \overline{\mathcal{K}}$. Then*

Algorithm 1. Attribute exploration for partial contexts

```
Input: $M = \{m_1, \ldots, m_n\}, \mathcal{K}_0$    {attribute set and partial context, realized by full context $\overline{\mathcal{K}}$.}
$\mathcal{K} := \mathcal{K}_0$    {initialize partial context.}
$\mathcal{L} := \emptyset$    {initial empty set of implications.}
$P := \emptyset$    {lectically smallest $\mathcal{L}$-closed set.}
while $P \neq M$ do
  Compute $\mathcal{K}(P)$
  if $P \neq \mathcal{K}(P)$ then {$P \to \mathcal{K}(P)$ is undecided.}
    Ask the expert if $P \to \mathcal{K}(P)$ is refuted by $\overline{\mathcal{K}}$
    if no then {$P \to \mathcal{K}(P)$ not refuted.}
      $\mathcal{K} := \mathcal{K}'$ where $\mathcal{K}'$ is a partial context such that
        $\mathcal{K} \leq \mathcal{K}' \leq \overline{\mathcal{K}}$    {optionally extend $\mathcal{K}$.}
      $\mathcal{L} := \mathcal{L} \cup \{P \to \mathcal{K}(P) \setminus P\}$
      $P_{\text{new}}$ := lectically next $\mathcal{L}$-closed set after $P$
    else {$P \to \mathcal{K}(P)$ refuted.}
      Get a partial context $\mathcal{K}'$ from the expert such that $\mathcal{K} \leq \mathcal{K}' \leq \overline{\mathcal{K}}$ and
      $P \to \mathcal{K}(P)$ is refuted by $\mathcal{K}'$
      $\mathcal{K} := \mathcal{K}'$
      $P_{\text{new}} := P$    {$P$ not changed.}
    end if
  else {trivial implication.}
    $P_{\text{new}}$ := lectically next $\mathcal{L}$-closed set after $P$
  end if
  $P := P_{\text{new}}$
end while
```

Algorithm 1 terminates and, upon termination, it outputs a partial context $\mathcal{K}$ and a set of implications $\mathcal{L}$ such that

1. *$\mathcal{L}$ is a base for Imp$(\overline{\mathcal{K}})$, and*
2. *$\mathcal{K}$ refutes every implication that is refuted by $\overline{\mathcal{K}}$.*

DLs and Partial Contexts

Let $(\mathcal{T}, \mathcal{A})$ be a consistent DL knowledge base, and M be a finite set of concept descriptions. An individual name a occurring in $\mathcal{A}$ gives rise to the *partial object description* $pod_{\mathcal{T},\mathcal{A}}(a, M) := (A, S)$ where

$$A := \{C \in M \mid \mathcal{T}, \mathcal{A} \models C(a)\} \text{ and } S := \{C \in M \mid \mathcal{T}, \mathcal{A} \models \neg C(a)\}.$$

The whole ABox induces the partial context

$$\mathcal{K}_{\mathcal{T},\mathcal{A}}(M) := \{pod_{\mathcal{T},\mathcal{A}}(a, M) \mid a \text{ an individual name in } \mathcal{A}\}.$$

Similarly, any element $d \in \Delta^{\mathcal{I}}$ of an interpretation $\mathcal{I}$ gives rise to the *full object description* $fod_{\mathcal{I}}(d, M) := (\overline{A}, \overline{S})$ where

$$\overline{A} := \{C \in M \mid d \in C^{\mathcal{I}}\} \text{ and } \overline{S} := \{C \in M \mid d \in (\neg C)^{\mathcal{I}}\}.$$

The whole interpretation induces the full context

$$\mathcal{K}_{\mathcal{I}}(M) := \{fod_{\mathcal{I}}(d, M) \mid d \in \Delta^{\mathcal{I}}\}.$$

Proposition 4. *Let $(\mathcal{T}, \mathcal{A}), (\mathcal{T}', \mathcal{A}')$ be DL knowledge bases such that $\mathcal{T} \subseteq \mathcal{T}'$ and $\mathcal{A} \subseteq \mathcal{A}'$, M a set of concept descriptions, and $\mathcal{I}$ a model of $(\mathcal{T}', \mathcal{A}')$. Then $\mathcal{K}_{\mathcal{T},\mathcal{A}}(M) \leq \mathcal{K}_{\mathcal{T}',\mathcal{A}'}(M) \leq \mathcal{K}_{\mathcal{I}}(M)$.*

We can straightforwardly transfer the notion of refutation of an implication from partial (full) contexts to knowledge bases (interpretations).

Definition 6. *The implication $L \to R$ over the attributes M is* refuted *by the knowledge base $(\mathcal{T}, \mathcal{A})$ if it is refuted by $\mathcal{K}_{\mathcal{T},\mathcal{A}}(M)$, and it is* refuted *by the interpretation $\mathcal{I}$ if it is refuted by $\mathcal{K}_{\mathcal{I}}(M)$. If an implication is not refuted by $\mathcal{I}$, then we say that it* holds in $\mathcal{I}$. *In addition, we say that $L \to R$* follows from $\mathcal{T}$ *if $\sqcap L \sqsubseteq_{\mathcal{T}} \sqcap R$, where $\sqcap L$ and $\sqcap R$ respectively stand for the conjunctions $\bigsqcap_{C \in L} C$ and $\bigsqcap_{D \in R} D$.*

Obviously, the implication $L \to R$ holds in $\mathcal{I}$ iff $(\sqcap L)^{\mathcal{I}} \subseteq (\sqcap R)^{\mathcal{I}}$. As an immediate consequence of this fact, we obtain:

Proposition 5. *Let $\mathcal{T}$ be a TBox and $\mathcal{I}$ be a model of $\mathcal{T}$. If $L \to R$ follows from $\mathcal{T}$, then it holds in $\mathcal{I}$.*

Completion of DL KBs: Formal Definition and Algorithm

We are now ready to define what we mean by a completion of a DL knowledge base. Intuitively, the knowledge base is supposed to describe an intended model. For a fixed set M of "interesting" concepts, the knowledge base is complete if it contains all the relevant knowledge about implications between these concepts. Based on the notions introduced above, this is formalized as follows.

Definition 7. *Let $(\mathcal{T}, \mathcal{A})$ be a consistent DL knowledge base, M a finite set of concept descriptions, and $\mathcal{I}$ a model of $(\mathcal{T}, \mathcal{A})$. Then $(\mathcal{T}, \mathcal{A})$ is M-*complete *(or* complete *if M is clear from the context)* w.r.t. $\mathcal{I}$ *if the following three statements are equivalent for all implications $L \to R$ over M:*

1. *$L \to R$ holds in $\mathcal{I}$;*
2. *$L \to R$ follows from $\mathcal{T}$;*
3. *$L \to R$ is not refuted by $(\mathcal{T}, \mathcal{A})$.*

Let $(\mathcal{T}_0, \mathcal{A}_0)$ be a DL knowledge base and $\mathcal{I}$ a model of $(\mathcal{T}_0, \mathcal{A}_0)$. Then $(\mathcal{T}, \mathcal{A})$ is an M-completion of $(\mathcal{T}_0, \mathcal{A}_0)$ w.r.t. $\mathcal{I}$ *if it is M-complete w.r.t. $\mathcal{I}$ and extends $(\mathcal{T}_0, \mathcal{A}_0)$, i.e., $\mathcal{T}_0 \subseteq \mathcal{T}$ and $\mathcal{A}_0 \subseteq \mathcal{A}$.*

An adaptation of the attribute exploration algorithm for partial contexts presented above can be used to compute a completion of a given knowledge base $(\mathcal{T}_0, \mathcal{A}_0)$ w.r.t. a fixed model $\mathcal{I}$ of this knowledge base. It is assumed that the *expert* has or can obtain enough information about this model to be able to answer questions of the form "Is $L \rightarrow R$ refuted by $\mathcal{I}$?". If the answer is "no," then $L \rightarrow R$ holds according to the expert's opinion, and is thus added to the implication base computed by the algorithm. In addition, the GCI $\sqcap L \sqsubseteq \sqcap R$ is added to the TBox. Since $L \rightarrow R$ is not refuted by $\mathcal{I}$, the interpretation $\mathcal{I}$ is still a model of the new TBox obtained this way. If the answer is "yes," then the expert is asked to extend the current ABox (by adding appropriate assertions on either old or new individual names) such that the extended ABox refutes $L \rightarrow R$ and $\mathcal{I}$ is still a model of this ABox.

It is possible to optimize this algorithm by employing DL reasoning. Because of Proposition 5, before actually asking the expert whether the implication $L \rightarrow R$ is refuted by $\mathcal{I}$, we can first check whether $\sqcap L \sqsubseteq \sqcap R$ already follows from the current TBox. If this is the case, then we know that $L \rightarrow R$ cannot be refuted by $\mathcal{I}$, and we tacitly accept and add this implication to the current set of implications $\mathcal{L}$. Similarly, there are also cases where an implication can be rejected without asking the expert because accepting it would make the knowledge base inconsistent. However, in this case the expert still needs to extend the ABox such that the implication is refuted. The following example illustrates this case, which was not taken into account in the original version of the completion algorithm in [5].

Example 2. Consider the knowledge base $(\mathcal{T}, \mathcal{A})$ with the empty TBox $\mathcal{T} = \emptyset$, and the ABox $\mathcal{A} = \{(\exists r.A \sqcap \forall r.\neg B)(a)\}$. Clearly, the implication $\{A\} \rightarrow \{B\}$ does not follow from $\mathcal{T}$. Moreover, it is not refuted by $\mathcal{A}$ because this ABox does not explicitly contain a named individual that is an instance of both A and $\neg B$. That is, the implication $\{A\} \rightarrow \{B\}$ is undecided. However, if the user accepted this implication, and thus the GCI $A \sqsubseteq B$ were added to $\mathcal{T}$, the knowledge base would become inconsistent since the assertion in $\mathcal{A}$ enforces the existence of an implicit individual that belongs to both A and $\neg B$. This shows that this GCI cannot hold in the underlying model $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A})$, and thus it is refuted in the full context $\mathcal{K}_{\mathcal{I}}(M)$.

The improved completion algorithm for DL knowledge bases obtained from these considerations is described in Algorithm 2. Note that Algorithm 2, applied to $\mathcal{T}_0$, $\mathcal{A}_0, M$ with the underlying model $\mathcal{I}$ of $(\mathcal{T}_0, \mathcal{A}_0)$, is an instance of Algorithm 1, applied to the partial context $\mathcal{K}_{\mathcal{T}_0,\mathcal{A}_0}(M)$ with the underlying full context $\mathcal{K}_{\mathcal{I}}(M)$ as realizer. For this reason, the next theorem is an easy consequence of Theorem 1.

Theorem 2. *Let $(\mathcal{T}_0, \mathcal{A}_0)$ be a consistent knowledge base, M a finite set of concept descriptions, and $\mathcal{I}$ a model of $(\mathcal{T}_0, \mathcal{A}_0)$, and let $(\mathcal{T}, \mathcal{A})$ be the knowledge base computed by Algorithm 2. Then $(\mathcal{T}, \mathcal{A})$ is a completion of $(\mathcal{T}_0, \mathcal{A}_0)$.*

Let us demonstrate the execution of Algorithm 2 on an extension of the knowledge base constructed in Example 1, where M consists of the concepts $\mathsf{Coastal}$, $\mathsf{Mediterranean}$, $\mathsf{EUmember}$, and $\mathsf{GermanSpeaking}$, the ABox contains additional information on some countries, and $\mathcal{I}$ is the "real world."

Example 3. Let the partial context derived from the initial ABox be the one depicted in Table 2. Given this ABox, and the TBox in Example 1, the first implication question posed to the expert is $\{\mathsf{GermanSpeaking}\} \rightarrow \{\mathsf{EUmember}, \mathsf{Coastal}\}$.

Algorithm 2. Completion of DL knowledge bases

```
**Input**: $M = \{m_1, \ldots, m_n\}$, $(\mathcal{T}_0, \mathcal{A}_0)$    $\{$attribute set; KB with model $\mathcal{I}$.$\}$
$\mathcal{T} := \mathcal{T}_0$,   $\mathcal{A} := \mathcal{A}_0$
$\mathcal{L} := \emptyset$    $\{$initial empty set of implications.$\}$
$P := \emptyset$    $\{$lectically smallest $\mathcal{L}$-closed subset of $M$.$\}$
while $P \neq M$ do
  Compute $\mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$
  if $P \neq \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ then $\{$check whether the implication follows from $\mathcal{T}$.$\}$
    if $\sqcap P \sqsubseteq_{\mathcal{T}} \sqcap \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ then
      $\mathcal{L} := \mathcal{L} \cup \{P \rightarrow \mathcal{K}_{\mathcal{T},\mathcal{A}}(P) \setminus P\}$
      $P_{\mathrm{new}}$ := lectically next $\mathcal{L}$-closed set after $P$
    else
      if $(\mathcal{T} \cup \{\sqcap P \sqsubseteq \sqcap \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)\}, \mathcal{A})$ is inconsistent then
        Get an ABox $\mathcal{A}'$ from the expert such that $\mathcal{A} \subseteq \mathcal{A}'$, $\mathcal{I}$ is a model of $\mathcal{A}'$,
          and $P \rightarrow \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ is refuted by $\mathcal{A}'$
        $\mathcal{A} := \mathcal{A}'$    $\{$extend the ABox.$\}$
      else
        Ask expert if $P \rightarrow \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ is refuted by $\mathcal{I}$.
        if no then $\{\sqcap P \sqsubseteq \sqcap \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ is satisfied in $\mathcal{I}$.$\}$
          $\mathcal{L} := \mathcal{L} \cup \{P \rightarrow \mathcal{K}_{\mathcal{T},\mathcal{A}}(P) \setminus P\}$
          $P_{\mathrm{new}}$ := lectically next $\mathcal{L}$-closed set after $P$
          $\mathcal{T} := \mathcal{T} \cup \{\sqcap P \sqsubseteq \sqcap(\mathcal{K}_{\mathcal{T},\mathcal{A}}(P) \setminus P)\}$
        else
          Get an ABox $\mathcal{A}'$ from the expert such that $\mathcal{A} \subseteq \mathcal{A}'$,
            $\mathcal{I}$ is a model of $\mathcal{A}'$, and $P \rightarrow \mathcal{K}_{\mathcal{T},\mathcal{A}}(P)$ is refuted by $\mathcal{A}'$
          $\mathcal{A} := \mathcal{A}'$    $\{$extend the ABox.$\}$
        end if
      end if
    end if
  else $\{$trivial implication.$\}$
    $P_{\mathrm{new}}$ := lectically next $\mathcal{L}$-closed set after $P$
  end if
  $P := P_{\mathrm{new}}$
end while
```

The answer is "no," since Austria is German-speaking, but it is not a coastal country. Assume that the expert turns Austria into a counterexample by asserting that it is German-speaking. The second question is then whether the implication $\{\mathsf{GermanSpeaking}\} \rightarrow \{\mathsf{EUmember}\}$ holds. The answer is again "no" since Switzerland is a German-speaking country, but not an EU member. Assume that the expert adds the new individual Switzerland to the ABox, and asserts that it is an instance of GermanSpeaking and $\neg\mathsf{EUmember}$. The next question is $\{\mathsf{Mediterranean}\} \rightarrow \{\mathsf{EUmember}, \mathsf{Coastal}\}$. The answer is again "no" because Turkey is a Mediterranean country, but it is not an EU member. Assume that the expert adds the individual Turkey to the ABox, and asserts that it is an instance of $\neg\mathsf{EUmember}$. The next question $\{\mathsf{Mediterranean}\} \rightarrow \{\mathsf{Coastal}\}$ follows from the TBox. Thus, it is not posed to the expert, and the algorithm continues

Table 2. The partial context before completion

	Coastal	Mediterranean	EUmember	GermanSpeaking
Italy	+	+	+	−
India	+	−	−	−
Germany	+	−	+	+
Austria	−	−	+	?

Table 3. The partial context after completion

	Coastal	Mediterranean	EUmember	GermanSpeaking
Italy	+	+	+	−
India	+	−	−	−
Germany	+	−	+	+
Austria	−	−	+	+
Switzerland	−	−	−	+
Turkey	+	+	−	−

with the last question $\{\mathsf{Coastal}, \mathsf{GermanSpeaking}\} \to \{\mathsf{EUmember}\}$. The answer to this question is “yes” because the only countries that are both coastal and German-speaking (Germany and Belgium) are also EU members. Thus the GCI $\mathsf{Coastal} \sqcap \mathsf{GermanSpeaking} \sqsubseteq \mathsf{EUmember}$ is added to the TBox, and the completion process is finished. The completion yields the final context displayed in Table 3.

4 Improving the Usability of the Completion Procedure

Based on on the approach described in the previous section, we had implemented a first experimental version of a DL knowledge base completion tool. Our experiments with this tool showed that during completion the expert sometimes makes errors. For better usability of the completion procedure, it is important to support the expert in detecting and correcting these errors. Moreover, although we assume that the expert is omniscient (i.e., is potentially able to answer all implication questions), we have observed that it is convenient to be able to defer a question and answer it later. In the following, we discuss these problems in more detail and show how we address them in our improved completion tool.

Detecting Errors

We say that the expert makes an *error* if he extends the knowledge base such that it no longer has the underlying model $\mathcal{I}$ as its model. Since the procedure has no direct access to $\mathcal{I}$, in general it cannot detect such errors without help from the expert. The only case were the procedure can automatically detect that an error has occurred is when the knowledge base becomes inconsistent. Obviously, the underlying model $\mathcal{I}$ cannot be a model of an inconsistent KB.

However, when an inconsistency is detected by DL reasoning, then it is not clear at which stage the actual error was made. In fact, although only the last extension of the knowledge base has made it inconsistent, the deviation from what holds in $\mathcal{I}$ may have occurred in a previous step. Pinpointing [6,7,21,24,32] can be used to compute all minimal subsets of the knowledge base that are already inconsistent, and thus help the expert to find the place where the error was made. But in the end, the expert needs to tell the completion tool which are the erroneous assertions and/or GCIs.

The expert may also be alerted by DL reasoning to errors in cases where the knowledge base is not inconsistent. In fact, after each extension of the KB, the DL reasoner re-classifies it, i.e., computes the implied subsumption and instance relationships. If one of them contradicts the experts knowledge about $\mathcal{I}$, she also knows that an error has occurred. Again, pinpointing can show all minimal subsets of the knowledge base from which this unintended consequence already follows.

Recovery from Errors

Once the sources of the error are found, the next task is to correct it without producing unnecessary extra work for the expert. Of course, one can just go back to the step where the first error was made, and continue the completion process from there, this time with a correct answer. The problem with this simple approach is that it throws away all the information about $\mathcal{I}$ that the expert has added (by answering implication queries) after the first error had occurred. Consequently, the completion procedure may again pose implication queries whose answer actually follows from this information. On the DL side, it is no problem to keep those assertions and GCIs that really hold in $\mathcal{I}$. More precisely, the completion procedure can keep the GCIs and assertions for which the expert has stated that they are not erroneous. In fact, our completion procedure allows for arbitrary extensions of the KB as long as the KB stays a model of $\mathcal{I}$.

On the FCA side, it is less clear whether one can keep the implications that have been added after the first error had been made. In fact, since the new (correct) answer differs from the previous incorrect one, the completion procedure may actually produce different implication questions. The proof of correctness of the procedure, as given in [4], strongly depends on the fact that implications are enumerated according to the lectic ordering of their left-hand sides. Thus, having implications in the implication set for which the left-hand side is actually larger than the left-hand side of the implication currently under consideration may potentially cause problems. However, not using these implications may lead to more implication questions being generated. Fortunately, this problem can be solved by using these implications as background knowledge [34] rather than as part of the implication base to be generated. Thus, when correcting an error, we move implications generated after the error had occurred, but marked by the expert as correct, to the background knowledge. Correctness then follows from the fact that Stumme's extension of attribute exploration to the case of implicational background knowledge [34] can further be extended to partial contexts (see the corresponding subsection below).

Deferring Questions

Although we assume that the expert is omniscient in the sense that she is potentially able to answer all implication questions, it is convenient to be able to defer answering certain questions. For example, in the biology application mentioned in the introduction, answering an implication question may necessitate searching the literature on human protein phosphatases, querying gene and protein databases, or even making new experiments. This research may take quite some time, and can possibly be delegated to other researchers. It would thus be good if the expert could in parallel continue with the completion process.

Our approach for achieving this is that we allow the expert to stop the completion process and change the linear order on the set M of interesting concepts. This results in a different lectic order, and thus other implication questions may be asked before the problematic one.[2] To ensure that correctness is not compromised by this approach, we leave the knowledge base as it is, but move all the implications collected so far to the background knowledge. The completion procedure is then restarted with the first set that is closed w.r.t. the background implications, i.e., the closure of $\emptyset$.

Attribute Exploration with Background Knowledge for Partial Contexts

Our approaches for recovering from errors and for allowing the expert to defer answering a question depend on the use of implications as background knowledge. Attribute exploration in the presence of implicational background knowledge [34] and also non-implicational background knowledge [12,13] has already been considered in the literature for full contexts. For partial context, it has been considered in [17]. However, as mentioned before, this work is based on assumptions that are different from ours, and thus cannot directly be used.

In [4] we have shown termination and correctness of Algorithm 1 under the condition that the initial set of implications $\mathcal{L}_0$ is empty (line 3). In the following we show that Algorithm 1 stays correct and terminating if we modify it such that

1. the initial set of implications $\mathcal{L}_0$ already contains background implications that are not refuted by $\overline{\mathcal{K}}$, i.e., satisfies $\overline{\mathcal{K}} \subseteq Mod(\mathcal{L}_0)$ (line 3); and
2. the variable P is initialized with the lectically smallest $\mathcal{L}_0$-closed set, i.e., with $\mathcal{L}_0(\emptyset)$ rather than with $\emptyset$ (line 4).

Theorem 3. *Let M be a finite set of attributes, $\overline{\mathcal{K}}$ and $\mathcal{K}_0$ respectively a full and a partial context over the attributes in M such that $\mathcal{K}_0 \leq \overline{\mathcal{K}}$, $\mathcal{L}_0$ an initial set of background implications such that $\overline{\mathcal{K}} \subseteq Mod(\mathcal{L}_0)$, and P_0 the lectically smallest $\mathcal{L}_0$-closed set $\mathcal{L}_0(\emptyset)$. Then the modified Algorithm 1 terminates on this input and upon termination it outputs a partial context $\mathcal{K}$ and a set of implications $\mathcal{L}$ such that*

[2] Note, however, that this need not always be the case, i.e., it could be that also with the changed order the problematic question is the next one.

1. *$\mathcal{L}$ is a base for $Imp(\overline{\mathcal{K}})$, and*
2. *$\mathcal{K}$ refutes every implication that is refuted by $\overline{\mathcal{K}}$.*

Since the proof of this theorem is almost identical to the one of Theorem 1, we only give a sketch (see the proof of Theorem 1 given in [4] for details).

Termination is based on the following observation: in every iteration of the algorithm, one

(a) either considers as left-hand side of the current implication a new set P that is lectically larger than the previous one,
(b) or stays with the same left-hand side P, but decreases the cardinality of the right-hand side.

Thus, both iterations of the form (a) and (b) cannot occur infinitely often. This argument applies unchanged to the modified algorithm.

To show *correctness*, we must show 1. and 2. in the statement of the theorem. Thus, we must show that $\mathcal{L}$ is both sound and complete for $Imp(\overline{\mathcal{K}})$, and that $\mathcal{K}$ refutes every implication that is refuted by $\overline{\mathcal{K}}$. *Soundness* of $\mathcal{L}$ is an immediate consequence of the fact that the invariant $\mathcal{K} \leq \overline{\mathcal{K}} \subseteq Mod(\mathcal{L})$ holds throughout the run of the algorithm. The only difference between the original and the modified algorithm is that the former starts with the empty set of implications whereas the latter starts with a possibly non-empty set $\mathcal{L}_0$ of implications. However, since $\mathcal{L}_0$ is assumed to satisfy $\overline{\mathcal{K}} \subseteq Mod(\mathcal{L}_0)$, the invariant is still satisfied at the start of the modified algorithm.

Because the invariant is satisfied, completeness of $\mathcal{L}$ for $Imp(\overline{\mathcal{K}})$ as well as the fact that $\mathcal{K}$ refutes every implication refuted by $\overline{\mathcal{K}}$ follow by Proposition 2 as soon as we have shown that every implication is decided w.r.t. $\mathcal{K}$ and $\mathcal{L}$. Thus, assume that $L \rightarrow R$ is undecided w.r.t. $\mathcal{K}$ and $\mathcal{L}$, i.e., it does not follow from $\mathcal{L}$ and is not refuted by $\mathcal{K}$. By Proposition 3, $\mathcal{L}(L) \rightarrow R$ also does not follow from $\mathcal{L}$. In addition, since $L \subseteq \mathcal{L}(L)$, it is also not refuted by $\mathcal{K}$.

We claim that $\mathcal{L}(L)$ is equal to one of the sets P_i considered during the run of the algorithm. In fact, since the final set $P_n = M$ is the lectically largest subset of M and since the lectic order $<$ is total, there is a unique smallest i such that $\mathcal{L}(L) < P_i$. First, assume that $i = 0$. Then $\mathcal{L}(L) < P_0 = \mathcal{L}_0(\emptyset)$. However, $\mathcal{L}_0 \subseteq \mathcal{L}$ implies that $\mathcal{L}(L)$ is as also $\mathcal{L}_0$-closed, which contradicts the fact that $\mathcal{L}_0(\emptyset)$ is the smallest $\mathcal{L}_0$-closed set. If $i > 0$, then $P_{i-1} < L < P_i$, and we can analogously derive a contradiction to the fact the P_i is the lectically next $\mathcal{L}_i$-closed set after P_{i-1}.

Thus, let i be such that $\mathcal{L}(L) = P_i$. Then the implication $P_i \rightarrow \mathcal{K}_i(P_i)$ is considered during iteration i of the algorithm. If this implication is not refuted by $\overline{\mathcal{K}}$, then one can show that $R \subseteq \mathcal{K}_i(P_i)$, and use this fact to show that $P_i \rightarrow R$ follows from $\mathcal{L}$, which contradicts our assumption that $L \rightarrow R$ is undecided (see [4] for details).

If $P_i \rightarrow \mathcal{K}_i(P_i)$ is refuted by $\overline{\mathcal{K}}$, then $\mathcal{K}_i$ is extended to a partial context $\mathcal{K}_{i+1}$ that refutes this implication. If $\mathcal{K}_{i+1}$ also refutes $P_i \rightarrow R$, then this implies that $\mathcal{K}$ refutes $L \rightarrow R$, which is again a contradiction (see [4] for details). Otherwise, note that $P_{i+1} = P_i$ and $\mathcal{L}_{i+1} = \mathcal{L}_i$, and thus in the next iteration the expert gets the implication $P_i \rightarrow \mathcal{K}_{i+1}(P_i)$. By our assumption, $P_i \rightarrow R$ is not refuted by $\mathcal{K}_{i+1}$, and thus $R \subseteq K_{i+1}(P_i)$. In addition, we have $\mathcal{K}_{i+1}(P_i) \subsetneq \mathcal{K}_i(P_i)$ due to the fact that $\mathcal{K}_{i+1}$ refutes $P_i \rightarrow \mathcal{K}_i(P_i)$.

If $P_i \rightarrow \mathcal{K}_{i+1}(P_i)$ is not refuted by $\overline{\mathcal{K}}$, then we can continue as in the first case above, and derive that $P_i \rightarrow R$ follows from $\mathcal{L}$. Otherwise, we can continue as in the second case. However, because in this case the size of the right-hand side of the implication given to the expert strictly decreases, we cannot indefinitely get the second case. This completes our sketch of the proof of Theorem 3.

Termination and correctness of the accordingly modified Algorithm 2 is a trivial consequence of this theorem, i.e., we can also start this algorithm with a non-empty set of background implications $\mathcal{L}_0$ provided that all implications in $\mathcal{L}_0$ are not refuted by $\mathcal{I}$.

5 OntoComP: Ontology Completion Plugin for Protégé

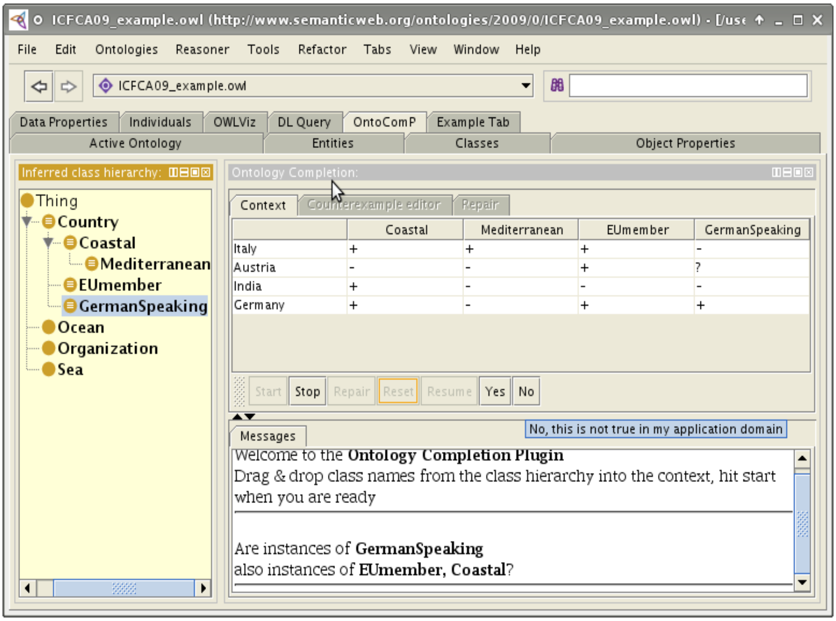
Based on the usability considerations sketched in the previous sections, we have implemented an improved version of our completion algorithm as an open-source tool called OntoComP,[3] which stands for ONTOlogy COMpletion Plugin. It is written in Java as a plugin for the Protégé 4 ontology editor [19]. It communicates with the underlying DL reasoner over the OWL API [8].

OntoComP can easily be integrated into an existing Protégé 4 installation. After installing this plugin, it appears in the Protégé 4 window as a new tab. For completing a knowledge base loaded into Protégé 4, one first needs to classify it with one of the DL reasoners supported by Protégé 4 (e.g., FaCT++ [35] or Pellet [33]). Then one can go to the OntoComP tab to start completion, and create the set M of interesting concepts by dragging and dropping the concept names that are supposed to be in this set from the class hierarchy displayed in the OntoComP tab. Figure 5 displays the OntoComP window during completion of the knowledge base of Example 3.

Counterexample Generation. OntoComP has a counterexample editor for supporting the user during counterexample generation. When the user rejects an implication, OntoComP opens a counterexample editor in a new tab, and displays those individuals from the ABox that can potentially be turned into a counterexample by adding assertions for them. Alternatively, the user can introduce a new individual together with assertions that make it a counterexample. During counterexample generation, OntoComP guides the user and notifies her once she has created a valid counterexample to the implication question.

Error Recovery. At any point during knowledge base completion, if the user notices that he has made a mistake in one of the previous steps, he can stop the completion and can request to see the history of all answers he has given. OntoComP displays all the questions asked in previous steps, the answers that were

[3] available under `http://code.google.com/p/ontocomp`



	Coastal	Mediterranean	EUmember	GermanSpeaking
Italy	+	+	+	-
Austria	-	-	+	?
India	+	-	-	-
Germany	+	-	+	+

No, this is not true in my application domain

Welcome to the **Ontology Completion Plugin**
Drag & drop class names from the class hierarchy into the context, hit start when you are ready

Are instances of **GermanSpeaking**
also instances of **EUmember, Coastal**?

Fig. 1. OntoComP window during completion

given to these questions, and the counterexamples accompanying negative answers. The user can browse the answering history, correct the wrong answers he has given in the previous steps, and can then continue completion. OntoComP then keeps all GCIs and counterexamples that were not marked as incorrect in the knowledge base, and moves all implications to the background knowledge. Pinpointing reasons for consequences (such as inconsistency or unintended subsumption or instance relationships) is not directly integrated into the current version of OntoComP. However, the user could use the pinpointing facilities provided by Protégé 4.

Deferring Questions. OntoComP allows the user to defer a question at any point during completion. It achieves this by the approach described in Section 4, i.e., it tries to change the order on M such that a different question is generated as the next question. As already mentioned in Section 4, this need not always succeed.

6 Future Work

In addition to further improving and evaluating our completion tool OntoComP, the main topic for future research in this direction will be to look at extensions of our definition of a complete KB. As a formalization of what "all relationships between interesting concepts" really means, we have used subsumption relationships between conjunctions of elements of a finite set of interesting concepts M. One could also consider more complex relationships by fixing a specific DL $\mathcal{D}$, and then taking, as attributes, all the $\mathcal{D}$-concept descriptions that can be built using a finite set of interesting concept and role names. This would result in a notion of completeness where each GCIs that can be built using $\mathcal{D}$ and the given finite set of concept and role names either follows from the TBox (in case it holds in the underlying model) or is refuted by the ABox (in case it does not hold in the underlying model).

The obvious problem caused by this extension is that, in general, the set of attributes becomes infinite, and thus termination of the exploration process is no longer a priori guaranteed. Different extensions of classical attribute exploration (i.e., for full contexts) in this direction are described in [30,31] for the DL $\mathcal{FLE}$, and in [2,3] for the DL $\mathcal{EL}$ and its extension by cyclic concept definitions with greatest fixpoint semantics, $\mathcal{EL}_{\mathrm{gfp}}$. In both approaches, variants of classical attribute exploration are introduced that consider as attributes all concept descriptions built using the given DL and a given finite set of concept and role names. It is shown that the introduced exploration algorithms terminate if the underlying model is finite. We will investigate whether these approaches can be adapted to knowledge base completion.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, Cambridge (2003)
2. Baader, F., Distel, F.: A finite basis for the set of $\mathcal{EL}$-implications holding in a finite model. In: Medina, R., Obiedkov, S. (eds.) ICFCA 2008. LNCS (LNAI), vol. 4933, pp. 46–61. Springer, Heidelberg (2008)
3. Baader, F., Distel, F.: Exploring finite models in the description logic $\mathcal{EL}_{\mathrm{gfp}}$. In: Ferré, S., Rudolph, S. (eds.) ICFCA 2009. LNCS (LNAI), vol. 5548, Springer, Heidelberg (2009)
4. Baader, F., Ganter, B., Sattler, U., Sertkaya, B.: Completing description logic knowledge bases using formal concept analysis. LTCS-Report LTCS-06-02, Chair for Automata Theory, Institute for Theoretical Computer Science, Dresden University of Technology, Germany (2006), `http://lat.inf.tu-dresden.de/research/reports.html`
5. Baader, F., Ganter, B., Sertkaya, B., Sattler, U.: Completing description logic knowledge bases using formal concept analysis. In: Proc. of the Twentieth Int. Joint Conf. on Artificial Intelligence (IJCAI 2007), pp. 230–235. AAAI Press, Menlo Park (2007)
6. Baader, F., Peñaloza, R., Suntisrivaraporn, B.: Pinpointing in the description logic $\mathcal{EL}^+$. In: Hertzberg, J., Beetz, M., Englert, R. (eds.) KI 2007. LNCS (LNAI), vol. 4667, pp. 52–67. Springer, Heidelberg (2007)
7. Baader, F., Suntisrivaraporn, B.: Debugging SNOMED CT using axiom pinpointing in the description logic $\mathcal{EL}^+$. In: Proc. of the Int. Conf. on Representing and Sharing Knowledge Using SNOMED (KR-MED 2008), Phoenix, Arizona (2008)
8. Bechhofer, S., Volz, R., Lord, P.: Cooking the semantic web with the OWL API. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 659–675. Springer, Heidelberg (2003)

9. Burmeister, P., Holzer, R.: On the treatment of incomplete knowledge in formal concept analysis. In: Ganter, B., Mineau, G.W. (eds.) ICCS 2000. LNCS, vol. 1867, pp. 385–398. Springer, Heidelberg (2000)
10. Burmeister, P., Holzer, R.: Treating incomplete knowledge in formal concept analysis. In: Ganter, B., Stumme, G., Wille, R. (eds.) Formal Concept Analysis. LNCS, vol. 3626, pp. 114–126. Springer, Heidelberg (2005)
11. Ganter, B.: Two basic algorithms in concept analysis. Technical Report Preprint-Nr. 831, Technische Hochschule Darmstadt, Darmstadt, Germany (1984)
12. Ganter, B.: Attribute exploration with background knowledge. Theoretical Computer Science 217(2), 215–233 (1999)
13. Ganter, B., Krauße, R.: Pseudo-models and propositional Horn inference. Discrete Applied Mathematics 147(1), 43–55 (2005)
14. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin (1999)
15. Haarslev, V., Möller, R.: RACER system description. In: Goré, R.P., Leitsch, A., Nipkow, T. (eds.) IJCAR 2001. LNCS (LNAI), vol. 2083, pp. 701–705. Springer, Heidelberg (2001)
16. Halaschek-Wiener, C., Parsia, B., Sirin, E., Kalyanpur, A.: Description Logic reasoning for dynamic ABoxes. In: Proc. of the 19th Int. Workshop on Description Logics (DL 2006). CEUR-WS, vol. 189 (2006)
17. Holzer, R.: Knowledge acquisition under incomplete knowledge using methods from formal concept analysis: Part I. Fundamenta Informaticae 63(1), 17–39 (2004)
18. Holzer, R.: Knowledge acquisition under incomplete knowledge using methods from formal concept analysis: Part II. Fundamenta Informaticae 63(1), 41–63 (2004)
19. Horridge, M., Tsarkov, D., Redmond, T.: Supporting early adoption of OWL 1.1 with Protege-OWL and FaCT++. In: Proc. of the Second Int. Workshop OWL: Experiences and Directions (OWLED 2006). CEUR-WS (2006)
20. Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From SHIQ and RDF to OWL: the making of a web ontology language. Journal of Web Semantics 1(1), 7–26 (2003)
21. Kalyanpur, A., Parsia, B., Sirin, E., Grau, B.C.: Repairing unsatisfiable concepts in OWL ontologies. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 170–184. Springer, Heidelberg (2006)
22. Kalyanpur, A., Parsia, B., Sirin, E., Grau, B.C., Hendler, J.A.: Swoop: A web ontology editing browser. Journal of Web Semantics 4(2), 144–153 (2006)
23. Knublauch, H., Fergerson, R.W., Noy, N.F., Musen, M.A.: The Protégé OWL plugin: An open development environment for semantic web applications. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 229–243. Springer, Heidelberg (2004)
24. Meyer, T., Lee, K., Booth, R., Pan, J.Z.: Finding maximally satisfiable terminologies for the description logic $\mathcal{ALC}$. In: Proc. of the 21st National Conf. on Artificial Intelligence (AAAI 2006), pp. 269–274. AAAI Press/The MIT Press (2006)
25. Motik, B.: Reasoning in Description Logics using Resolution and Deductive Databases. Ph.D. Dissertation, Universität Karlsruhe (TH), Germany (2006)
26. Motik, B., Shearer, R., Horrocks, I.: Optimized reasoning in description logics using hypertableaux. In: Pfenning, F. (ed.) CADE 2007. LNCS (LNAI), vol. 4603, pp. 67–83. Springer, Heidelberg (2007)
27. Oberle, D., Volz, R., Staab, S., Motik, B.: An extensible ontology software environment. In: Handbook on Ontologies, Int. Handbooks on Information Systems, pp. 299–320. Springer, Heidelberg (2004)

28. Obiedkov, S.A.: Modal logic for evaluating formulas in incomplete contexts. In: Priss, U., Corbett, D.R., Angelova, G. (eds.) ICCS 2002. LNCS, vol. 2393, pp. 314–325. Springer, Heidelberg (2002)
29. Reeg, S., Weiß, W.: Properties of Finite Lattices. Diplomarbeit, TH Darmstadt, Germany (1990)
30. Rudolph, S.: Exploring relational structures via $\mathcal{FLE}$. In: Wolff, K.E., Pfeiffer, H.D., Delugach, H.S. (eds.) ICCS 2004. LNCS, vol. 3127, pp. 196–212. Springer, Heidelberg (2004)
31. Rudolph, S.: Relational Exploration: Combining Description Logics and Formal Concept Analysis for Knowledge Specification. Ph.D. Dissertation, Fakultät Mathematik und Naturwissenschaften, TU Dresden, Germany (2006)
32. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Proc. of the Eighteenth Int. Joint Conf. on Artificial Intelligence (IJCAI 2003), pp. 355–362. Morgan Kaufmann, San Francisco (2003)
33. Sirin, E., Parsia, B.: Pellet: An OWL DL reasoner. In: Proc. of the 2004 Int. Workshop on Description Logics (DL 2004). CEUR Workshop Proc., vol. 104. CEUR-WS.org (2004)
34. Stumme, G.: Attribute exploration with background implications and exceptions. In: Data Analysis and Information Systems. Statistical and Conceptual approaches. Proc. of GfKl 1995. Studies in Classification, Data Analysis, and Knowledge Organization, vol. 7, pp. 457–469. Springer, Heidelberg (1996)
35. Tsarkov, D., Horrocks, I.: FaCT++ description logic reasoner: System description. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 292–297. Springer, Heidelberg (2006)
36. Wille, R.: Restructuring lattice theory: An approach based on hierarchies of concepts. In: Ordered Sets, pp. 445–470. Reidel, Dordrecht (1982)
37. Wolstencroft, K., Brass, A., Horrocks, I., Lord, P.W., Sattler, U., Turi, D., Stevens, R.: A little semantic web goes a long way in biology. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 786–800. Springer, Heidelberg (2005)

Concept Lattice Orbifolds – First Steps

Daniel Borchmann and Bernhard Ganter

Institut für Algebra
Technische Universität Dresden

Abstract. Concept lattices with symmetries may be simplified by "folding" them along the orbits of their automorphism group. The resulting diagram is often more intuitive than the full lattice diagram, but well defined annotations are required to make the folded diagram as informative as the original one. The folding procedure can be extended to formal contexts.

A typical situation where such lattice foldings are useful is when hierarchies of structures are considered "up to isomorphisms".

1 Introduction

Much effort has been made to develop techniques for handling large and complex concept lattices. For lattices built from real-word data, methods allowing for aspects and views of the lattice are often a good choice. In more mathematical situations, a different strategy is promising: using symmetry. The lattice of, say, all quasi-orders on a fixed base set, or the lattice of all subgroups of a given group, are structures with rich automorphism groups, and it is advisable to use these for simplification.

The basic elements of theory for such investigations do already exist. They were invented in Monika Zickwolff's thesis of 1991 [8], and were tailored for applications in *rule exploration*, a generalisation of attribute exploration to first order predicate logic. First algorithmic results were also obtained at that time [5]. Since then, little progress has been made, presumably because the methods tend to be difficult.

In recent years we have often met situations in which the application of symmetry techniques would have been appropriate. We see a growing demand for a solid and intuitive theory. In the present paper, we give an introduction to the basic ideas, mainly by means of an example. Most of the results presented here are already contained in Zickwolff's work, but in a very condensed form. Our aim is to make them better accessible to the FCA community.

Any lattice or ordered set with automorphisms (in fact, any relational structure) can be "folded" in such a manner that the orbits of the group become the elements of a new structure. However, in order not to loose information, this "orbifold" needs to be carefully annotated, so that the original structure can be reconstructed.

Formal contexts can be folded as well, and it is possible to compute concept lattice orbifolds from context orbifolds. Such computations require a combination of lattice and group algorithms and are not easy to handle.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 22–37, 2009.

The present paper concentrates on folding orders and lattices. In Section 5 we sketch a first example of a context orbifold. The details are to be treated in a subsequent paper.

2 Group Annotated Ordered Sets

Definition 1. (Group annotated ordered set): Let $\underline{P} := (P, \leq)$ be an ordered set and let $\underline{G} := (G, \circ)$ be some group. A mapping

$$\lambda : P \times P \to \mathcal{P}(G)$$

is called a **$\underline{G}$-annotation** of $\underline{P}$ iff

1. $\lambda(a, b) \neq \emptyset$ if and only if $a \leq b$ in P,
2. each set $\lambda(a, a)$, $a \in P$, is a subgroup G_a of $\underline{G}$, and
3. $\lambda(a, b) \circ \lambda(b, c) \subseteq \lambda(a, c)$ for all $a \leq b \leq c$ in P.

$(P, \leq, \lambda)$ is then called a **$\underline{G}$-annotated ordered set**. $\Diamond$

The following example, though small, seems complicated at first. In the sequel we shall introduce techniques easing readability. Moreover, it will be shown where the example comes from.

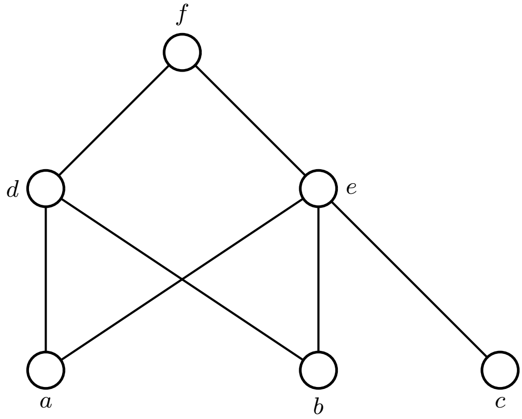


Fig. 1. A small ordered set

Example 1. Let $\underline{P} := (\{a, b, c, d, e, f\}, \leq)$ be the six-element ordered set depicted in Figure 1. Let G be the alternating group on the four element set $\{1, 2, 3, 4\}$, i.e., the group of all even permutations of these elements. Table 1 gives an annotation map.

Giving an annotation for an ordered set by means of a full table, as it was done in Table 1, is informative but unpleasant to read. We therefore introduce a simplified notation based on double cosets of subgroups.

Table 1. An A_4-annotation of the ordered set in Figure 1

$\lambda(a,a)=$	$\{id,(12)(34)\}$
$\lambda(b,b)=$	$\{id,(12)(34)\}$
$\lambda(c,c)=$	$\{id,(234),(243)\}$
$\lambda(d,d)=$	$\{id,(12)(34),(13)(24),(14)(23)\}$
$\lambda(e,e)=$	$\{id\}$
$\lambda(f,f)=$	$\{id,(13)(24)\}$
$\lambda(a,d)=$	$\{(124),(132),(143),(234)\}$
$\lambda(a,e)=$	$\{(132),(143)\}$
$\lambda(a,f)=$	$\{(234),(243),(123),(132),(124),(143)\}$
$\lambda(b,d)=$	$\{id,(12)(34),(13)(24),(14)(23)\}$
$\lambda(b,e)=$	$\{(134),(142)\}$
$\lambda(b,f)=$	$\{id,(12)(34),(13)(24),(14)(23),(123),(243)\}$
$\lambda(c,e)=$	$\{(12)(34),(132),(142)\}$
$\lambda(c,f)=$	$\{id,(243),(234),(123),(13)(24),(143)\}$
$\lambda(d,f)=$	$\{id,(12)(34),(13)(24),(14)(23)\}$
$\lambda(e,f)=$	$\{(12)(34),(14)(23),(124),(132)\}$

It is immediate from the definition that for each pair $a \leq b$ in an annotated ordered set the set $\lambda(a,b)$ is a union of double cosets of the "stabiliser" subgroups $G_a := \lambda(a,a)$ and $G_b := \lambda(b,b)$, i.e., that

$$\lambda(a,a) \circ \lambda(a,b) \circ \lambda(b,b) = \lambda(a,b).$$

Since the double cosets of any subgroup pair partition the group, it suffices to give a system of representatives of these double cosets. Moreover, since

$$\lambda(a,c) \circ \lambda(c,b)$$

also is a union of double cosets, we may simplify further and and define as follows:

Definition 2. Let λ be a $\underline{G}$-annotation of an ordered set $\underline{P}$. A **simplified annotation** $\lambda_\bullet$ corresponding to λ gives for every pair $a \leq b$ in P a set of double coset representatives of

$$\lambda(a,b) \setminus \bigcup_{a<c<b} \lambda(a,c) \circ \lambda(c,b).$$

$\diamondsuit$

Note that $\lambda_\bullet(a,b)$ may be empty. As a convention, such pairs will be omitted in our listings of $\lambda_\bullet$. Similarly, we shall not list neighbouring pairs $a \prec b$ for which $\lambda_\bullet(a,b)$ consists only of the neutral element of $\underline{G}$. Following this, Table 1 simplifies to, e.g., the data displayed in Table 2.

The ordered set in Figure 1 can be interpreted as a set of graphs on four vertices $\{1,2,3,4\}$, ordered by embeddability, see Figure 2. Each connected graph with four vertices, with the exception of the complete graph, occurs exactly once up to *even* isomorphism, which means that each such graph occurs exactly once,

Table 2. The simplified annotation to Table 1

$\lambda_\bullet(a,a)=$	$\{id,(12)(34)\}$
$\lambda_\bullet(b,b)=$	$\{id,(12)(34)\}$
$\lambda_\bullet(c,c)=$	$\{id,(234),(243)\}$
$\lambda_\bullet(d,d)=$	$\{id,(12)(34),(13)(24),(14)(23)\}$
$\lambda_\bullet(e,e)=$	$\{id\}$
$\lambda_\bullet(f,f)=$	$\{id,(13)(24)\}$
$\lambda_\bullet(a,d)=$	$\{(234)\}$
$\lambda_\bullet(a,e)=$	$\{(132)\}$
$\lambda_\bullet(b,e)=$	$\{(134)\}$
$\lambda_\bullet(c,e)=$	$\{(12)(34)\}$
$\lambda_\bullet(e,f)=$	$\{(12)(34),(124)\}$

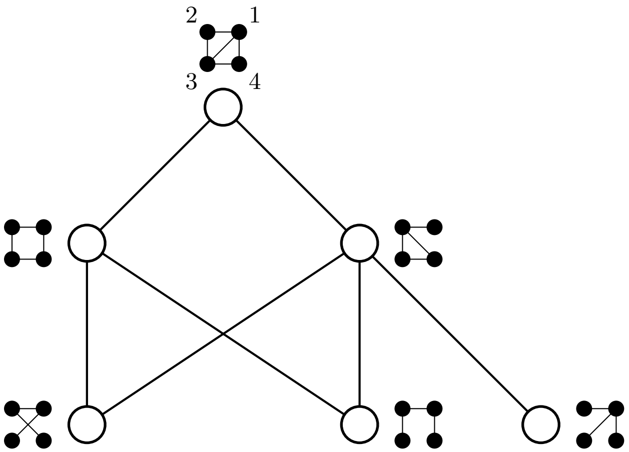


Fig. 2. The ordered set from Figure 1, labelled by graphs

except for the path, which occurs twice. The four-element path has only even automorphisms and is for this reason listed with two copies.

The small graphs in Figure 2 are all labelled in the manner indicated for the top element: counterclockwise, starting with the upper right vertex. The annotation listed in Table 1 can now be read off from this diagram. Then recall that the group under consideration is the alternating group, acting on these vertices. The annotation is obtained as follows:

1. For each $p \in P$, the annotation $\lambda(p,p)$ is simply the automorphism group of the labelling graph, allowing only permutations from the alternating group, i.e., only even permutations.
2. For $p < q$ in P, the set $\lambda(p,q)$ consists of all even permutations γ for which γ^{-1} is an embedding from p into q. (In other words: for which p is a subset of γq.)

Note that the second condition includes the first one if we allow $p = q$.

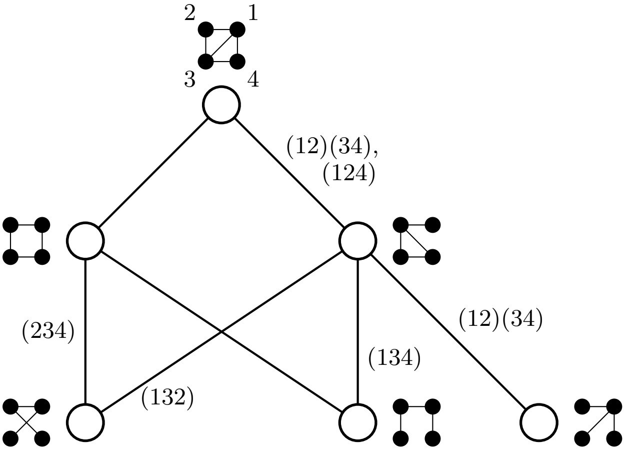
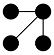
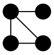
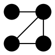
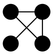
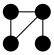


Fig. 3. Simplified annotation of the diagram. The labels are double coset representatives. The stabiliser $\lambda(p,p)$ is the automorphism group of the graph labelling p, restricted to the alternating group A_4. Non-neighbouring pairs are not drawn because their simplified labels are empty (in this example).

In small examples the simplified annotation can be written directly to the diagram, in particular when the stabiliser groups $\lambda(p,p)$ can be read off from the labelling. This is shown in Figure 3.

On the example of the pair $c \prec e$ we explain how to read the diagram in Figure 3:

- Point c is labelled by the graph , point e by .
- The graph at c is embeddable into the graph at e, but the given diagram is not a subdiagram. The graph at c is a subgraph of several isomorphic copies of the graph at e.
- There are precisely three isomorphic copies (all obtained by even permutations) of the graph at e that contain the diagram at point c, these are: , , and .
- These copies are obtained from the original label by the even permutations (12)(34), (132), and (142). These constitute the annotation $\lambda(c,e)$, cf. Table 1.
- The simplified annotation lists only (12)(34), because the other two permutations can be obtained from (12)(34) using automorphisms from the stabiliser groups. For example

$$(132) = (234) \circ (12)(34) \circ id,$$

where $(132) \in \lambda(c,c)$, $(12)(34) \in \lambda_{\bullet}(c,e)$, and $id \in \lambda(e,e)$.

3 Folding Orders and Lattices

Figure 3 gives a clue what annotation maps are used for. The six-element ordered set shown there represents a much larger order, having 37 elements. These are the connected graphs on $\{1,2,3,4\}$ (K_4 omitted). The smaller ordered set is obtained through folding the larger one: Isomorphic graphs are identified. The induced folding of the order relation is expressed by the annotation map. A general formulation is provided by the next definition.

Definition 3. Let $\underline{P} := (P, \leq_P)$ be an ordered set and let $\Gamma \leq \mathrm{Aut}(\underline{P})$ be a subgroup of its automorphism group. A **Γ-orbifold** of $\underline{P}$ (also called an **order transversal**) is a triple

$$(P \setminus\setminus \Gamma, \leq, \lambda),$$

where

- $P \setminus\setminus \Gamma := \{p^\Gamma \mid p \in P\}$ is the set of orbits of Γ on P,
- $\leq$ is the order relation defined on $P \setminus\setminus \Gamma$ by
$$p^\Gamma \leq q^\Gamma :\Longleftrightarrow \exists_{\gamma \in \Gamma}\, p \leq_P \gamma q,$$
- and the mapping
$$\lambda : (P \setminus\setminus \Gamma) \times (P \setminus\setminus \Gamma) \to \mathfrak{P}(\Gamma)$$
is defined using some fixed system Y of representatives of $P \setminus\setminus \Gamma$ by
$$\lambda(a^\Gamma, b^\Gamma) := \{\gamma \in \Gamma \mid a \leq_p \gamma b\},$$
where $a, b \in Y$.

If $\underline{P}$ is a lattice, we speak of a **lattice orbifold**. ◊

Some details of this definition require justification. For example, it must be argued that $\leq$ is well defined and indeed an order. We include this in the proof of the following lemma.

Lemma 1. Let $\underline{P} := (P, \leq_P)$ be an ordered set and let $\Gamma \leq \mathrm{Aut}(\underline{P})$ be a subgroup of its automorphism group. Then every orbifold of $\underline{P}$ is a Γ-annotated ordered set.

Proof. We first show that $\leq$, defined by

$$p^\Gamma \leq q^\Gamma :\Longleftrightarrow \exists_{\gamma \in \Gamma}\, p \leq_P \gamma q,$$

is well defined, i.e., independent of the choice of the representatives p, q. For representatives

$$p_1 \in p^\Gamma, q_1 \in q^\Gamma$$

of the same orbits we find automorphisms $\alpha, \beta \in \Gamma$ such that $p_1 = \alpha p$ and $q_1 = \beta q$. Then

$$\begin{aligned} p \leq_P \gamma q &\Longleftrightarrow \alpha p \leq_P \alpha\gamma q \\ &\Longleftrightarrow \alpha p \leq_P \alpha\gamma\beta^{-1}\beta q \\ &\Longleftrightarrow p_1 \leq_P \gamma_1 q_1, \text{ where } \gamma_1 = \alpha\gamma\beta^{-1}. \end{aligned}$$

Thus $\exists_\gamma\, p \leq_P \gamma q \iff \exists \gamma_1\, p_1 \leq_P q_1$, as desired.

The rest of the proof is straightforward: $\leq$ obviously is an order on $P \setminus\!\setminus \mathit{\Gamma}$ and λ is an annotation map. That $\lambda(a, b)$ is nonempty for $a \leq b$ is immediate from the definition of $\leq$. Clearly

$$\lambda(a, a) = \{\gamma \mid a \leq \gamma a\} = \{\gamma \mid a = \gamma a\}$$

is a subgroup of $\mathit{\Gamma}$, it is the stabiliser $\mathit{\Gamma}_a$ of a in $\mathit{\Gamma}$. For the third condition we obtain

$$\begin{aligned}\lambda(a, b) \circ \lambda(b, c) &= \{\alpha \mid a \leq_P \alpha b\} \circ \{\beta \mid b \leq_P \beta c\}\\ &= \{\alpha \circ \beta \mid a \leq_P \alpha b, b \leq_P \beta c\}\\ &\subseteq \{\gamma \mid a \leq_P \gamma c\}\\ &= \lambda(a, c).\end{aligned}$$

□

Now that we are able to fold ordered sets we also would like to unfold them in a way that reconstructs the original order. This is provided by the next definition.

Definition 4. Let $(P, \leq, \lambda)$ be a G-annotated ordered set, and let $G_p := \lambda(p, p)$ for all $p \in P$. The **unfolding** (or **reconstruction**) of $(P, \leq, \lambda)$ is defined as

$$\operatorname{rec}(P, \leq, \lambda) := (\dot\bigcup_{p \in P} G/G_p, \leq_r),$$

with

$$gG_p \leq_r hG_q :\Longleftrightarrow g^{-1}h \in \lambda(p, q).$$

◊

Proposition 1. The unfolding $\operatorname{rec}(P, \leq, \lambda)$ of a G-annotated ordered set $(P, \leq, \lambda)$ is an ordered set having a group of automorphisms isomorphic to G.

Proof. Let

$$N := \dot\bigcup_{p \in P} G/G_p = \dot\bigcup \{gG_p \mid g \in G\}$$

be the set of all stabiliser cosets. Proving that $\leq_r$ is an order on N is easy: Clearly $\leq_r$ is reflexive, since $id \in \lambda(p, p)$. Antisymmetry follows from the fact that for $p \neq q$ at least one of the sets $\lambda(p, q)$ must be empty, and transitivity follows from the multiplicativity condition for annotation maps.

G operates on its power set as a permutation group $\mathit{\Gamma}$ of left multiplications. Let $\phi : G \to \mathit{\Gamma}$ denote the canonical isomorphism. Each $\phi(h) \in \mathit{\Gamma}$ maps N to N by

$$gG_p \overset{\phi(h)}{\mapsto} hgG_p,$$

and

$$\begin{aligned}g_1G_p \leq_r g_2G_q &\Longleftrightarrow g_1^{-1}g_2 \in \lambda(p, q)\\ &\Longleftrightarrow g_1^{-1}h^{-1}hg_2 \in \lambda(p, q)\\ &\Longleftrightarrow (hg_1)^{-1}hg_2 \in \lambda(p, q)\\ &\Longleftrightarrow hg_1G_p \leq_r hg_2G_q.\end{aligned}$$

Therefore each $\phi(h) \in \mathit{\Gamma}$ acts as an automorphism on $(N, \leq_r)$, and $\mathit{\Gamma}$ is a subgroup of $\operatorname{Aut}(N, \leq_r)$. □

4 Isomorphisms of Annotated Ordered Sets

The annotation we have studied above depends on the choice of representatives for the isomorphism classes of graphs. If we choose other representatives, we obtain another annotation map. If we are lucky, the new annotation may be considerably simpler. An example is shown in Figure 4.

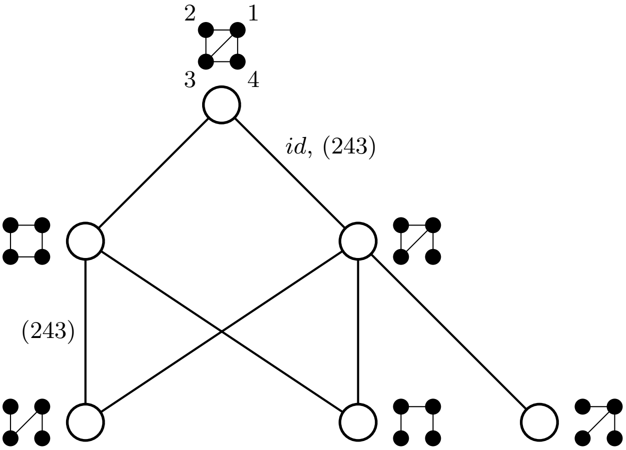


Fig. 4. An alternative simplified annotation obtained by using isomorphic graph diagrams.

The two diagrams in Figures 3 and 4 represent the same situation, and should be called isomorphic. They however differ considerably. It is not surprising that a rather complicated notion of isomorphy is needed.

Definition 5 (Isomorphy of group-annotated ordered sets). Let Γ_1 and Γ_2 be groups, and let $\underline{P}_1 = (P_1, \leq_1, \lambda_1)$ be a Γ_1-annotated ordered set and $\underline{P}_2 = (P_2, \leq_2, \lambda_2)$ a Γ_2-annotated ordered set. Then $\underline{P}_1$ and $\underline{P}_2$ are said to be **isomorphic** if the following conditions hold:

- there exists an order isomorphism $\alpha : (P_1, \leq_1) \longrightarrow (P_2, \leq_2)$ and
- there exists a group isomorphism $\delta : \Gamma_1 \longrightarrow \Gamma_2$ and
- there exists a mapping $\phi : P_1 \longrightarrow \Gamma_2$

such that

$$\delta[\lambda_1(a, b)] = \phi(a)^{-1}\lambda_2(\alpha a, \alpha b)\phi(b)$$

holds for all $a \leq b$ in P_1. ◊

The two annotations in Figures 3 and 4 are indeed isomorphic according to this definition. The two groups are identical, so that we may choose δ to be the identity map. Two orders are canonically isomorphic and isomorphic to the order in Figure 1, so that we can omit α and simply use the letters from Figure 1 as element names for both. It remains to find a mapping $\phi : P \to A_4$ such that

$$\lambda_1(a,b) = \phi(a)^{-1}\lambda_2(a,b)\phi(b)$$

holds for all $a \leq b$ in P. For this, we may take

x	a	b	c	d	e	f
$\phi(x)$	(132)	id	id	id	(142)	id

For example, according to Table 1 we have $\lambda_1(c,e) = \{(12)(34),(132),(142)\}$, and from Figure 4 we read off that

$$\lambda_2(c,e) = \lambda_2(c,c) \circ \{id\} \circ \lambda_2(e,e)$$

(recall that a missing edge label stand for $\{id\}$). We conclude that

$$\begin{aligned}\lambda_2(c,e) &= \lambda_2(c,c) \circ \{id\} \circ \lambda_2(e,e)\\ &= \{id,(234),(243)\} \circ \{id\}\\ &= \{id,(234),(243)\}.\end{aligned}$$

Therefore

$$\begin{aligned}\phi(c)^{-1} \circ \lambda_2(c,e) \circ \phi(e) &= id \circ \lambda_2(c,e) \circ (142)\\ &= \{id,(234),(243)\} \circ (142)\\ &= \{(142),(12)(34),(132)\},\end{aligned}$$

which is indeed $\lambda_1(c,e)$, as can be seen from Table 1.

Our first theorem states that the two structure necessarily are isomorphic.

Theorem 1. Any two $\varGamma$-orbifolds of an ordered set $\underline{P}$ are isomorphic. More generally, if $\underline{P_1}$ and $\underline{P_2}$ are isomorphic ordered sets, $\alpha : P_1 \to P_2$ is an isomorphism and $\varGamma_1 \leq \mathrm{Aut}(\underline{P_1})$ and $\varGamma_2 \leq \mathrm{Aut}(\underline{P_2})$ are groups of automorphisms such that

$$\varGamma_2 = \alpha \circ \varGamma_1 \circ \alpha^{-1},$$

then each $\varGamma_1$-orbifold of $\underline{P_1}$ is isomorphic to each $\varGamma_2$-orbifold of $\underline{P_2}$.

Proof. We only prove the special case. A proof of the general statement can be found in [8]. Let

$$(P \setminus\!\!\setminus \varGamma, \leq, \lambda_1) \quad \text{and} \quad (P \setminus\!\!\setminus \varGamma, \leq, \lambda_2)$$

be two $\varGamma$-orbifolds of $\underline{P}$ and let Y_1 and Y_2 be the two orbit transversals used to define the annotation maps λ_1 and λ_2. For each $y \in Y_1$ there exists an automorphism $\phi_y \in \varGamma$ such that

$$\phi_y(y) \in Y_2.$$

We get for $a, b \in Y_1$ that

$$\begin{aligned}\lambda_1(a,b) &= \{\gamma \mid a \leq \gamma b\}\\ &= \{\gamma \mid \phi_a^{-1}\phi_a a \leq \gamma \phi_b^{-1}\phi_b b\}\\ &= \{\gamma \mid \phi_a a \leq \phi_a \gamma \phi_b^{-1}\phi_b b\}\\ &= \{\gamma \mid \phi_a \gamma \phi_b^{-1} \in \lambda_2(a,b)\}\\ &= \phi_a^{-1}\lambda_2(a,b)\phi_b.\end{aligned}$$

The mapping $y \mapsto \phi_y$ therefore has the properties required by Definition 5. □

5 An Example of a Concept Lattice Orbifold

The ordered set in Figures 1–4 is part of a lattice orbifold. The lattice to be folded is the boolean lattice of all graphs with vertex set $V := \{1, 2, 3, 4\}$. There are 64 such graphs, and 11 up to isomorphism. This lattice can naturally be written as the concept lattice of the ×6–formal context $(\binom{V}{2}, \binom{V}{2}, \neq)$, see Figure 5.

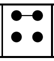
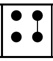
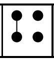
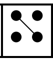
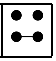
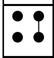
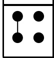
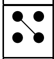
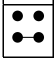
						
		×	×	×	×	×
	×		×	×	×	×
	×	×		×	×	×
	×	×	×		×	×
	×	×	×	×		×
	×	×	×	×	×	

Fig. 5. The standard context for the lattice of all graphs on four points

(A, B) is a formal concept of this context iff A is (the edge set of) some graph on V and B is (the edge set of) its complement.

When folding this lattice, we have several groups to choose between. The full automorphism group is, of course, isomorphic to the symmetric group S_6. The S_6-orbifold of this lattice is simply a chain of length four, with trivial annotation. Two formal concepts are in the same orbit iff their extents have equal cardinality.

More interesting in the sense of graph theory is the subgroup Γ_4 isomorphic to S_4, that is induced by the action of the vertex permutation on the edges. Two concepts are in the same orbit of this group iff their extents are isomorphic as graphs.

In Figures 1–4 the group Γ of our choice was the alternating group A_4, consisting of the 12 even permutations of V, in its induced action on the two-element

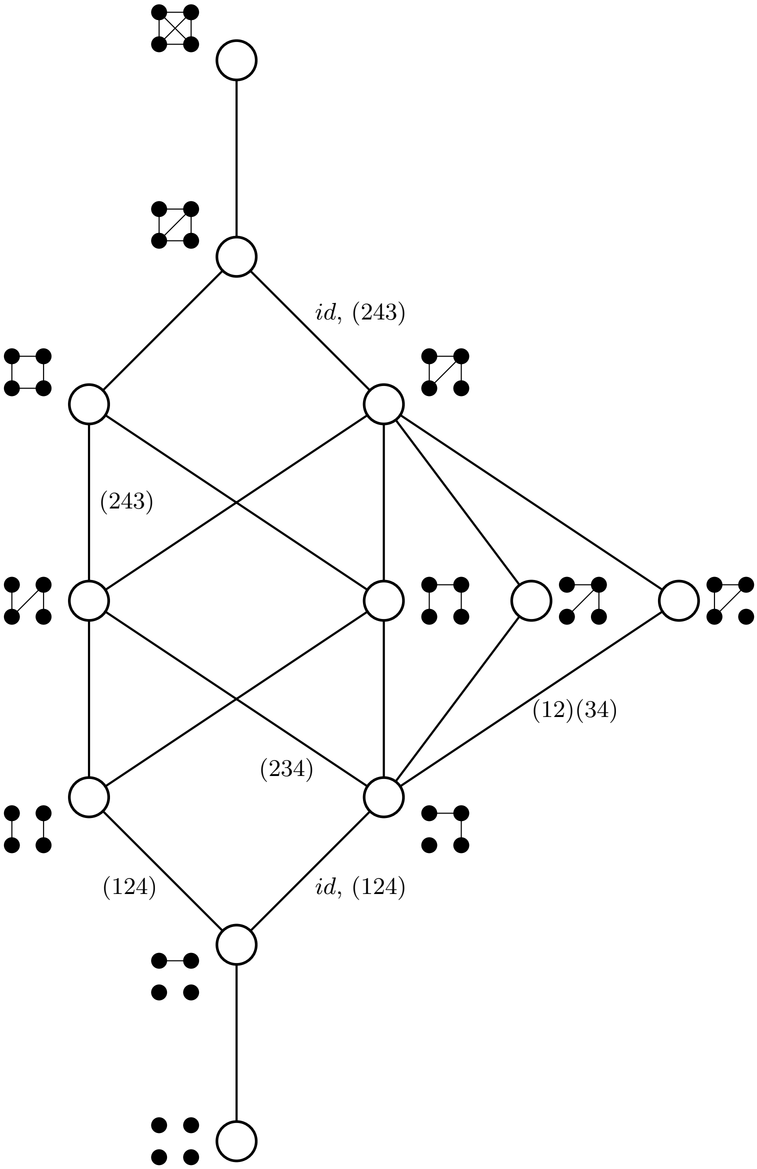


Fig. 6. An A_4– orbifold of the lattice of all graphs on four points

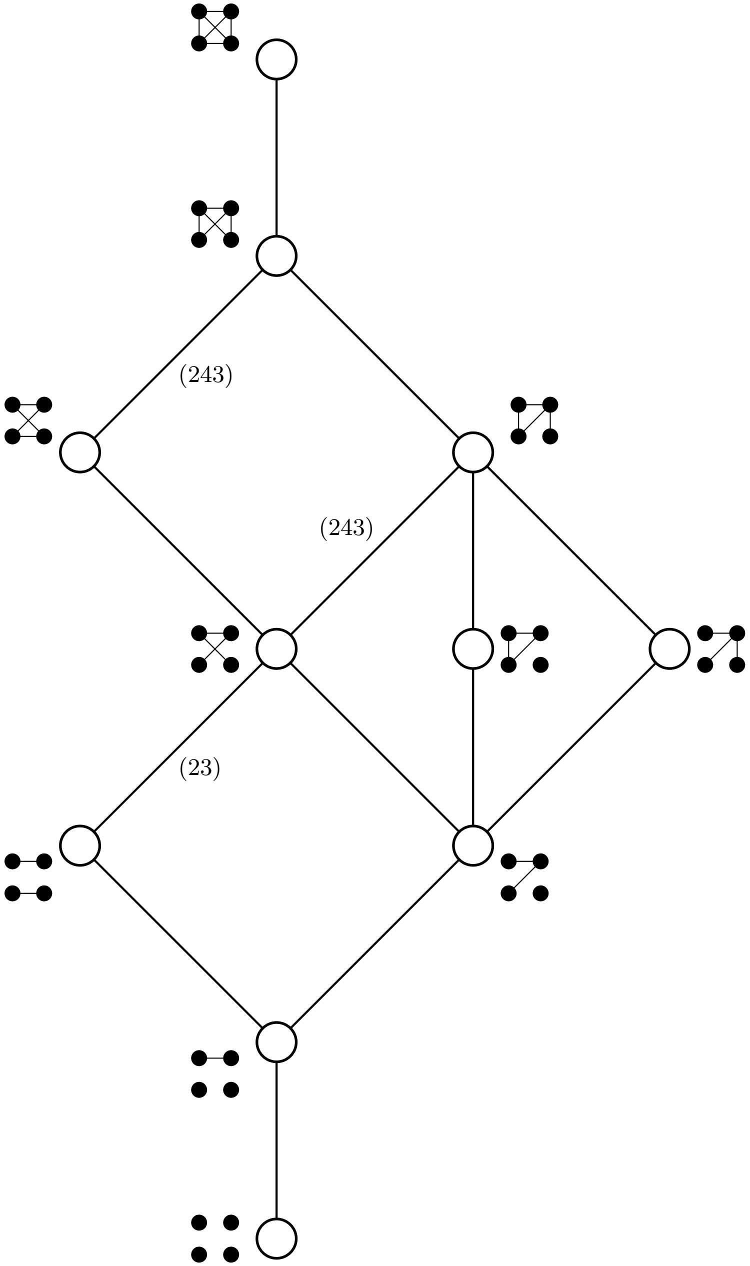


Fig. 7. An S_4– orbifold of the lattice of all graphs on four points

subsets. The Γ-orbifold of the lattice of all graphs on V is shown in Figure 6. Obviously, it is not a lattice. The orbifold diagram has a dual automorphism because the lattice it was generated from has one.

Since Γ is a subgroup of the group S_4, the Γ-orbifold in Figure 6 can itself be folded to obtain the diagram in Figure 7.

6 Context Orbifolds and a Lattice of Lattices

The symmetry group Γ may also be used to fold the formal context. The resulting **context orbifold** is

$$(G \setminus\!\!\setminus \Gamma, M \setminus\!\!\setminus \Gamma, \lambda),$$

where

$$\lambda : (G \setminus\!\!\setminus \Gamma) \times (M \setminus\!\!\setminus \Gamma) \to \mathcal{P}(\Gamma)$$

is the mapping defined by

$$\lambda(g^{\Gamma}, m^{\Gamma}) := \{\gamma \in \Gamma \mid g \; I \; \gamma m\}.$$

In practical computations we use the group structure for simplification. The orbits are replaced by orbit representatives, and since the values of the λ-mapping are unions of double cosets of the respective stabiliser groups (of g and m), they may be given by double coset representatives. However, a context orbifold may have many different such representations, and a theorem similar to Theorem 1 is required (and can be given) to guarantee representation invariance.

In the case of our example (in Figure 5) the context orbifold is a 1×1–table, since Γ is transitive both on objects and on attributes. The annotation gives the set $A_4 \setminus \{id, (12)(34)\}$.

As a more instructive example we give an orbifold representation of the "lattice of all concept lattices" with attribute set $\{a, b, c\}$. Recall that a **closure system** on a set M is a set $\mathcal{C} \subseteq \mathfrak{P}(M)$ of subsets of M which contains M and is closed under arbitrary intersections. The family of concept intents of any formal context is a closure system (as well as the family of concept extents). Any closure system is the system of intents of some formal context, and this context is determined by its intents up to clarifying, reducing and renaming objects.

The intersection of closure systems on M yields a closure system. The family of all closure systems on M therefore is itself a closure system, on the power set $\mathfrak{P}(M)$ of M, and therefore forms a complete lattice. The mathematical properties of these lattices have been studied by Caspard and Monjardet [2]. It is well known that this is encoded by the formal context

$$(\mathfrak{P}(M), \mathrm{Imp}(M), \models),$$

where $\mathfrak{P}(M)$ is the set of all subsets of M, $\mathrm{Imp}(M)$ is the set of all implications on M, and the relation $\models$ is defined as

$$S \models A \to B \quad :\Longleftrightarrow \quad A \not\subseteq S \text{ or } B \subseteq S.$$

	$\emptyset \to a$	$\emptyset \to b$	$\emptyset \to c$	$a \to b$	$a \to c$	$b \to a$	$b \to c$	$c \to a$	$c \to b$	$a, b \to c$	$a, c \to b$	$b, c \to a$
$\emptyset$				×	×	×	×	×	×	×	×	×
$\{a\}$	×					×	×	×	×	×	×	×
$\{b\}$		×		×	×			×	×	×	×	×
$\{a, b\}$	×	×		×		×		×	×		×	×
$\{c\}$			×	×	×	×	×			×	×	×
$\{a, c\}$	×		×		×	×	×	×		×		×
$\{b, c\}$		×	×	×	×		×		×	×	×	

Fig. 8. The reduced formal context for the lattice of closure systems on $\{a, b, c\}$. Each permutation of $\{a, b, c, \}$ induces an automorphism.

	$\emptyset \to c$	$a \to c$	$a, b \to c$
$\emptyset$	$\emptyset$	id	id
$\{a\}$	id	$(ab), (abc)$	$id, (abc)$
$\{a, b\}$	id	$id, (acb)$	(bc)

Fig. 9. An orbifold of the formal context in Figure 8. Objects and attributes are given by coset representatives. The cells of the table contain sets of double coset representatives, in analogy to Definition 2.

The extents of this formal context are precisely the closure systems on M, and the intents are the corresponding implicational theories. The extent lattice therefore is indeed the lattice of all closure systems on M.

The formal context given above is not reduced, and for computations it is convenient to use the standard context

$$(\mathfrak{P}(M) \setminus \{M\}, \mathrm{Imp}_r(M), \models),$$

where

$$\mathrm{Imp}_r(M) := \{A \to \{b\} \mid A \subseteq M, b \notin A\}.$$

For $M := \{a, b, c\}$ this yields the formal context in Figure 8. This formal context has 61 concepts, corresponding to the 61 closure systems on $\{a, b, c\}$. The cardinalities of these lattices are known up to $|M| = 6$ (see Habib and Nourine[3]). The values can be verified using the standard algorithm for generating concept lattices. Note, however, that the numbers grow rapidly. For $n = 1, \ldots, 6$ the numbers of closure systems on an n-element set are 2, 7, 61, 2480, 1385552, 75973751474 [7]. Up to isomorphism, there are 1, 2, 5, 19, 184, 14664, 108295846 closure systems. Note that there is a misprint in the sixth term of Sloane's sequence A108799 [7], as was noticed by Mike Behrisch [1].

The formal context in Figure 8 obviously has six automorphisms induced by the permutations of $\{a, b, c\}$. Folding the context by the induced action Γ of this symmetric group yield the context orbifold displayed in Figure 9. The concept lattice of the formal context in Figure 8 has 61 elements. Its lattice

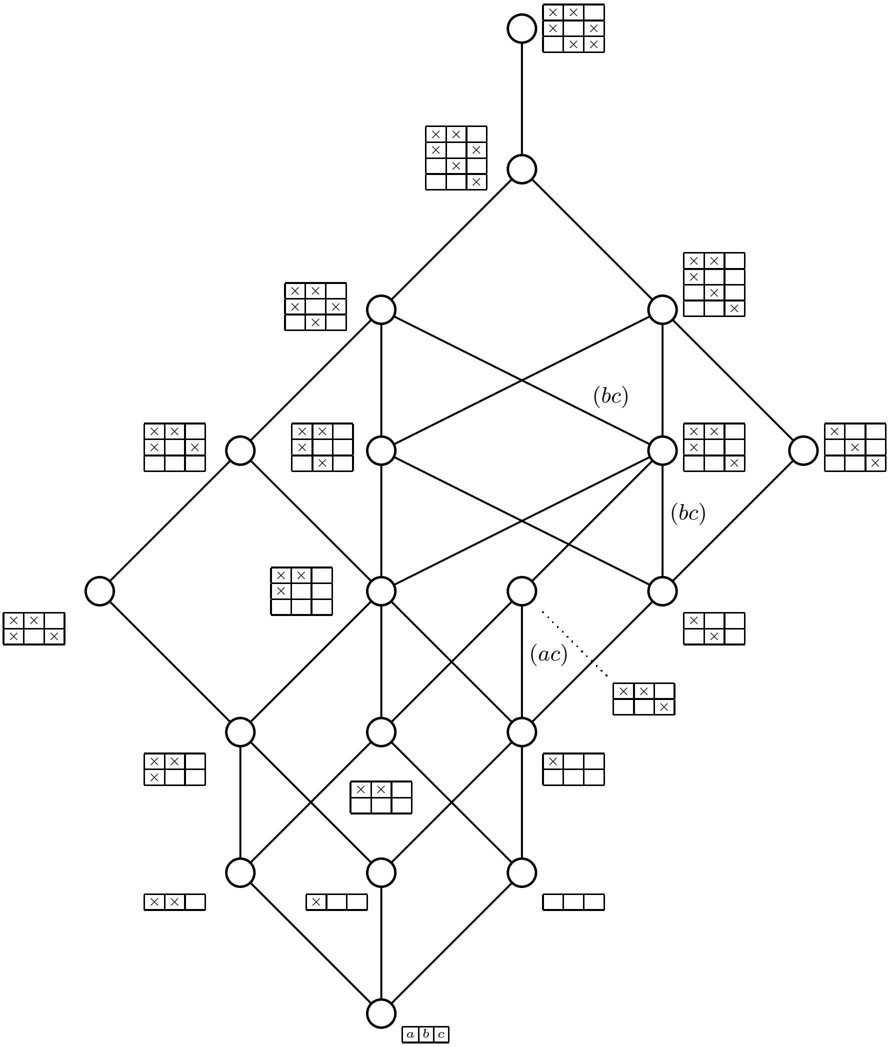


Fig. 10. The lattice orbifold for the lattice of the 61 closure systems on the set $\{a, b, c\}$. Each closure system is the system of intents of a unique row-reduced formal context with attribute set $\{a, b, c\}$. These contexts are given up to permutations of $\{a, b, c\}$. The context at the least element has empty object set.

orbifold has 19 elements. It is displayed in Figure 10. Note that the diagram in Figure 10 is very intuitive, because it represents the closure systems "up to isomorphism". However, the containment order "up to isomorphism" does not give a lattice, it is actually not a mathematically precise notion right away. The annotated diagram, together with the definitions on which the annotation is built, make the idea of a hierarchy of structures "up to isomorphism" precise and mathematically accessible.

7 Outlook

A detailed theoretical framework and a good algorithmic basis are needed to make context and lattice orbifolds applicable. Algorithms must be given to compute the lattice orbifold directly from the context orbifold, and, even more interestingly, to compute the folded stem base. A package based on the GAP system [4] has been implemented and is available upon request. Some of these questions will be treated in a subsequent paper, but many are still open.

8 Conclusion

The interplay between concept lattice orbifolds and context orbifolds offers a powerful technique for the investigation of lattices with symmetries. Although the necessary foundations were provided by Zickwolff [8] in a very general setting, it requires some effort to adapt them to the case of contexts and lattices. We have shown here how this can be done and that interesting results can be obtained.

References

1. Behrisch, M.: Personal communication (2006)
2. Caspard, N., Monjardet, B.: The lattices of closure systems, closure operators, and implicational systems on a finite set: a survey. Discrete Applied Mathematics 127(2), 241–269 (2003)
3. Habib, M., Nourine, L.: The number of Moore families on $n = 6$. Discrete Mathematics 294, 291–296 (2005)
4. The GAP Group, GAP – Groups, Algorithms, and Programming, Version 4.4.12 (2008), http://www.gap-system.org
5. Ganter, B., Reuter, K.: Finding closed sets: a general approach. Order 8, 283–290 (1991)
6. Ganter, B., Wille, R.: Formal Concept Analysis – Mathematical Foundations. Springer, Heidelberg (1999)
7. Sloane, N.J.A.: The On-Line Encyclopedia of Integer Sequences. Sequences A102896 and A108799, http://www.research.att.com/~njas/sequences/A108799
8. Zickwolff, M.: Rule Exploration: First Order Logic in Formal Concept Analysis. Dissertation, Darmstadt (1991)

The Advent of Formal Diagrammatic Reasoning Systems

Frithjof Dau

SAP Research CEC Dresden

Abstract. In knowledge representation and reasoning systems, diagrams have many practical applications and are used in numerous settings. Indeed, it is widely accepted that diagrams are a valuable aid to intuition and help to convey ideas and information in a clear way. On the other side, logicians have viewed diagrams as informal tools, but which cannot be used in the manner of formal argumentation. Instead, logicians focused on symbolic representations of logics. Recently, this perception was overturned in the mid 1990s, first with seminal work by Shin on an extended version of Venn diagrams. Since then, certainly a growth in the research field of formal reasoning with diagrams can be witnessed. This paper discusses the evolution of formal diagrammatic logics, focusing on those systems which are based on Euler and Venn-Peirce diagrams, and Peirces existential graphs. Also discussed are some challenges faced in the area, some of which are specifically related to diagrams.

1 Introduction

Formal Concept Analysis (FCA) is a mathematical theory applied successfully in a wide range. The impact and success of FCA and the large number of applications in the real world cannot be explained solely with the mathematical results and the mathematical power of FCA. The driving force behind FCA lies in the understanding of mathematics as a science which encompasses the philosophical basis and the social consequences of this discipline as well, and a main goal of FCA from its very beginning has been the support of rational communication and the representation and processing of knowledge. Lattice theory is reworked in order to integrate and to rationalize origins, connections to and interpretations in the real world. As Wille says in [78]:

> The aim is to reach a structured theory which unfolds the formal thoughts according to meaningful interpretations allowing a broad communication and critical discussion of the content.
>
> Wille, 1996

Thus the results of lattice theory had in FCA to be presented in a way which makes them understandable, learnable, available and criticizable, particularly for non-mathematicians. One means to achieve this goal is the diagrammatic representations in form of their Hasse diagrams. FCA, being is a mathematical theory

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 38–56, 2009.

formalizing the philosophical notion of concepts, has been later extended to *contextual logic* in order to revitalize the traditional philosophical understanding of logic which is based the the doctrines of concepts, judgments and conclusions. Again, an important aspect of contextual logic is that its core notions, i.e. both judgments and conclusions, can be diagrammatically represented (for this purpose, Sowa's conceptual graphs [70] are utilized).

It is widely accepted that diagrams play an important role for representing information in accessible and intuitive ways. In mathematics, however, there does still exist a long-standing prejudice against non-symbolic representation, particularly in mathematical logic. Without doubt diagrams are often used in mathematical reasoning, but usually only as illustrations or thought aids. Diagrams, many mathematicians say, are not rigorous enough to be used in a proof, or may even mislead us in a proof. Thus diagrams have been excluded from formal proof techniques and were considered only as a heuristic aid.

Interestingly, most of the ancient systems which can be considered as predecessors of formal logic are diagrammatic systems. We name Euler circles, Venn diagrams and Venn-Peirce diagrams, Frege's Begriffsschrift and Peirce's existential graphs. It was not until the end of the 19th century that symbolic notations took over in mathematical logic.

After more than a century of an absolute dominance of symbolic notations for logic, the last two decades show an increasing interest in formal elaborations of diagrammatic reasoning systems (DRSs), that is, formal logic systems with a precise syntax, semantics, and (diagrammatic) reasoning facilities. In this paper, we investigate this advent of diagrammatic reasoning systems from various perspectives. In order to do so and to motivate the use of DRSs, we first discuss in Sec. 2 possible applications of DRSs in software engineering and knowledge representation. An introduction into the Euler-Venn-Peirce family of diagrams as well as into existential graphs is provided in Sec. 3. Seminal work in the field of DRSs which present and elaborate these historical system is presented in section 4. Although that this seminal work laid the path towards a mathematically precise development of DRSs, it can be argued that they still do not fulfill the requirements of a rigorous mathematical system. Sec. 5 presents methodologies for developing DRSs in a precise manner. Contemporary systems which follow these methodologies are then presented in Sec. 6. Finally, we conclude with a discussion of the area.

2 Examples of Application Areas

In order to motivate the development of DRSs, we present in this section two different application areas of DRSs in the fields of software engineering and knowledge representation in the Semantic Web.

2.1 Software Engineering

In Software Engineering, we observe an increasing demand and development of diagrammatic languages which are used for describing, specifying and

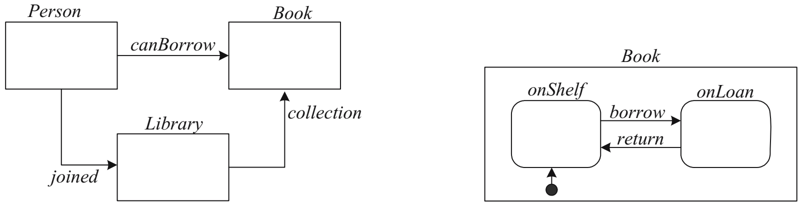


Fig. 1. A UML class diagram and a UML state chart

communicating a wide range of modeling aspects. In software engineering, different kinds of stakeholders are involved in the modeling process. Besides IT professionals like programmers and developers, this includes people like managers and customers, which often do not have a dedicated technical expertise. As there is a need of software specifications being accessible to all stakeholders, symbolic languages and formalizations are not suited for modeling purposes, and various diagrammatic languages like UML (unified modeling language)[1] or BPMN (business process modeling notation)[2] have been developed.

UML is in fact a whole suite of (mostly) diagrammatic notations, including *class diagrams*, *state charts* and various others. In Fig. 1, a UML class diagram and a UML state chart are depicted. Both of them refer to a scenario for describing a library lending system. The class diagram expresses relationship between the classes *Person*, *Library* and *Book*: persons can borrow books, persons can join libraries, and libraries have collection of books. Books can be in two different states, namely *onShelf* or *onLoan*, as it is expressed by the state chart. Initially (expressed by the bold dot and arrow), books are on the shelf. If a book is borrowed, its state changes to being on loan, and vice versa, a book on loan becomes being on shelf if it is returned.

There does exist a part of the UML which is non-diagrammatic: This is the Object Constraint Language (OCL) which has been developed to describe formal constraints on software models. As a very simple example of a constraint we might wish to enforce on a library lending system is that people can only borrow books that are in the collections of libraries they have joined. Using the OCL, this is achieved as follows:

```
Context Person inv:((self.joined.collection–>asSet)–>includesAll(self.canBorrow))
```

Being a symbolic notation, OCL does not follow the general diagrammatic approach of UML and is for this reason probably not as easily understandable diagrammatic parts of UML. Thus it is reasonable to develop a diagrammatic variant of OCL. A groundbreaking approach towards this aim has been undertaken by Kent, who introduced in [41] the class of *constraint diagrams*. An example for such a diagram is provided in Fig. 2. This diagrams expresses the same constraint as its symbolic counterpart we have just given, but compared to the symbolic notation, the diagram fits better in the general visual theme of OCL.

[1] `http://www.uml.org`

[2] `http://www.bpmn.org`

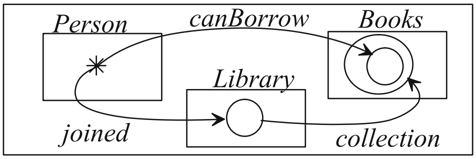


Fig. 2. A constraint diagram

Mainly driven by the Visual Modeling Group (VMG) in Brighton, UK[3], constraint diagrams are developed as a formal DRS, including a formal syntax, FOL-based, semantics, and diagrammatic reasoning facilities.

2.2 Knowledge Representation and the Semantic Web

Is has long been argued that diagrams are particularly useful for knowledge representation systems [56,28,48]. In this section, we focus on knowledge representation within the Semantic Web, particularly on RDF(S) and OWL.

The underlying layer for knowledge representation within the Semantic Web is the Resource Description Framework (RDF) and its extension RDF Schema (RDFS). The essential idea of RDF is that each atomic piece of information can be represented as a triple `(subject predicate object)`, and an RDF Knowledge Base is a set of RDF triples. There are different representations of RDF KBs. First of all, there exists a machine processable XML serialization of RDF. Secondly, we have the notion of an RDF KB as a set of RDF triples. Thirdly, we can represent an RDF KB by means of an labeled graph, where a triple `(s p o)` is modeled by two vertices labeled with `s` and `o`, respectively, and which are connected by an edge labeled with `o`.

Tim-Berners Lee, inventor of the WWW and the driving force behind the semantic web, recently reformulated his vision of the Semantic Web as a "Giant Global Graph".[4] This understanding lead to a significant change in the architecture of the WWW: Now the W3C considers RDF *graphs* as basic data structures for representing information.[5] This paradigmatic shift clearly hints to the importance of graph-based logics within the Semantic Web framework.

A more sophisticated level in the W3C stack (the hierarchy of W3C description languages) is the level of ontologies. The W3C recommendation is OWL (Web Ontology Language), which has recently been extended to OWL 2.0. The formal background of OWL and OWL 2.0 is the family of Description Logics (DLs, see [1]). DLs are a common family of knowledge representation formalisms tailored to express knowledge about concepts and concept hierarchies. They include sound and complete decision procedures for reasoning about such knowledge.

The formal notation of DLs has the flavor of a variable-free first order logic. In fact, DLs correspond to (decidable) fragments of first order logic, and they have

[3] `http://www.cmis.brighton.ac.uk/research/vmg`

[4] `http://dig.csail.mit.edu/breadcrumbs/node/215` created 2007/11

[5] `http://www.w3.org/Consortium/technology`, created 2008/01

a well-defined, formal syntax, a semantics in the form of Tarski-style models, and sound and complete calculi (e.g. based on Tableaux-algorithms). It is often emphasized that DLs offer, in contrast to other knowledge representation languages, sound, complete and (empirically) practically useful reasoning services.

The fact that the notation of DLs is variable-free makes them easier to comprehend than the common first order logic formulas which include variables. Nonetheless, for untrained users, the symbolic notation of DLs can be hard to learn and comprehend. A main alternative to the symbolic notation is the development of a diagrammatic representation of DLs. In [55], the introduction to the *Description Logic Handbook*, Nardi and Brachman write a "major alternative for increasing the usability of Description Logics as a modeling language" is to "implement interfaces where the user can specify the representation structures through graphical operations."

For RDF, mathematical elaborations based on graph theory have been developed. They include Tarski-style semantics as well as sound and complete calculi, latter either based on "projections" (see [4,5]) or on diagrammatic rules (see [19]). That is, RDF is developed as a fully-fledged diagrammatic logic. A first attempt at a diagrammatic representation for DL is can be found in [28], where Gaines elaborates a graph-based representation for the textual DL CLASSIC. More recently, the focus has shifted from the development of proprietary diagrammatic representations to representations within the framework of UML (Unified Modeling Language). In 2003, the *Object Management Group* requested a metamodel for the purpose of defining ontologies. Following this proposal, [7] provides a UML-based, diagrammatic representation for OWL DL. In these approaches, the focus is on a graphical *representation* of DL, however, as often emphasized, *reasoning* is seen as a distinguishing feature of DL and such reasoning is not supported diagrammatically by that treatment. A first step towards developing DLs as fully fledged diagrammatic logics, based on Peirce's Existential Graphs, has been carried out for the DL $\mathcal{ALC}$ (the smallest propositionally closed DL) in [27]. Research is in progress to extend this approach to more expressive DLs.

3 Historical Systems

In this section, we shortly discuss the historical background of DRSs, namely Euler circles (a.k.a. Euler diagrams) [24], Venn diagrams [77], Venn-Peirce diagrams [59] and Peirce's existential graphs [31,33,60].

An Euler diagram is a finite collection of closed curves drawn in the plane. For example, in Fig 3, d_1 is an Euler diagram which expresses that nothing is both a car and a van. Venn diagrams differ from Euler circles in the respect that the curves are drawn in a way that all possible set combinations are shown in the diagram, and shadings are used to show that certain set combinations must be empty. So the Venn diagram d_2 expresses the same information as d_1. Finally, in Venn-Peirce diagrams, o-signs are utilized to assert the emptiness of a set. More importantly, Peirce also introduced $\otimes$-signs which denote elements so that –in contrast to Euler Circles or Venn-diagrams– now the non-emptiness of sets can

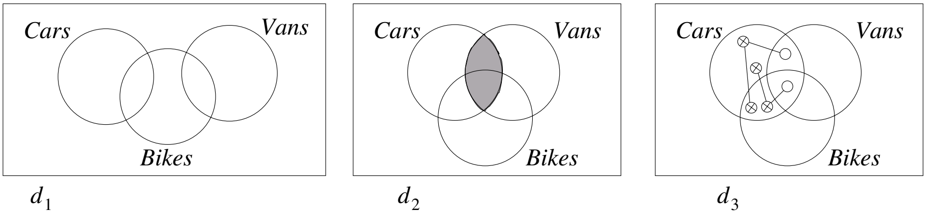


Fig. 3. An Euler diagram, a Venn diagram, and a Venn-Peirce diagram

be explicitely expressed. These signs can be assembled with lines to sequences, and the lines are read in a disjunctive manner. If we consider the diagram d_3, the outer spider asserts that the set $cars \cap \overline{Vans}$ is non-empty (denoted by the two $\otimes$-signs) or that the set $Cars \cap Vans \cap \overline{Bikes}$ is empty (denoted by the o-sign). So the d_3 is a diagrammatic representation of the formula

$$\begin{aligned}&(Cars \cap \overline{Vans} \neq \emptyset \vee Cars \cap Vans \cap \overline{Bikes} = \emptyset) \wedge \\ &(Cars \cap \overline{Vans} \neq \emptyset \vee Cars \cap Vans \cap Bikes = \emptyset).\end{aligned}$$

which expresses that either $Cars \cap Vans = \emptyset$ or $Cars \cap \overline{Vans} \neq \emptyset$.

Existential graphs (existential graphs) are a different diagrammatic system with a focus on representing and reasoning with relations. The system of existential graphs is divided into three parts: Alpha, Beta and Gamma, which presuppose and are built upon each other. Alpha corresponds to propositional logic, Beta corresponds to first order logic, and Gamma encompasses features of higher order logic, including modal logic, self-reference and more. In contrast to Alpha and Beta, Gamma was never finished by Peirce, and even now, only fragments of Gamma (mainly the modal logic part) are elaborated to contemporary mathematical standards. In this paper, only Alpha and Beta are introduced.

The existential graphs of Alpha consist only of two different syntactical entities: (atomic) propositions, and so-called *cuts* which are represented by fine-drawn, closed, doublepoint-free curves. Essentially, writing different graphs on the plane expresses their conjunction, and enclosing a graph by a cut denotes its negation. Below, two examples of Alpha graphs are given. In the left graph, the propositions 'it rains', 'it is stormy' and 'it is cold' are written side by side, thus the graph means 'it rains and it is stormy and it is cold'. The right graph has the meaning 'it is not true that it rains and it is stormy and that it is not cold', i.e. 'if it rains and if its stormy, then it is cold'.

If we go from the Alpha part of existential graphs to the Beta part, predicate names of arbitrary arity may be used, and a new sign, the LINE OF IDENTITY, is

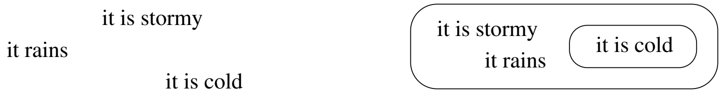


Fig. 4. Two Alpha graphs

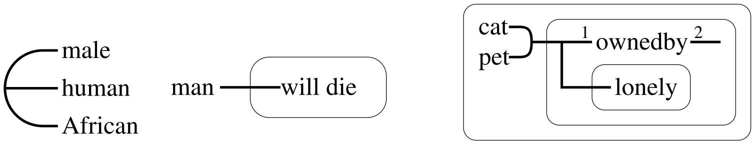


Fig. 5. Three Beta graphs

introduced. Lines of identity are used to denote both the existence of objects and the identity between objects. They can be connected to networks. The meaning of the Beta graphs in Fig. 5 are ‘there exists a male, human african’, ‘there exists a man who will not die’, and ‘it is not true that there is a pet cat such that it is not true that it is not lonely and owned by somebody’, i.e., ‘every pet cat is owned by someone and is not lonely’.

4 Seminal Work on Historical Systems

In this section, we provide an overview on important seminal work which aims at elaborating the historical systems of the last section.

4.1 Shin’s Extended Venn Diagrams

Certainly a landmark work on diagrammatic reasoning is Shin’s elaboration of Venn-Peirce-diagrams in [67]. In fact, the main point of [67] is to argue that developing a diagrammatic reasoning system which possesses the rigor of formal mathematical logic is possible; the elaboration of Venn-Peirce-diagrams can be seen as a proof of concept for her approach.

We shorty discuss Shin’s so called *Venn-II system*. Essentially, Venn-II is based on Venn diagrams where Peirce’s ⊗-sequences are added and where diagrams can be taken in disjunction. The Venn-II diagram in Fig. 6 expresses the same as the Venn-Peirce diagram d_3 in Fig. 3; the line connecting the two boxes represents disjunction (similar to the lines in ⊗-sequences).

The semantics are formalized in much the same way as traditional approaches. Shin defines *set assignments* which are analogous to structures and then specifies conditions under which a set assignment satisfies a diagram; see [67] for more details. In [67], ten reasoning rules are defined for Venn-II and shown to form a sound and complete set.

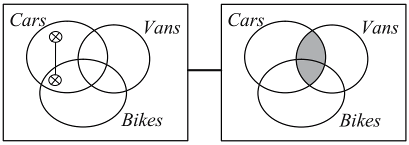


Fig. 6. A Venn-II diagram

Shin shows that Venn-II is equivalent in expressive power to Monadic First Order Logic (MFOL) without equality. This is a noteworthy result: Though it is straightforward to convert Venn-II diagrams into MFOL sentences; the converse requires some effort as there is not a natural mapping from arbitrary MFOL sentences to Venn-II diagrams.

Shin claims that her system is formally rigorous, but a closer observation reveals that this claim cannot be sustained. The main reason is that Shin defined the syntax of Venn-II at the concrete (drawn) diagram level. This resulted in a lack of mathematical preciseness in her work, leading to unclear definitions and errors in proofs (see [23]). Of course, this lack of rigor should not detract from the importance of Shin's work because she laid the essential foundations for the acceptance of diagrams as formal tools.

4.2 Seminal Work on Existential Graphs

The main treatises on existential graphs are probably the books of Zeman [80], Roberts [65] and Shin [68]). Each of this books focuses on different aspects of existential graphs.

Probably the most prominent book on existential graphs is D. Robert's 'The Existential Graphs of Charles S. Peirce'. This book offers the most comprehensive description of the whole system of existential graphs –Alpha, Beta and Gamma– and its genesis. Particularly, Gamma is described to a large degree. Robert's treatise is definitely an outstanding work. However, from a mathematical point of view, this book is clearly insufficient. Roberts does not provide any (technical or mathematical) definitions for existential graphs, neither their syntax, semantic, nor inference rules, and he relies solely on the graphical representations of graphs.

In contrast to Roberts, J. J. Zeman's book 'The Graphical Logic of C. S. Peirce' is, from a mathematical point of view, the best of the books which are here discussed. Zeman provides a mathematical elaboration of Alpha, Beta, and the part Gamma which extend Beta by adding the broken cut. Whereas Roberts solely relies on the graphical representations of graphs, Zeman defines existential graphs inductively as abstract structures, i.e. in fact as sequences of signs. Like in the other treatises, Zeman does not provide a mathematical, extensional semantics for Peirce's graphs. Instead of that, he defines mappings between the systems Alpha, Beta, Gamma and appropriate systems of propositional, first order, and modal logic. These translations from graphs to symbolic are correct, but they are arguably very technical and clumsy. Even quite simple existential graphs are translated to rather complex first order logic-formulas.

Sun Joo Shin's book 'The Iconic Logic of Peirce's Graphs' discusses only Alpha and Beta. Her interest in existential graphs is philosophically driven, and she uses existential graphs as a case study for her goal to provide a formal approach to diagrammatic reasoning. As the title of her book suggests, she focuses on the diagrammatic aspects, particularly the iconicity, of existential graphs. She compares symbolic and diagrammatic approaches to mathematical logic and works out that the *long-standing prejudice against non-symbolic representation in logic, mathematics, and computer science* is due to the fact that diagrammatic systems are evaluated in terms of symbolic systems. Then, based on her arguments, she reconsiders Alpha and Beta from iconic aspects and rewrites the reading algorithms, that is, the translations from graphs to symbolic logic, and the transformation rules in order to improve their iconicity and naturalness. mathematical preciseness. Similar to her approach for Venn diagrams, Shin only uses the graphical representation of existential graphs (which is quite surprising, as Shin elaborates carefully the semiotic aspects of existential graphs), which again results in a lack of mathematical preciseness. Particularly, Shin's definitions (and later on, theorems and proofs) cannot be considered to be mathematical. This leads to a mistake in her reading algorithm, and –even worse– some of her newly implemented transformation rules are not sound [18]. Finally, Shin does not provide an extensional semantics for Peirce's graphs: her reading algorithms are translations to symbolic logic, thus translations from one formal system to another.

5 Methodologies for Formalizing Diagrams

In the previous sections on seminal work on the historical systems, we already addressed that from a mathematical point of view, the elaborations of the historical systems are not sufficient due to a lack of formal preciseness. The formal shortcomings of the seminal works are mainly due to the fact that a general methodology for a formal elaboration of diagrammatic logics was missing. To put it more precisely: A thorough scrutiny on how to deal with diagrams (instead of symbolic notations like formulas) was not carried out.

In order to elaborate diagrammatic logics in a solid formal manner, it is crucial to note that we deal with two different entities: A mathematical structure and its diagrammatic representation. In any diagrammatic representation of a mathematical structure, we have to disregard certain graphical properties of the diagrams, while other properties are important. This shall be exemplified with the following diagrams of Alpha existential graphs in Fig. 7.

The shape of the cuts (negation ovals) or the place of propositional variables and other cuts in the area of a given cut has no significance, thus all diagrams convey the same meaning. They can be read as 'it is not true that A and B, but not C hold', i.e., 'A and B imply C'. Peirce did not understand EGs as graphical entities at all. For him, the three diagrams are not *different graphs* with the same meaning, but *different representations*, i.e., diagrams, of the same graph. This is a crucial distinction, which obviously corresponds to the distinction between *types* (graphs) and *tokens* (graph replicas), as it is known from philosophy. The type-token issue if far from being settled; nonetheless, this important distinction helps us to draw the following conclusion: In any elaboration of a DRS, the

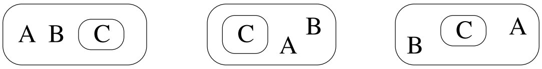


Fig. 7. Three diagrams of one Alpha graph

diagrams should not be defined as graphical entities. Instead, we need a definition of mathematical structures which encompass exactly the facts which are represented in the diagrams, and the diagrams should be understood as (mere) representations of these structures.

Two thorough discussions on the type-token-issue for DRSs can be found in [34,17]. Both papers argue that it is essential to provide mathematical definitions for the type-level (called *abstract syntax* in [34]). For the token-level (called *concrete syntax* in [34]), the papers come to different conclusions: Whereas [34] argues that the token-level as well as the relationship between these two levels has to be formally captured as well, [17] argues that this is in fact not needed. For the purpose of this paper, this difference has no significance. The crucial point here is the following: In symbolic approaches to formal logic, once we have chosen a formula –i.e. a sequence of signs– is chosen, the representation of the formula is uniquely given. So in symbolic logic, we have only one level of representation to deal with. In graph-based DRSs on the other hand, a given graph can have very different diagrammatic representations. For example, in Fig. 7 we have different representations of the same Alpha graph. So, for representing information, symbolic and linear notions of logic have a **one layer architecture**, and diagrammatic logic systems have a **two layer architecture**.

The general advantages of this two layer structure for the pragmatics of diagrams is already discussed to a large extent [49,66,6,68,57]. It is usually argued that the additional diagrammatic layer, often referred to as ‘secondary notation’, provides the essential means to improve the pragmatics of a representation system. As Oberlander writes in [57]: “secondary notation is the very stuff of graphical pragmatics–meaningful structures which go beyond the plain semantics of the system.”

6 Contemporary Systems

The last two decades witness the rise of formal diagrammatic reasoning systems in two respects: First, more expressive diagrammatic reasoning systems have emerged and, more importantly, are formalized in a mathematical precise manner. Second, for some of these logics, computer programs have been developed and applied to various settings. This section summarizes some recent advances in the field.

6.1 Spider and Constraint Diagrams

Spider diagrams (SDs) and constraint diagrams (CDs) are a DRS based on Euler circles and Venn-Peirce diagrams. They are elaborated as abstract mathematical structures, including extensional semantics and inference rules. Their development is mainly driven by the Visual Modeling Group in Brighton.

Spider Diagrams. Spider diagrams combine features of Venn diagrams and the more user friendly Euler diagrams. They can be thought of as extending Venn-II

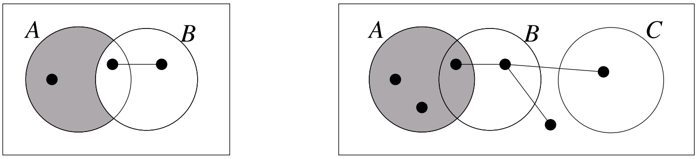


Fig. 8. Two spider diagrams

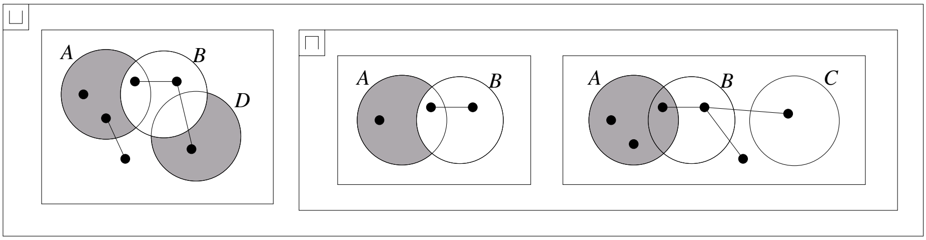


Fig. 9. A compound spider diagram

diagrams. Various different systems exist today, for example [35,36,38,40,74]. In Fig.8, two examples of so-called *unary* SDs are depicted.

The left diagram contains two *existential spiders*. Each spider denotes a uniquely given object (i.e., different spiders necessarily denote different objects). In contrast to Venn-Peirce diagrams, shading a region does not necessarily mean the corresponding set is empty. Instead, a region does not contain *more* elements than the elements represented by some spiders. So the SDs reads as follows: there are two sets A and B, the set $A - B$ contains exactly one element, and the set B contains at least one element. In the right diagram, a third contour representing a set C is involved. This diagram uses the notions of Euler circles: as the contour labeled C does not overlap with the A and B-contours, C and $A \cup B$ are disjoint[6]. Moreover, there are three elements u, v and w (represented by the three spiders) such that $u, v \in A$ and $w \notin A - B$. Further, due to its shading, the set $A - B$ must not contain any other elements than u and v, i.e. it contains exactly two elements, and $A \cap B$ must not contain any other elements than w, i.e., it contains no element or at most one element.

Unary SDs can be propositionally combined with the logical operators $\sqcap$ ('and') and $\sqcup$ ('or'). In Fig. 9 an SD which uses these conjunctors is shown. It has been shown that SDs are equivalent in expressive power to MFOL with equality [72]; thus they have a higher expressiveness than Venn-II.

Constraint Diagrams. SDs only allow reasoning about sets, namely unary predicates, and provide no possibility to represent or reason about any sort of relations. Moreover, as already mentioned, the spiders in SDs are 'existential

[6] The usage of disjoint features in Euler circles has a drawback: not all abstract SDs are drawable (see [75]).

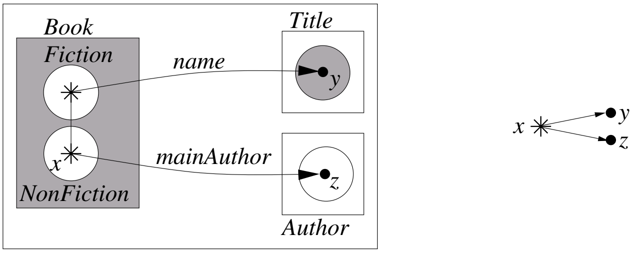


Fig. 10. A constraint diagram with a reading tree

spiders' as they can be read as existentially quantified objects. Constraint diagrams are essentially an extension of SDs with universal spiders (quantifiers) and arrows which represent binary relations. A full constraint notation was introduced by Kent [42] in an informal manner. Since then, several papers attempt to elaborate a full mathematical treatment of Kent's vision, including syntax, semantics, and a sound and complete calculus for constraint diagrams. For example, in [26], the syntax and semantics of full constraint diagrams is developed but a sound and complete calculus is elusive. The first ever constraint reasoning system (i.e., including a sound and complete calculus) was developed by Stapleton [73] but compared to Kent's approach it has several limitations.

An example for a constraint diagram was already given in Fig. 2. In the formal approach to constraint diagrams, we have both existential and universal spiders, which renders the formalization of the diagrams difficult, as there is no natural order placed on the quantifiers. This difficulty is overcome by augmenting each diagram with a *reading tree*; the formalization of constraint diagrams can be found in [25]. An example can be seen in Fig. 10. This diagram expresses that $Book$, $Title$ and $Author$ are pairwise disjoint, $Fiction$ and $NonFiction$ form a partition of $Book$, and finally every book x has a unique name which is its title and x has at least one main author.

Applications. For Euler diagrams, the Visual Modelling Group has developed a diagrammatic theorem prover called Edith.[7] Edith automates the search for Euler diagram proofs using a variety of reasoning rule sets and heuristics [71]. For a given rule set, Edith finds a shortest proof. In Edith, users can create diagrams, apply rules to write a proof and ask Edith to seek a proof from one diagram to another. With regard to applications, SDs have been used to detect component failures in safety critical hardware [12] and (slight variations of them) to represent non-hierarchical file systems [21] and for viewing clusters which contain concepts from multiple ontologies [32].

As mentioned in section 2, one area that could benefit from the development of diagrammatic logics is software engineering. Constraint diagrams were designed with this application area in mind and extend the spider diagram language. Constraint diagrams have been used in a variety of areas including formal

[7] `http://www.cmis.brighton.ac.uk/research/vmg/autoreas.htm`

object oriented specification [37,44] and a visual semantic web editing environment [51,81]. Prototype tools to support their use are available from [64].

6.2 Conceptual Graphs

John Sowa's conceptual graphs (CGs) are based on Peirce's existential graphs. Sowa writes that they are an *extension of existential graphs with features adopted from linguistics and AI. The purpose of the system is to express meaning in a form that is logically precise, humanly readable, and computationally tractable.* CGs are designed to be used in fields like software specification and modeling, knowledge representation, natural language generation and information extraction, and these fields have to cope with problems of implementational, mathematical, linguistic and even philosophical nature. In CGs, we can diagrammatically represent various entities and logical constructors, like concepts and relations, individuals, quantifiers, conjunction, different levels of negations, contexts, etc.

In Fig. 11, three well-known examples of CGs are provided. The graph d_1 has the meaning that Yoyo is a cat and Yoyo is on some (unnamed) mat. In d_2, so-called *contexts* are used. The meaning of this graph it that the person Tom believes the proposition that the person Mary wants the situation that Mary marries a sailor. In short: The person Tom believes that the person Mary wants to marry a sailor. Finally, in d_3, special contexts denoting negation are used. The device of two nested negation contexts can be understood as an implication. So the meaning of the graph is 'if a farmer owns a donkey, then he beats it'.

Due to the complexity of the system of CGs, it is nearly impossible and perhaps not even desirable to consider the overall system as an approach to formal logic. In fact, the whole system of CGs, as described by Sowa, goes beyond FOL. It can be argued that Sowa's original system does not fully meet the requirements of a formal logic system [20], but several fragments of CGs have been elaborated in a mathematically rigorous manner.

The most prominent fragment are *simple* CGs, where no sort of context or negation is used. For this fragment, there does exist a variety of different formalizations with different notations which only differ in details. A comprehensive comparison of the different approaches can be found in [39]. For all these

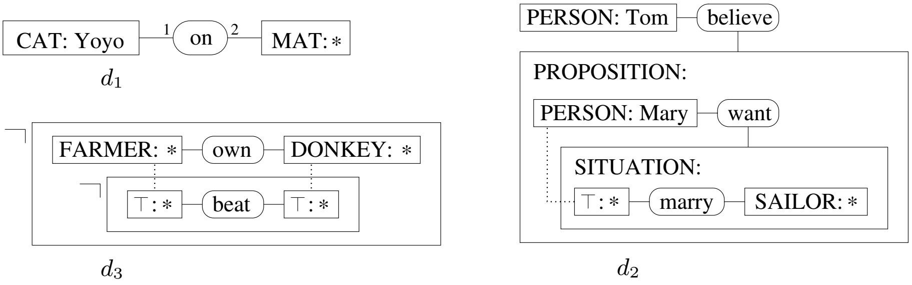


Fig. 11. Three conceptual graphs

approaches, a formal syntax, semantics (either via a translation of CGs to FOL or via direct model-theoretic approach) and reasoning facilities are provided. Examples are the works of Prediger and Dau, where the reasoning facilities come in form of graph transformation rules [9,15,52,61,62], works of Chein/Mugnier et al, where the entailment between CGs are described by meaning-preserving graph homomorphisms called *projections* [8,11,52], or works where so-called *standard-models* are considered [15,61,62]. One can even express "if-then"-statements with simple CGs, at is has been discussed in [2,3,52].

Simple CGs with contexts go beyond FOL. For these graphs, there exist different formalizations as well. We want to mention the work of Prediger [61,63], where the semantics for the graphs is based on triadic Formal Concept Analysis. Prediger provides a sound and complete set of rules for these graphs. A different approach has been taken by Simonet in [69,10] by translating the graphs to FOL-formulas, where the contexts are modeled by assigning to each concept box an additional argument which models the nesting of the box. In this approach, the notion of projections is accordingly extended.

For CG without contexts, different kinds of negation have been added. First we have to mention approaches where only relations, but not whole subgraphs, can be negated. A first step has been taken bay Kerdiles in [43]. Kerdiles extends the projections of simple CGs to simple CGs with atomic negation. This approach has been extended in [54,50], where for this framework, different logical approaches – classical logic, intuitionistic logic, logic with a closed-world semantics– are elaborated and compared. Finally, Klinger provides in [45,46,47] a different FCA-based approach which extends Predigers work to atomic negation.

Finally, we can consider CGs with full negation, where whole subgraphs can be negated. Kerdiles considers in [43] full negation as well. A comprehensive approach to add full negation to CGs is provided by Dau in [13,14,16]. In [13], Dau argues that to express full negation to CGs, a new syntactical entity to express negation has to be added, and he suggests to use the negation ovals of Peirce's existential graphs. These CGs with cuts are equivalent to full FOL with equality and it is equipped with a sound and complete set of graph transformation rules.

CGs have been implemented in a variety of computer programs. Two comprehensive frameworks are provided by Amine[8] and Cogitant[9]. Amine is a java open source platform for the development of intelligent systems and multi-agent systems covering several topics of the theory of CGs and AI. Cogitant is a set of C++-classes enabling to easily handle CGs as well as other objects of the CG-framework like the taxonomies of concepts and relations as well as rules.

7 Discussion

Steps like realizing the importance of the distinction between abstract and concrete syntax, the development of sound and complete diagrammatic reasoning

[8] `http://amine-platform.sourceforge.net`

[9] `http://cogitant.sourceforge.net`

systems or the development of DRS applications show that the research field of DRSs has made significant progress in the last decade, but there is still much more to be done.

Diagrammatic representations of information is investigated from various perspectives to a large extent, and their usability is sustained both by empirical and theoretical research. Anyhow, for the particular field of DRSs, this research has to be conducted to investigate the claim that the diagrams of DRSs are from a usability point of view superior to symbolic notations. This research must not be restricted to the diagrams: A distinguishing feature of DRSs is the provision of diagrammatic *reasoning* facilities, which thus have to be investigated as well.

It might be doubted that diagrams are *generally* easier to understand than symbolic notations. Instead, though diagrams certainly provide very effective means of displaying some information (like subset relationships), for other kinds of statements symbolic notations might turn out to be better suited. For this reason, developing heterogeneous or hybrid notations that incorporate both symbolic and diagrammatic parts is a promising approach. There has already some research been conducted in this direction, e.g. [30,76]. In fact, when a formal language is developed with the goal of a high usability, the design of the language depends on the requirements of intended application areas. This might render finding the right language difficult.

The syntax, semantics and reasoning facilities for DRSs take place on the abstract level. In symbolic notations, there does not exist a distinction between the abstract and concrete level, which leads to straightforward ways of representing statements in these notations. For DRSs, the situation is different. For a given statement on the abstract level, we have to find a convenient concrete representation. In other words: The automatic drawing of diagrams is a core research challenge for developing DRSs applications.

Finally, research on theorem provers for DRSs is in its infancy. There is certainly a need for sophisticated theorem provers which both work in an efficient way (though it is not a goal to outperform existing theorem provers for symbolic notations) and which produce proofs where the steps of the proofs are easily understood by users. Again, such theorem provers are needed to support the usability claim of diagrams.

The research questions can only be addressed by a joint effort of researchers from different fields like mathematicians, computer scientists, (cognitive) psychologists, or even designers. Thus it will be challenging to find appropriate answers. Anyhow, as the rise of diagrammatic representations in several computer science areas like software engineering, knowledge representation or semantic web shows, solving these questions is of highly practical interest.

Acknowledgment. The author wants to thank Gem Stapleton from the Visual Modelling Group for her valuable help and provision of diagrams.

References

1. Baader, F., Calvanese, D., McGuinness, D.L., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, Cambridge (2003)
2. Baget, J.-F.: A simulation of co-identity with rules in simple and nested graphs. In: Tepfenhart, W.M., Cyre, W. (eds.) ICCS 1999. LNCS, vol. 1640, pp. 442–455. Springer, Heidelberg (1999)
3. Baget, J.-F., Mugnier, M.-L.: Extensions of Simple Conceptual Graphs: The Complexity of Rules and Constraints. JAIR 16, 425–465 (2002)
4. Baget, J.-F.: Homomorphismes d'hypergraphes pour la subsumption en rdf/rdfs. In: Langages et modèles à objets 2004 (actes 10e conférence), RSTI - L'objet (numéro spécial), vol. 10(2-3), pp. 203–216 (2004)
5. Baget, J.-F.: Rdf entailment as a graph homomorphism. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 82–96. Springer, Heidelberg (2005)
6. Berger, S.: Studies on the uses and usefulness of diagrams. Master's thesis, ILLC, University of Amsterdam (2000)
7. Brockmans, S., Volz, R., Eberhart, A., Löffler, P.: Visual modeling of owl dl ontologies using uml. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 198–213. Springer, Heidelberg (2004)
8. Chein, M., Mugnier, M.-L.: Conceptual Graphs: Fundamental Notions. Revue d'Intelligence Artificielle 6(4), 365–406 (1992)
9. Chein, M., Mugnier, M.-L.: Conceptual graphs are also graphs. Technical report, LIRMM, Université Montpellier II, Rapport de Recherche 95003 (1995)
10. Chein, M., Mugnier, M.-L., Simonet, G.: Nested Graphs: A Graph-based Knowledge Representation Model with FOL Semantics. In: Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR 1998). Morgan Kaufmann, San Francisco (1998), `http://www.lirmm.fr/~mugnier/`
11. Chein, M., Mugnier, M.-L.: Concept types and coreference in simple conceptual graphs. In: Wolff, et al. [79], pp. 303–318
12. Clark, R.P.: Failure mode modular de-composition using spider diagrams. In: Proceedings of Euler Diagrams 2004. Electronic Notes in Theoretical Computer Science, vol. 134, pp. 19–31 (2005)
13. Dau, F.: Negations in simple concept graphs. In: Ganter and Mineau [29], pp. 263–276.
14. Dau, F.: Concept graphs and predicate logic. In: Delugach and Stumme [29], pp. 72–86.
15. Dau, F.: Concept graphs without negations: Standardmodels and standardgraphs. In: Ganter, B., de Moor, A., Lex, W. (eds.) ICCS 2003. LNCS (LNAI), vol. 2746, pp. 243–256. Springer, Heidelberg (2003); this paper is a part of [16] as well
16. Dau, F.: The Logic System of Concept Graphs with Negation. LNCS (LNAI), vol. 2892. Springer, Heidelberg (2003)
17. Dau, F.: Types and tokens for logic with diagrams: A mathematical approach. In: Wolff, et al. [79], pp. 62–93.
18. Dau, F.: Fixing shin's reading algorithm for peirce's existential graphs. In: Barker-Plummer, D., Cox, R., Swoboda, N. (eds.) Diagrams 2006. LNCS (LNAI), vol. 4045, pp. 88–92. Springer, Heidelberg (2006)

19. Dau, F.: Rdf as graph-based, diagrammatic logic: Syntax, semantics, calculus, normalforms. In: Esposito, F., Raś, Z.W., Malerba, D., Semeraro, G. (eds.) ISMIS 2006. LNCS, vol. 4203, pp. 332–337. Springer, Heidelberg (2006)
20. Dau, F.: Formal, diagrammatic logic with conceptual graphs. In: Hitzler, P., Scharfe, H. (eds.) Conceptual structures in Practice. CRC Press(Chapman and Hall/Taylor & Francis Group) (2009)
21. DeChiara, R., Erra, U., Scarano, V.: VennFS: A Venn diagram file manager. In: Proceedings of Information Visualisation, pp. 120–126. IEEE Computer Society, Los Alamitos (2003)
22. Delugach, H.S., Stumme, G. (eds.): ICCS 2001. LNCS (LNAI), vol. 2120. Springer, Heidelberg (2001)
23. Scotto di Luzio, P.: Patching up a logic of Venn diagrams. In: Proceedings 6th CSLI Workshop on Logic, Language and Computation. CSLI Publications (2000)
24. Euler, L.: Lettres a une princesse d'allemagne sur divers sujets de physique et de philosophie. Letters 2, 102–108 (1775); Berne, Socit Typographique
25. Fish, A., Flower, J., Howse, J.: The semantics of augmented constraint diagrams. Journal of Visual Languages and Computing 16, 541–573 (2005)
26. Fish, A., Flower, J., Howse, J.: The semantics of augmented constraint diagrams. Journal of Visual Languages and Computing 16, 541–573 (2005)
27. Frithjof Dau, P.E.: A diagrammatic reasoning system for the description logic $\mathcal{ALC}$. Journal of Visual Languages and Computing (2008) (to be published)
28. Gaines, B.R.: An interactive visual language for term subsumption languages. In: IJCAI, pp. 817–823 (1991)
29. Ganter, B., Mineau, G.W. (eds.): ICCS 2000. LNCS (LNAI), vol. 1867. Springer, Heidelberg (2000)
30. Hammer, E.: Logic and Visual Information. CSLI Publications (1995)
31. Hartshorne, C., Weiss, P., Burks, A.W.: Collected Papers of Charles Sanders Peirce. Harvard University Press, Cambridge (1931–1935)
32. Hayes, P., Eskridge, T., Saavedra, R., Reichherzer, T., Mehrotra, M., Bobrovnikoff, D.: Collaborative knowledge capture in ontologies. In: Proceedings of the 3rd International Conference on Knowledge Capture, pp. 99–106 (2005)
33. Houser, N., Roberts, D.D., Van Evra, J.(eds.): Studies in the Logic of Charles Sanders Peirce. Indiana University Press (1997)
34. Howse, J., Molina, F., Shin, S.-J., Taylor, J.: On diagram tokens and types. In: Hegarty, M., Meyer, B., Narayanan, N.H. (eds.) Diagrams 2002. LNCS, vol. 2317, pp. 146–160. Springer, Heidelberg (2002)
35. Howse, J., Molina, F., Taylor, J.: On the completeness and expressiveness of spider diagram systems. In: Anderson, M., Cheng, P., Haarslev, V. (eds.) Diagrams 2000. LNCS, vol. 1889, pp. 26–41. Springer, Heidelberg (2000)
36. Howse, J., Molina, F., Taylor, J., Kent, S., Gil, J.: Spider diagrams: A diagrammatic reasoning system. Journal of Visual Languages and Computing 12(3), 299–324 (2001)
37. Howse, J., Schuman, S.: Precise visual modelling. Journal of Software and Systems Modeling 4, 310–325 (2005)
38. Howse, J., Stapleton, G., Taylor, J.: Spider diagrams. LMS Journal of Computation and Mathematics 8, 145–194 (2005)
39. Chein, M., Aubert, J.-P., Baget, J.-F.: Simple conceptual graphs and simple concept graphs. In: Øhrstrøm, et al. [58], pp. 87–101
40. John, C.: Reasoning with projected contours. In: Blackwell, A.F., Marriott, K., Shimojima, A. (eds.) Diagrams 2004. LNCS (LNAI), vol. 2980, pp. 147–150. Springer, Heidelberg (2004)

41. Kent, S.: Constraint diagrams: Visualizing invariants in object oriented modelling. In: Proceedings of OOPSLA 1997, pp. 327–341. ACM Press, New York (1997)
42. Kent, S.: Constraint diagrams: Visualizing assertions in object-oriented models. In: OOPSLA, pp. 327–341. ACM Press, New York (1997)
43. Kerdiles, G.N.: Saying it with Pictures: A Logical Landscape of Conceptual Graphs. ILLC Dissertation Series, DS 2001-09 (2001)
44. Kim, S.-K., Carrington, D.: Visualization of formal specifications. In: 6th Aisa Pacific Software Engineering Conference, pp. 102–109. IEEE Computer Society Press, Los Alamitos (1999)
45. Klinger, J.: Simple semiconcept graphs: A boolean logic approach. In: Delugach and Stumme [22]
46. Klinger, J.: Semiconcept graphs with variables. In: Priss, U., Corbett, D.R., Angelova, G. (eds.) ICCS 2002. LNCS (LNAI), vol. 2393, p. 369. Springer, Heidelberg (2002)
47. Klinger, J.: The Logic System of Protoconcept Graphs. Dissertation, Darmstadt University of Technology. Shaker Verlag, Aachen (2005)
48. Kremer, R.: Visual languages for konwledge representation. In: Proc. of 11th Workshop on Knowledge Acquisition, Modeling and Management (KAW 1998), Banff, Alberta, Canada. Morgan Kaufmann, San Francisco (1998)
49. Larkin, J.H., Simon, H.A.: Why a diagram is (sometimes) worth ten thousand words. Cognitive Science 11(1), 65–100 (1987)
50. Leclère, M., Mugnier, M.-L.: Simple conceptual graphs with atomic negation and difference. In: Øhrstrøm, et al. [58], pp. 331–345
51. Lovdahl, J.: Towards a Visual Editing Environment for the Languages of the Semantic Web. PhD thesis, Linkoping University (2002)
52. Mugnier, M.-L.: Knowledge representation and reasonings based on graph homomophism. In: Ganter and Mineau [29], pp. 172–192
53. Mugnier, M.-L., Chein, M. (eds.): ICCS 1998. LNCS (LNAI), vol. 1453. Springer, Heidelberg (1998)
54. Mugnier, M.-L., Leclère, M.: On querying simple conceptual graphs with negation. Data Knowl. Eng. 60(3), 468–493 (2007)
55. Nardi, D., Brachman, R.J.: An introduction to description logics. In: Baader, et al [1], pp. 1–40
56. Nosek, J.T., Roth, I.: A comparison of formal knowledge representation schemes as communication tools: Predicate logic vs semantic network. International Journal of Man-Machine Studies 33(2), 227–239 (1990)
57. Oberlander, J.: Grice for graphics: pragmatic implicature in network diagrams. Information Design Journal 8(6), 163–179 (1996)
58. Øhrstrøm, P., Schärfe, H., Hitzler, P. (eds.): ICCS 2006. LNCS, vol. 4068. Springer, Heidelberg (2006)
59. Peirce, C.: Collected Papers, vol. 4. Harvard University Press (1933)
60. Peirce, C.S.: Reasoning and the logic of things. In: Kremer, K.L., Putnam, H. (eds.) The Cambridge Conferences Lectures of 1898, Harvard Univ. Press, Cambridge (1992)
61. Prediger, S.: Kontextuelle Urteilslogik mit Begriffsgraphen – Ein Beitrag zur Restrukturierung der Mathematischen Logik. Dissertation, Darmstadt University of Technology. Shaker Verlag, Aachen (1998)
62. Prediger, S.: Simple concept graphs: A logic approach. In: Mugnier and Chein [53], pp. 225–239
63. Prediger, S.: Nested concept graphs and triadic power context families: A situation–based contextual approach. In: Ganter and Mineau [29], pp. 263–276.

64. Reasoning with Diagrams (2006), http://www.cs.kent.ac.uk/projects/rwd/
65. Roberts, D.D.: The Existential Graphs of Charles S. Peirce. Mouton, The Hague (1973)
66. Shimojima, A.: On the Efficacy of Representation. PhD thesis, The Department of Philosophy, Indiana University (1996), http://www.jaist.ac.jp/ashimoji/e-papers.html
67. Shin, S.-J.: The Logical Status of Diagrams. Cambridge University Press, Cambridge (1994)
68. Shin, S.-J.: The Iconic Logic of Peirce's Graphs. Bradford Book, Massachusetts (2002)
69. Simonet, G.: Two fol-semantics for simple and nested conceptual graphs. In: Mugnier and Chein [53], pp. 240–254.
70. Sowa, J.F.: Conceptual structures: information processing in mind and machine. Addison-Wesley, Reading (1984)
71. Stapleton, G., Masthoff, J., Flower, J., Fish, A., Southern, J.: Automated theorem proving in Euler diagrams systems. Submitted to Journal of Automated Reasoning (2005)
72. Stapleton, G., Thompson, S., Howse, J., Taylor, J.: The expressiveness of spider diagrams. Journal of Logic and Computation 14(6), 857–880 (2004)
73. Stapleton, G.: Reasoning with Constraint Diagrams. PhD thesis, Visual Modelling Group, Department of Mathematical Sciences, University of Brighton (2004), http://www.cmis.brighton.ac.uk/Research/vmg/GStapletonthesis.html
74. Stapleton, G., Howse, J.: Enhancing the expressiveness of spider diagram systems. In: Accepted for Distributed Multimedia Systems, International Workshop on Visual Languages and Computings, Grand Canyon, USA, Knowledge Systems Institute (2006)
75. Stapleton, G., Howse, J., Taylor, J.: Spider diagrams. LMS Journal of Computation and Mathematics 8, 145–194 (2005); J Howse: This is the definitive spider diagrams paper but is a little technical in places
76. Swoboda, N., Allwein, G.: Heterogeneous reasoning with Euler/Venn diagrams containing named constants and FOL. In: Proceedings of Euler Diagrams 2004. ENTCS, vol. 134. Elsevier Science, Amsterdam (2005)
77. Venn, J.: On the diagrammatic and mechanical representation of propositions and reasonings. Phil. Mag. (1880)
78. Wille, R.: Restructuring mathematical logic: An approach based on peirce's pragmatism. In: Ursini, A., Agliano, P. (eds.) Logic and Algebra, pp. 267–281. Marcel Dekker, New York (1996)
79. Wolff, K.E., Pfeiffer, H.D., Delugach, H.S. (eds.): ICCS 2004. LNCS, vol. 3127. Springer, Heidelberg (2004)
80. Zeman, J.J.: The Graphical Logic of C. S. Peirce. PhD thesis, University of Chicago (1964), http://www.clas.ufl.edu/users/jzeman/
81. Zhao, Y., Lövdahl, J.: A reuse based method of developing the ontology for e-procurement. In: Proceedings of the Nordic Conference on Web Services, pp. 101–112 (2003)

The Logic of Learning

Luc De Raedt

Department of Computer Science, Katholieke Universiteit Leuven,
Celestijnenlaan 200A, BE-3001 Heverlee, Belgium

Abstract. I use the term logical and relational learning (LRL) to refer to the subfield of machine learning and data mining that is concerned with learning in expressive logical or relational representations. It is the union of inductive logic programming, (statistical) relational learning and multi-relational data mining and constitutes a general class of techniques and methodology for learning from structured data (such as graphs, networks, relational databases) and background knowledge. During the course of its existence, logical and relational learning has changed dramatically. Whereas early work was mainly concerned with logical issues (and even program synthesis from examples), in the 90s its focus was on the discovery of new and interpretable knowledge from structured data, often in the form of rules or patterns. Since then the range of tasks to which logical and relational learning has been applied has significantly broadened and now covers almost all machine learning problems and settings. Today, there exist logical and relational learning methods for reinforcement learning, statistical learning, distance- and kernel-based learning in addition to traditional symbolic machine learning approaches. At the same time, logical and relational learning problems are appearing everywhere. Advances in intelligent systems are enabling the generation of high-level symbolic and structured data in a wide variety of domains, including the semantic web, robotics, vision, social networks, and the life sciences, which in turn raises new challenges and opportunities for logical and relational learning.

In this talk, I will start by providing a gentle introduction to the field of logical and relational learning and then continue with an overview of the logical foundations for learning, which are concerned with various settings for learning (such as learning from entailment, from interpretations and from proofs), with the logical inference rules and generality relationships used, and the effects they have on the properties of the search-space. More details can be found in

Luc De Raedt. Logical and Relational Learning. Springer. 2008.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, p. 57, 2009.

What Can Formal Concept Analysis Do for Data Warehouses?*

Rokia Missaoui and Léonard Kwuida

Département d'informatique et d'ingénierie
Université du Québec en Outaouais
C.P. 1250, succursale B, Gatineau (Québec) Canada, J8X 3X7
{rokia.missaoui,leonard.kwuida}@uqo.ca

Abstract. Formal concept analysis (FCA) has been successfully used in several Computer Science fields such as databases, software engineering, and information retrieval, and in many domains like medicine, psychology, linguistics and ecology. In data warehouses, users exploit data hypercubes (i.e., multi-way tables) mainly through online analytical processing (OLAP) techniques to extract useful information from data for decision support purposes.

Many topics have attracted researchers in the area of data warehousing: data warehouse design and multidimensional modeling, efficient cube materialization (pre-computation), physical data organization, query optimization and approximation, discovery-driven data exploration as well as cube compression and mining. Recently, there has been an increasing interest to apply or adapt data mining approaches and advanced statistical analysis techniques for extracting knowledge (e.g., outliers, clusters, rules, closed n-sets) from multidimensional data. Such approaches or techniques cover (but are not limited to) FCA, cluster analysis, principal component analysis, log-linear modeling, and non-negative multiway array factorization. Since data cubes are generally large and highly dimensional, and since cells contain consolidated (e.g., mean value), multidimensional and temporal data, such facts lead to challenging research issues in mining data cubes. In this presentation, we will give an overview of related work and show how FCA theory (with possible extensions) can be used to extract valuable and actionable knowledge from data warehouses.

1 Introduction

A data warehouse (DW) is an integration of consolidated and non volatile data from multiple and possibly heterogeneous data sources for the purpose of decision support making. It contains a collection of data cubes which can be exploited via online analytical processing (OLAP) operations such as roll-up (increase in the aggregation level) and drill-down (decrease in the aggregation level) according to one or more dimension hierarchies, slice (selection), dice (projection), and

* Partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC).

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 58–65, 2009.

Table 1. A data cube

Product	City=Montreal			City=Ottawa		
	Year 2005	2006	2007	2005	2006	2007
P1	10	30	25	10	35	40
P2	15	16	15.5	40	20	35
P3	99	44	66	60	77	44

pivot (rotation) [12]. In a multidimensional context with a set of dimensions, a dimension (e.g., product, location, time) is a descriptive axis for data presentation under several perspectives. A dimension hierarchy contains levels, which organize data into a logical structure (e.g., country, state and city for the location dimension). A member (modality) of a dimension is one of the data values for a given hierarchy level of that dimension. A fact table contains numerical measures and keys relating facts to dimension tables. A cube $\mathbf{X} = \langle D, M \rangle$ is a visual representation of a fact table, where D is a set of n dimensions of the cube (with associated hierarchies) and M its corresponding measures. The following table gives the total sales according to product, location and time.

The presentation will be organized as follows. First, we give an overview of the main issues in data warehousing applications and related advancements. We then see how some of these issues have been tackled using FCA. Finally, we discuss the coupling of FCA with Statistics so that some highlighted issues in data warehousing technology could benefit from suggested couplings.

2 Main Issues in DW

Many topics have attracted researchers in the area of data warehousing: data warehouse design and multidimensional modeling [1], data integration and cleaning, cube decomposition and summarization, materialized view selection, efficient OLAP query optimization, discovery-driven exploration of cubes, cube reorganization, data mining in cubes, and so on. In order to avoid computing a whole data cube, many studies have focused on iceberg cube calculation [37], partial materialization of data cubes [16], semantic summarization of cubes (e.g., quotient cubes [22] and closed cubes [7]), and approximation of cube computation [30]. Recently, there has been an increasing interest for applying/adapting data mining techniques and advanced statistical analysis (e.g., cluster analysis, principal component analysis, log-linear modeling) for knowledge discovery [25,26,29] and data compression purposes in data cubes [2,3,4,27].

In this presentation, we will focus on physical data organization as well as materialization, approximation/compression, summarization and mining of data cubes.

2.1 Optimal Physical Data Organization

Given a query workload on a database or data warehouse and a use-defined storage constraint, the objective of "optimal" physical data organization is to identify the "best" physical data organization (e.g., data indexing and clustering), i.e. the one that ensures a global optimization of the workload [11]. When the physical organization relies mainly on indexing techniques, the optimization problem consists to identify the optimal set of indices (on table attributes) that need to be implemented to speed up query execution. Indices reduce significantly the cost of processing complex queries, but induce an overhead in terms of index storage and maintenance. To optimize star join queries (i.e., joins between a fact table and multiple dimension tables), bitmap and join indices (or a combination of the two types) are commonly used. However, selecting these indices is a difficult task due to the exponential number of candidate attributes to be indexed. Most of approaches for index selection follow two main steps: (i) pruning the search space (i.e., reducing the number of candidate attributes) and (ii) selecting indices from the pruned search space using a cost model and statistics about data. In [5], the first step of index selection exploits alternatively two existing algorithms for closed itemset generation to identify candidate attributes to be indexed while the second step relies on a greedy algorithm to select bitmap join indices that minimize processing and storage costs. A related topic is the (horizontal and vertical) fragmentation of data across a network in distributed data warehouses.

2.2 Cube Materialization

From a data cube of n dimensions one can generate a lattice of 2^n cuboids where the 0-D cuboid (or apex cuboid) represents the highest summarization while the $p - D$ cuboids represent the possible combinations of p out of n dimensions. If each dimension D_i has a corresponding hierarchy with L_i levels, then the number of possible cuboids of a data cube of n dimensions is $\Pi_{i=1,n}(L_i + 1)$. Since the number of cuboids exponentially grows according to the number of dimensions and the size of dimension hierarchies, partial materialization of cuboids is requested rather than a full one. However, the identification of the "optimal" set of cuboids to materialize is an NP-hard problem similar to the problem of index selection and hence most of the proposed techniques rely in part on heuristics and pruning techniques. The question is the following: can FCA be exploited to select the cuboids to be materialized by considering concept intents (itemsets) as combinations of dimensions with some specific features (e.g., frequency, cost)?

Besides cube materialization, one is faced with the following related issues: (i) how the materialized cuboids can be exploited for efficiently process queries? (ii) how and when the set of materialized cuboids should be revised when the initial query workload changes over time? and (iii) how to propagate data changes to materialized cuboids?

2.3 Cube Approximation and Compression

Approximate query answering in data warehouses has been used in order to accelerate aggregate computation and query execution at the expense of some information loss. Existing work has been conducted based mainly on sampling

[2], clustering [38], wavelets [10] or maximum entropy principle [28]. Palpanas *et al.* [28] propose an approach based on information entropy to (i) detect deviations, and (ii) estimate the original multidimensional data from aggregates for approximate query answering purposes. A wavelet-based approach is used in [10] to approximate query answering, and proves to be more effective than sampling techniques. In a similar spirit, [30] uses the probability density distribution of data in order to propose a compressed representation of data cubes which reduces data storage and leads to approximate answers to aggregate queries. In [29], an approach based on log-linear modeling is used to identify exceptions in data cubes by comparing anticipated cell values against actual values. In [3,4], log-linear modeling is used for data compression. In [27], two kinds of probabilistic models, namely log-linear models and non negative multi-way array factorization are used for data approximation through parsimonious model selection, data mining through component (cluster), outlier and association extraction, and approximate query answering.

2.4 Cube Mining

Substantial work has been conducted for data mining in data warehouses [15]. This includes (but is not limited to) outlier detection in multidimensional data [21], cubegrade generation [17], constrained gradient analysis [13], association rule mining, and discovery-driven examination of cubes [29]. To the best of our knowledge, Kamber *et al.* [20] were the first who addressed the issue of mining association rules from multidimensional data. They introduced the concept of *metarule-guided mining* which consists in using rule templates defined by users in order to guide the mining process. Cubegrades are a generalization of association rules and express the significant changes that affect measures when a cube is modified through specialization (drill-down), generalization (roll-up) or mutation (switch). Tjioe and Taniar [32] propose a method for extracting associations from multiple dimensions at multiple levels of abstraction by focusing on summarized data according to the COUNT measure. In order to do so, they prepare multidimensional data for the mining process according to a set of algorithms which prune all rows in the fact table which have an aggregate value less than a predefined one.

3 FCA and Data Warehousing Technology

To date, FCA has been used to mainly handle the following topics of the data warehousing technology:

- Cube visualization using nested line diagrams [31]
- Cube summarization [8,24]
- computation of closed patterns in 3-D [18,19] and $n-$ary relations [9]

In [31], the design of conceptual information systems is described and the utilization of visualization techniques for their exploration is discussed with an application to OLAP-based systems. In order to handle the problem of data cube computation and manage data storage and query processing in an efficient way, Lakshmanan *et al.* [22] have introduced the notion of *quotient cube* as a succinct summary of a data cube that saves the roll-up/drill-down semantics. Such a cube is produced through a partitioning method (not related to FCA) that groups cube cells into equivalent classes. To propagate data changes to a pre-computed quotient cube, incremental algorithms based on concept lattice construction are proposed in [24]. Recently, Casali *et al.* [8] have proposed a lattice-based approach which computes the cube closure over a multidimensional search space (a cube lattice), and define the notion of *closed cube lattice* which is isomorphic to both the concept lattice and the quotient cube. The latter can be derived from the closed cube given a closure-based characterization. In [18,19], approaches towards generating closed sets from 3-D relations are proposed. Based on the theory of *trilattices* [23] as concept lattices constructed from triadic contexts, the work in [18] proposes an algorithm called Trias which extracts tri-concepts from ternary relations. With the same objective in mind, Ji *et al.* [19] propose CubeMiner as an algorithm for generating closed ternary patterns from ternary tables by splitting the data collection into smaller groups and frequently checking the unicity of generated closed patterns. In [9], a constrained-based mining procedure named Data-Peeler extracts from $n-$ary (multidimensional) tables all closed $n-$sets that satisfy piece-wise (anti)-monotonic constraints.

4 What Can Statistics Do for Data Warehouses and Formal Concept Analysis

In large data collections in general and in data warehouses in particular, one complex task is to reduce the dimensionality of data before conducting any data analysis and mining task. This is necessary to help the user get a human readable knowledge from the data. For this purpose, statistical and machine learning methods have been proposed based either on feature selection (e.g., fractals) or data extraction (e.g., principal component analysis). The challenge is to reduce the dimensionality without significant loss in data. The questions to be raised are as follows: what can these methods bring to FCA? Can they be combined with the FCA methods? The approach from the FCA side for multidimensional data analysis consists to use trilattices [23] or in general polyadic FCA [34].

In [36], Wolff compares some methods of graphical data analysis. These are factor analysis, principal component analysis, correspondence analysis, cluster analysis, multidimensional scaling, partial order scalogram analysis and FCA (line diagrams). All these methods, except FCA, use a projection in the plane that leads to some loss of information. In [14], the authors focus on a comparison between correspondence analysis and FCA and notice that *the exact representation* of data obviously has the drawback that even small many-valued contexts may have large concept lattices. However, nested line diagrams and atlas decompositions are useful to display concept lattices of a reasonable size. For large size collections, a preprocessing step is needed to prune or combine some attributes or objects to get “generalized” ones even before dimension reduction.

The combinatorial explosion of the size of concept and implication (or association rule) sets is not the only limitation of FCA. Another weak point from exact representation is its vulnerability to even small changes on the data. Towards settling this problem, there are many attempts: association rules have been introduced, based on some statistical measures, as an alternative to exact rules (implications); Iceberg concepts or frequent closed itemsets are proposed to prune the concept set; Alpha Galois concepts [33] and fault-tolerant patterns [6] are proposed as replacement for concepts.

From these observations, we expect that combining FCA and statistics might be a good compromise and will be useful to data warehousing technology as well as FCA. In [35] the author shows how Formal Concept Analysis and Statistics can benefit from each other strengths. Now, how can we combine FCA and Statistics? One approach could be to analyze statistical methods and develop their FCA counterparts, or to develop conceptual foundations of Statistics. This includes the study of how data analysis techniques such as biplot, principal component and correspondence analysis can be coupled with FCA techniques to produce concept lattices of factors where a factor is a combination of attributes. Concerning conceptual foundations of Statistics, the efforts will be put on investigating how the carrier set in measure theory (σ-algebras) can be replaced by complete lattices (carrier set of concepts). This should be intimately related to valuations on lattices.

References

1. Agrawal, R., Gupta, A., Sarawagi, S.: Modeling multidimensional databases. In: ICDE 1997: Proceedings of the Thirteenth International Conference on Data Engineering, Washington, DC, USA, 1997, pp. 232–243. IEEE Computer Society Press, Los Alamitos (1997)
2. Babcock, B., Chaudhuri, S., Das, G.: Dynamic sample selection for approximate query processing. In: SIGMOD 2003: Proceedings of the 2003 ACM SIGMOD international conference on Management of data, pp. 539–550. ACM Press, New York (2003)
3. Barbará, D., Wu, X.: Using loglinear models to compress datacubes. In: Lu, H., Zhou, A. (eds.) WAIM 2000. LNCS, vol. 1846, pp. 311–323. Springer, Heidelberg (2000)
4. Barbara, D., Wu, X.: Loglinear-based quasi cubes. J. Intell. Inf. Syst. 16(3), 255–276 (2001)
5. Bellatreche, L., Missaoui, R., Necir, H., Drias, H.: A data mining approach for selecting bitmap join indices. Journal of Computing Science and Engineering 1(2), 177–194 (2007)
6. Besson, J., Robardet, C., Boulicaut, J.-F.: Mining a new fault-tolerant pattern type as an alternative to formal concept discovery. In: Schärfe, H., Hitzler, P., Øhrstrøm, P. (eds.) ICCS 2006. LNCS, vol. 4068, pp. 144–157. Springer, Heidelberg (2006)
7. Casali, A., Nedjar, S., Cicchetti, R., Lakhal, L.: Convex cube: Towards a unified structure for multidimensional databases. In: Wagner, R., Revell, N., Pernul, G. (eds.) DEXA 2007. LNCS, vol. 4653, pp. 572–581. Springer, Heidelberg (2007)

8. Casali, A., Nedjar, S., Cicchetti, R., Lakhal, L.: Closed Cube Lattices. In: New Trends in Data Warehousing and Data Analysis. Annals of Information Systems, vol. 3, pp. 1–20. Springer, Heidelberg (2009)
9. Cerf, L., Besson, J., Robardet, C., Boulicaut, J.-F.: Data peeler: Constraint-based closed pattern mining in n-ary relations. In: SDM, pp. 37–48. SIAM, Philadelphia (2008)
10. Chakrabarti, K., Garofalakis, M.N., Rastogi, R., Shim, K.: Approximate query processing using wavelets. VLDB J. 10(2-3), 199–223 (2001)
11. Chaudhuri, S., Datar, M., Narasayya, V.: Index selection for databases: A hardness study and a principled heuristic solution. IEEE Transactions on Knowledge and Data Engineering 16(11), 1313–1323 (2004)
12. Chaudhuri, S., Dayal, U.: An overview of data warehousing and olap technology. SIGMOD Rec. 26(1), 65–74 (1997)
13. Dong, G., Han, J., Lam, J.M.W., Pei, J., Wang, K.: Mining multi-dimensional constrained gradients in data cubes. In: VLDB 2001: Proceedings of the 27th International Conference on Very Large Data Bases, pp. 321–330. Morgan Kaufmann Publishers Inc., San Francisco (2001)
14. Gabler, S., Wolff, K.E.: Comparison of visualizations in formal concept analysis and correspondence analysis. In: Greenacre, M., Blasius, J. (eds.) Visualization of Categorical Data, pp. 85–97. Academic Press, San Diego (1998)
15. Han, J., Kamber, M.: Data mining: concepts and techniques. Morgan Kaufmann Publishers Inc., San Francisco (2000)
16. Harinarayan, V., Rajaraman, A., Ullman, J.D.: Implementing data cubes efficiently. In: SIGMOD 1996: Proceedings of the 1996 ACM SIGMOD international conference on Management of data, pp. 205–216. ACM Press, New York (1996)
17. Imielinski, T., Khachiyan, L., Abdulghani, A.: Cubegrades: Generalizing association rules. Data Min. Knowl. Discov. 6(3), 219–257 (2002)
18. Jaeschke, R., Hotho, A., Schmitz, C., Ganter, B., Stumme, G.: Trias - an algorithm for mining iceberg tri-lattices. In: Proceedings of the 6th IEEE International Conference on Data Mining (ICDM 2006), Hong Kong, December 2006, pp. 907–911. IEEE Computer Society Press, Los Alamitos (2006)
19. Ji, L., Tan, K.-L., Tung, A.K.H.: Mining frequent closed cubes in 3d datasets. In: VLDB 2006: Proceedings of the 32nd international conference on Very large data bases, pp. 811–822. VLDB Endowment (2006)
20. Kamber, M., Han, J., Chiang, J.: Metarule-Guided Mining of Multi-Dimensional Association Rules Using Data Cubes. In: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD 1997), Newport Beach, CA, USA, August 1997, pp. 207–210. The AAAI Press, Menlo Park (1997)
21. Knorr, E.M., Ng, R.T., Tucakov, V.: Distance-based outliers: algorithms and applications. The VLDB Journal 8(3-4), 237–253 (2000)
22. Lakshmanan, L.V.S., Pei, J., Zhao, Y.: Quotient cube: How to summarize the semantics of a data cube. In: Proceedings of the 28th International Conference on Very Large Databases, VLDB, pp. 778–789 (2002)
23. Lehmann, F., Wille, R.: A triadic approach to formal concept analysis. In: Ellis, G., Rich, W., Levinson, R., Sowa, J.F. (eds.) ICCS 1995. LNCS, vol. 954, pp. 32–43. Springer, Heidelberg (1995)
24. Li, C.-P., Tung, K.-H., Wang, S.: Incremental maintenance of quotient cube based on galois lattice. J. Comput. Sci. Technol. 19(3), 302–308 (2004)
25. Lu, H., Feng, L., Han, J.: Beyond intratransaction association analysis: mining multidimensional intertransaction association rules. ACM Trans. Inf. Syst. 18(4), 423–454 (2000)

26. Messaoud, R.B., Boussaid, O., Rabaséda, S.: A new olap aggregation based on the ahc technique. In: DOLAP 2004: Proceedings of the 7th ACM international workshop on Data warehousing and OLAP, pp. 65–72. ACM Press, New York (2004)
27. Missaoui, R., Goutte, C., Choupo, A.K., Boujenoui, A.: A probabilistic model for data cube compression and query approximation. In: DOLAP 2007: Proceedings of the ACM tenth international workshop on Data warehousing and OLAP, pp. 33–40. ACM Press, New York (2007)
28. Palpanas, T., Koudas, N., Mendelzon, A.: Using datacube aggregates for approximate querying and deviation detection. IEEE Transactions on Knowledge and Data Engineering 17(11), 1465–1477 (2005)
29. Sarawagi, S., Agrawal, R., Megiddo, N.: Discovery-driven exploration of olap data cubes. In: Schek, H.-J., Saltor, F., Ramos, I., Alonso, G. (eds.) EDBT 1998. LNCS, vol. 1377, pp. 168–182. Springer, Heidelberg (1998)
30. Shanmugasundaram, J., Fayyad, U., Bradley, P.S.: Compressed data cubes for olap aggregate query approximation on continuous dimensions. In: KDD 1999: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 223–232. ACM Press, New York (1999)
31. Stumme, G.: Conceptual on-line analytical processing. In: Information organization and databases: foundations of data organization, pp. 191–203 (2000)
32. Tjioe, H.C., Taniar, D.: Mining association rules in data warehouses. International Journal of Data Warehousing and Mining 1(3), 28–62 (2005)
33. Ventos, V., Soldano, H.: Alpha galois lattices: an overview. In: Ganter, B., Godin, R. (eds.) ICFCA 2005. LNCS, vol. 3403, pp. 298–313. Springer, Heidelberg (2005)
34. Voutsadakis, G.: Polyadic concept analysis. Order 19(3), 295–304 (2002)
35. White, D.R.: Statistical entailments and the galois lattice. Social Networks 18, 201–215 (1996)
36. Wolff, K.E.: Comparison of graphical data analysis methods. In: Faulbaum, F., Bandilla, W. (eds.) SoftStat 1995. Advances in Statistical Software, vol. 5, Lucius&Lucius, Stuttgart, pp. 139–151 (1996)
37. Xin, D., Han, J., Li, X., Wah, B.W.: Star-cubing: Computing iceberg cubes by top-down and bottom-up integration. In: VLDB (2003)
38. Yu, F., Shan, W.: Compressed data cube for approximate olap query processing. J. Comput. Sci. Technol. 17(5), 625–635 (2002)

Time and Logic: A.N. Prior's Formal Analysis of Temporal Concepts

Peter Øhrstrøm

Department of Communication and Psychology, Aalborg University, Denmark
poe@hum.aau.dk

Abstract. This paper deals with A.N. Prior's analysis of the concepts of dynamic and static time, i.e., McTaggart's so-called A- and B-concepts. The relations and mutual dependencies between these temporal concepts are investigated, and Prior's response to McTaggart's views is discussed. Furthermore, Prior's notion of branching time is analysed. It is argued that Prior can be criticized for identifying 'plain future' with 'necessary future'. Finally, Prior's four grades of tense-logical involvement are introduced and discussed. It is argued that the third grade is the most attractive from a philosophical point of view.

Keywords: Temporal logic, dynamic and static time, A- and B-concepts, tense logic, A.N. Prior.

1 Introduction

The 20th century has seen a very important revival of the formal studies of temporal concepts. The most important contribution to the modern logic of time was made in the 1950s and 1960s by A. N. Prior (1914–1969). In his endeavours, Prior was greatly inspired by ancient and medieval thinkers and especially their work on time and logic. In fact, his introduction of modern temporal logic may be conceived as a rediscovery and a reformulation of the ancient and medieval logic of time. Following this long tradition, Prior held that logic should include the study of temporal reasoning.

Prior pointed out that when discussing the temporal aspects of reality we use two different conceptual frameworks, the A-concepts (corresponding to dynamic time) and the B-concepts (corresponding to static time). These two sets of concepts had been introduced by J.M.E. McTaggart (1866-1925). In his temporal logic, Prior logically analysed the tension between McTaggart's A-concepts and B-concepts. In addition, Prior introduced an elaborated notion of branching time, and using this notion, he demonstrated that there are four different ways to answer the fundamental questions about the tension between A- and B-concepts.

It turns out that this discussion of temporal logic is closely related to the ancient and medieval debates on the fundamental problems in philosophy regarding human freedom and (in)determinism. Using the idea of branching time Prior demonstrated that there are in fact models of time which are logically consistent with his ideas of free choice and indeterminism.

After Prior's founding work in temporal logic, a number of important concepts have been studied within this framework. It has turned out that Prior's temporal logic

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 66–81, 2009.

is essential for the formal and logical analysis of important philosophical ideas such as 'free choice', ethical responsibility, indeterminism, and 'the passage of time'. The introduction of time into logic has also led to the development of formal systems which are particularly well suited to representing and studying temporal phenomena in computer science such as temporal aspects of program execution, temporal databases, and time in natural language processing.

In section 2 of this paper McTaggart's ideas on dynamic and static time will be introduced. Prior's response to these ideas will be discussed in section 3. The various ideas of branching time will be discussed in section 4, and in section 5 I intend to discuss Prior's use of these ideas of branching time in his formal formulation of the four grades of tense-logical involvement. It will be argued that Prior's analysis of McTaggart's ideas on time, as well as his further elaboration of the ideas constitutes an interesting framework which can be useful within the philosophical study of time as well as within the formal studies of temporal relationships and processes.

2 McTaggart's Ideas on Dynamic and Static Time

In his famous paper, "The Unreality of Time" [1908], J.M.E. McTaggart offered an early discussion of tensions between dynamic and static time and of the relations between time and tense. He also offered a paradox which in fact gave rise to an argument in favour of the unreality of time. However, McTaggart's ideas had no significant role to play in relation to temporal logic before Prior published his analysis of the paradox, and it would be misleading to see these early ideas as a proper theory of tenses. McTaggart's ideas were carefully discussed in C.D. Broad's "Examination of McTaggart's Philosophy" published in [1938], and the analysis of McTaggart's so-called paradox became very important in the philosophical debate about time in the 1970s and later mainly because of Prior's analysis, which was in fact partly influenced by Broad's work. In particular, the debate has turned out to be crucial when it comes to an understanding of the relations between time and tense.

The distinction between the logic of tenses and the logic of earlier and later (in terms of instants or in terms of durations) is essential for the understanding of modern temporal logic. This distinction was introduced by McTaggart in his famous paper mentioned above [1908], which was restated in his book *The Nature of Existence* [1927]. In this paper McTaggart suggested the distinction between the so-called A- and B-series conceptions of time. According to the A-series conception, the tenses (past, present, and future) are the key notions for a proper understanding of time, whereas the earlier-later calculus is secondary. According to the B-series conception time is understood as a set of instants organized by the earlier-later relation, whereas the tenses are secondary. McTaggart explicitly identified and discussed this basic dichotomy between the A-series and the B-series.

McTaggart himself arrived at the conclusion that A-concepts are more fundamental than B-concepts. He did not, however, use this analysis as an argument in favour of A-theory. On the contrary, he used it for a refutation of the reality of time. He argued that A-concepts give rise to a contradiction - which has become known as 'McTaggart's Paradox'. Due to this putative contradiction within the fundamental conceptualisation of time, he went on to claim that time is not real, since the idea of time understood in terms of the A-concepts on his view turns out to lead to a contradiction.

The core of McTaggart's argument is that the notions of 'past', 'present' and 'future' are incompatible predicates which are nevertheless applicable to all events. In his own words:

> Past, present, and future are incompatible determinations… But every event has them all. If M is past, it has been present and future. If it is future, it will be present and past. If it is present, it has been future and will be past. Thus all three characteristics belong to each event. How is this consistent with their being incompatible? [McTaggart 1927, in Poidevin and MacBeath p.32]

McTaggrt's argument is based on the following three assumptions:

(1) The A-concepts ('past', 'present', and 'future') are real.
(2) The three predicates ('past', 'present', and 'future') are supposed to be mutually exclusive – given that any concrete event happens just once (even though a type of event may be repeated).
(3) Any of the three predicates can be applied to any event.

In a book on history, it makes sense to speak of 'the death of Queen Anne' as a past event - call it e_1. However, according to (3) e_1 can also be future. This could be the case if the event is discussed in a document written in the lifetime of Queen Anne. In such a document it could certainly well make sense to speak about her death as a future event. Apparently, according to (2) this gives rise to an inconsistency, since how can e_1 be both past and future – and present as well, by a similar argument? The answer must be that there is another event e_2, relative to which for instance e_1 has been present and future, and is going to be past. Now, the same kind of apparent inconsistency can be established with respect to e_2, and the problem can only be solved by introducing a new event e_3, for which a new apparent inconsistency will arise etc. - which seems to mean that we have to go on ad infinitum in order to solve the inconsistency. The consequence appears to be that the inconsistency can never be resolved. McTaggart concludes that it must be a mistake to use of the A-concepts as describing something real about the events.

3 Prior's Response to McTaggart's Ideas

At first Prior did not consider the writings of McTaggart to be relevant in relation to the ideas of temporal logic. However, later he clearly realised the importance of McTaggart's distinctions in relation to the basic ideas in temporal logic. Since then, the notions and arguments in McTaggart's paper have become an important ingredient, of all major treatments of the philosophical problems related to the various kinds of temporal logic and their mutual relations.

It was Peter Geach who sometime in the early 1960s made Prior aware of the importance and relevance of McTaggart's distinction between the so-called A- and B-series conceptions of time [Prior 1967, p. vi]. Prior's interest in McTaggart's observations was first aroused when he realised that McTaggart had offered an argument to the effect that the B-series presupposes the A-series rather than vice versa [1967, p.2].

Prior was particularly concerned with McTaggart's argument against the reality of tenses. He pointed out that the argument is in fact based on one crucial assumption, which can certainly be questioned. In his own words: "McTaggart's underlying assumption, which generates each of the moves that lead us to a contradiction, appears to be that 'has been', 'will be', and the strictly present-tense 'is' *must* be explicated in terms of a *non*-temporal 'is' attaching either an event or a 'moment' to a 'moment'. McTaggart himself observes, however, that 'propositions which deal with the place of anything in the A-series such as 'The battle of Waterloo is in the past', and 'It is now raining', are of a kind which can be 'sometimes true and sometimes false'. The 'is' that occurs in such propositions therefore cannot be non-temporal." [1967, p.6] In this way Prior's analysis clearly demonstrates that McTaggart's assumption is in fact very controversial. Nevertheless, Prior's studies brought renewed fame to McTaggart's argument.

Prior clearly agreed with McTaggart in rejecting the static view of time, i.e., the view based on the B-concepts. In Prior's opinion, the reality of everything including our own existence in the world should definitely not be conceived as "a timeless tapestry". In his view the static view of time is mistaken. In his own words:

> I believe that what we see as a progress of events is a progress of events, a coming to pass of one thing after another, and not just a timeless tapestry with everything stuck there for good and all... This belief of mine... is bound up with a belief in real freedom. One of the big differences between the past and the future is that once something has become past, it is, as it were, out of our reach - once a thing has happened, nothing we can do can make it not to have happened. But the future is to some extent, even though it is only to a very small extent, something we can make for ourselves.... if something is the work of a free agent, then it wasn't going to be the case until that agent decided that it was. [Copeland 1996, p. 47-8]

During the 20th century there has been much debate concerning A- and B-series conceptions of time and concerning the validity of McTaggart's argument and the various reformulations of it. Some authors like David Mellor have maintained that there is a valid version of the argument, which should in fact force us to reject the tense-logical view of time, i.e., the A-series conception. According to Mellor, nothing in reality has tenses and "the A-series is disproved by a contradiction inherent in the idea that tenses change" [Mellor 1981, p.89]. Others have followed Prior in holding that all versions of McTaggart's argument are flawed. In his careful analysis of McTaggart's paradox, William Lane Craig [2000a, p.169 ff.] has argued that no contradiction need be involved in a proper formalization of the A-series, and it may be concluded that McTaggart's argument is simply misdirected as a refutation of the tensed theory of time [Craig 2000a, p.207]. McTaggart's paradox can be solved if iterated tenses like *PF* and *FP* are introduced. It may be seen as part of McTaggart's argument that in this way we shall need still longer iterated tenses (like *PPF*, *FFPF*, *PPFFP*, …) in order to solve the apparent contradiction, and we thereby have to deal with the problems of an infinite regress of this kind. It is, however, not obvious that any serious logical problem would follow from such an infinite regress. In addition, as Richard Sylvan [1996, p.122] has argued, the construction of iterated tenses in response to McTaggart's argument will probably not give rise to a proper infinite regress since “expressivewise the regress stops” as a consequence of the logical properties of the tense logic in question. The point is, that it is likely to be the case in all relevant tense-logical systems that the number of non-equivalent (iterated) tenses is finite. This statement may be seen as a generalisation of Hamblin’s famous fifteen-theorem for dense time (see [Øhrstrøm and Hasle 1995 p.176 ff.]).

McTaggart’s paradox is clearly related to the problem of truth in temporal logic. What makes a statement like, ‘It is now four o’clock’, true or false? As Poidevin and MacBeath [1993, p.2] have clearly described in their account of modern philosophy of time, this question can be answered in two different ways. The A-theorists say that the statement “It is now four o’clock” is true if and only if the time we have given the name “four o’clock”, is in fact present. The B-theorists, on the other hand, claim that there are no tensed facts. According to their view the statement “It is now four o’clock” is true if and only if the utterance is made at four o’clock. Similarly, the A-theorists claim that the statement “Julius Caesar was killed” is true because Julius Caesar was in fact killed, whereas the B-theorists say that this statement is true because the time of utterance is after the killing of Julius Caesar. In this way the A-theorists hold that tensed statements have tensed truth-conditions, while the B-theorists find that tensed sentences are made true or false by tenseless truth-conditions. In their book Poidevin and MacBeath [1993] have presented A.N. Prior and D.H. Mellor as prominent representatives of respectively the A- and the B-view.

It may be useful to consider the formal aspects of the A- and B-notions a little closer. In order to do so we first of all have to deal with the general features of the tense-logical formulae which are essential for the formulation of the A-series conception. These formulae can be introduced inductively by the following rules of well formed formulae (wff):

(i) any propositional variable is a wff
(ii) if φ is a wff, then $\sim\varphi$ is also a wff
(iii) if φ and ϕ are wffs, then $(\varphi \wedge \phi)$ is also a wff
(iv) if φ is a wff, then $F\varphi$ is also a wff
(v) if φ is a wff, then $P\varphi$ is also a wff
(vi) nothing else is a wff.

As we shall see in the following, an A-theorist would probably at this point like to add a formalism of instant propositions. The B-theorists, on the other hand, would probably emphasise the need for truth-conditions established in terms of a model M = (TIME,<,v), where TIME is a set of temporal elements like instants or durations, < is a binary relation on TIME (corresponding to ‘before’), and v is a valuation function from TIME and the set of propositional variables to {0,1}. The expression $v(t,p)$ is said to be the truth-value of the propositional variable p at t. Given such a model, the notion of truth for any tense-logical formula can be given by the following inductive definition:

$\mathrm{M}, t \models p$ if $v(t,p) = 1$
$\mathrm{M}, t \models \sim\varphi$ if not $\mathrm{M}, t \models \varphi$
$\mathrm{M}, t \models (\varphi \wedge \phi)$ if $\mathrm{M}, t \models \varphi$ and $\mathrm{M}, t \models \phi$
$\mathrm{M}, t \models F\varphi$ if $\mathrm{M}, t' \models \varphi$ for some t' with $t<t'$
$\mathrm{M}, t \models P\varphi$ if $\mathrm{M}, t' \models \varphi$ for some t' with $t'<t$.

If $M,t \models \varphi$ the proposition φ is true at t according to the model M. The B-theorist will emphasise that in this way, truth of the tense-logical formulae of the object language is defined in terms of a tenseless metalanguage. For this reason, the B-theorist will point out that the A-language clearly depends on the B-language. Obviously, the A-theorist has to follow another line of argumentation. It seems that at least two options are open for him.

The first possibility for the A-theorist when he wants to respond to the criticism from the B-theorist was explicitly formulated by A.N. Prior, according to whom there is no sharp distinction between an object language and a metalanguage. Using what is now called a hybrid logic (see Braüner [2002a]), in which the instants are just a special kind of propositions, Prior was able to define $T(t,\varphi)$ (standing for 'φ is true at t') for any tense-logical formula, φ, in terms of the tense-logical language itself. Such instant-propositions describe the world uniquely at any given instant, and are for this reason also called world-state propositions. Like Prior we shall use *a*, *b*, *c*, ... as instant-propositions instead of t_1, t_2, ... In fact, Prior assumed that such propositions *are* what ought to be meant by 'instants':

> A world-state proposition in the tense-logical sense is simply an index of an instant; indeed, I would like to say that it is an instant, in the only sense in which 'instants' are not highly fictitious entities. [Prior 1967, p.188-89]

The traditional distinction between the description of the content and the indication of time for an event is thereby dissolved. Prior listed the following three axioms for instant propositions:

(I1) $\exists a: a$

(I2) $\sim\Box\sim a$

(I3) $\Box(a \supset p) \vee \Box(a \supset \sim p)$

Formally, this means that the symbolic language has to be extended with standard quantification theory and with a necessity operator □ (its dual possibility operator ◊ defined as ~□~).

Intuitively, an instant proposition may be conceived as a conjunction of all propositions belonging to a maximal consistent set of well formed formulae.

Prior argued that there is a basic flaw in McTaggart's argument. According to his view, the contradictions arise from an attempt at forcing the A-series notions into a B-series framework [1967, p.6]. Prior argued that events may be described in terms of instant-propositions, of which it also holds that they 'happen', i.e. are true, exactly once. Using *a* as an arbitrary instant proposition, McTaggart's claim that the three tense-logical predicates are mutually exclusive can be formulated as:

$$a \supset (\sim Pa \wedge \sim Fa)$$
$$Pa \supset (\sim a \wedge \sim Fa)$$
$$Fa \supset (\sim a \wedge \sim Pa)$$

Here *Pa* stands for 'it has been the case that *a*', whereas *Fa* stands for 'it will be the case that *a*'. McTaggart's other claim that any event can be past, present, and future, can be expressed in the following way, where the *I*-operator stands for 'the present':

$$Ia \supset (PFa \wedge FPa)$$
$$Pa \supset (PIa \wedge PFa)$$
$$Fa \supset (FPa \wedge FIa)$$

But no contradiction follows from these 6 theses above. It is thus revealed that McTaggart's paradox is in no way a cogent argument against the A-series notions, let alone the reality of time. Prior concluded that McTaggart's argument could not shake his fundamental belief in the ontological status of the tenses. Prior maintained that tense logic embodied a crucial ontological and epistemological point of view according to which "the tenses (it will be, it was the case) are primitive; only present objects exist" [Prior & Fine, 1977, p.116]. To Prior, the present and the real were one and the same concept. Shortly before he died, he formulated his view in the following way: "...the present simply is the real considered in relation to two particular species of unreality, namely past and future." [Prior 1972, p.320]

The second possibility for the A-theorist when he wants to respond to the criticism from the B-theorist is the use of so-called homophonic theories of truth in which the constructions of the object language are interpreted in terms of analogous constructions of the metalanguage. Torben Braüner [2002a] has demonstrated that tense logics permit the existence of such a homophonic theory of truth, provided that they are stronger than the rather basic tense-logical system K_b. It seems that Prior never attempted to work out the details of a homophonic theory of truth. However, as pointed out by Torben Braüner [2002b], Prior was clearly aware of the possibility of a homophonic theory of truth as it is evident from the following quotation:

> The function of the operator *F*, in short, is that of forming a future-tense statement from the corresponding present-tense one, and the future-tense statement is not about the present tense one, but is about whatever the present tense statement is about. . . . But although the statement 'It will be the case that Professor Carnap is flying to the moon', that is, 'Professor Carnap will be flying to the moon', is not exactly a statement about the statement 'Professor Carnap is flying to the moon', we may say that the future-tense statement is true if and only if the present-tense statement will be true. [Prior, 1957, pp. 8–9]

It seems clear from the extensive debate that a valid version of McTaggart's argument can only be established if some additional philosophical assumptions are made. None of them represent a priori impossible positions. On the other hand, they can all be questioned. For this reason, it may be concluded that it is still logically possible to hold any of the two main positions, the A-theory and the B-theory. In fact, as we shall see in section 5, Prior demonstrated that various relevant variations of the positions should be taken into consideration.

4 The Ideas of Branching Time

The idea of branching time had not yet been formulated in Prior's *Time and Modality* [1957], which otherwise marked the major breakthrough of the new logic of time. As an explicit (or formalised) idea, branching time was first suggested to Prior in a letter from Saul Kripke in September 1958 [now kept at The Prior collection, Bodleian Library, Oxford]. This letter contains an initial version of the idea and a system of branching time, although it was of course not worked out in detail. Kripke suggested that we may consider the present as a point of ‘rank 0’, and possible ‘events’ or ‘states’ at the next moment as points of ‘rank 1’; for every such possible state in turn, there would be various possible future states at the next moment from ‘rank1’, the set of which could be labelled ‘rank2’, and so forth. In this way it is possible to form a tree structure representing the entire set of possible futures expanding from the present (rank 0) - indeed a set of possible futures can be said to be identified for any state, or node in the tree. In this structure every point determines a subtree consisting of its own present and future.

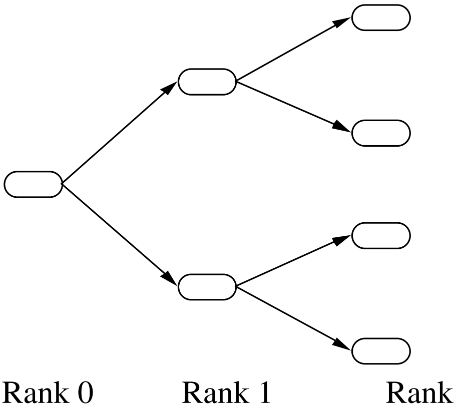


Fig. 1. Branching time according to Saul Kripke, 1958

Prior clearly found this view of time highly interesting, and in the following years he substantially developed it. He worked out the formal details of several different systems, which constitute different and even competing interpretations of this idea, as we shall see below. Eventually, he incorporated the idea of branching into the concept of time itself. In his *Past, Present and Future* [1967] he made extensive use of the idea in the presentation of his tempo-modal systems.

In order to describe the semantics for these systems, Prior [1967, p.126 ff.] needs a notion of ‘chronicles’ or ‘histories’, i.e., maximal and linear subsets in *(TIME,<)*.

It is important to point out that the branching time systems which Prior developed formally are all backwards linear. This means that there is a unique past relative to any moment in the branching time system. There are no ‘alternative pasts’. In this way the asymmetry between past and future is introduced in a very clear manner.

Given Prior’s notion of instant-propositions, the interpretation of the branching time models becomes rather interesting. On this view, every point (or moment) corresponds to an instant-proposition which is in fact maximal in the sense that it implies everything which is true i.e. everything which is, has been, will be or could have been the case. In this sense, every point of the diagram reflects (or includes) the totality of the branching time diagram.

Prior discussed two models of branching time, the Ockhamistic and the Peircean. Prior introduced these names because the Ockhamistic logic is very similar to the

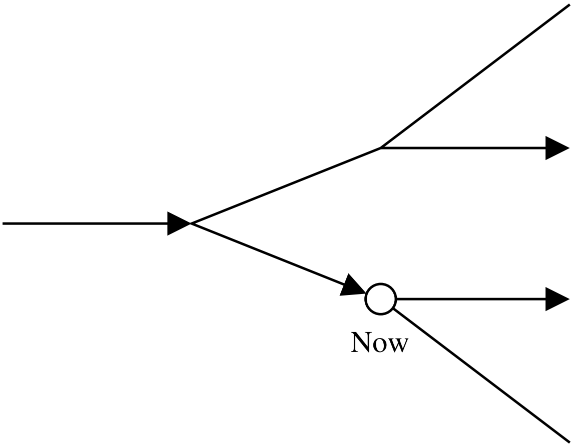


Fig. 2. An Ockhamistic model of branching time. At every branching point there will be one possible future which is the "true future".

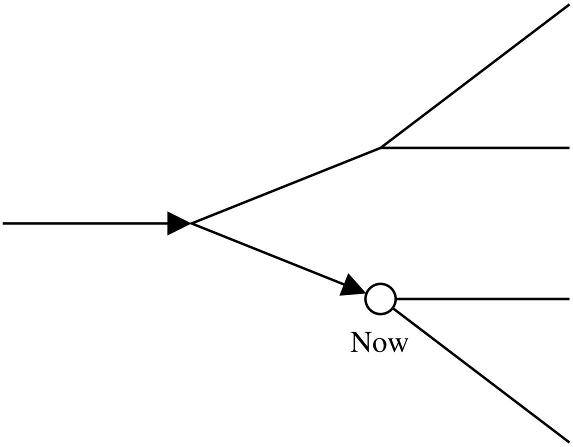


Fig. 3. A Peircean model of branching time. There is no difference between the status of the possible futures at any branching point.

logic suggested by the medieval philosopher William of Ockham (ca. 1285-1349), and because the Peircean logic is based on some ideas of indeterminism which can be found in the writings of C.S. Peirce (1839-1914). Graphically, the two kinds of branching time models can be presented in the following way:

The semantics corresponding to these branching time models can be established in the following way: An operator *Ock* is an Ockhamistic valuation operator in a given Ockhamistic structure, if for any temporal instant *t* in any chronicle *c* (i.e. $t \in c$) and any tense-logical statement *p*, *Ock(t,c,p)* is a meta-statement which can be read '*p* is true at *t* in the chronicle *c'*

(a) $Ock(t,c, p \wedge q)$ iff both $Ock(t,c,p)$ and $Ock(t,c,q)$
(b) $Ock(t,c,\sim p)$ iff not $Ock(t,c,p)$
(c) $Ock(t,c,Fp)$ iff $Ock(t',c,p)$ for some $t' \in c$ with $t < t'$
(d) $Ock(t,c,Pp)$ iff $Ock(t',c,p)$ for some $t' \in c$ with $t' < t$
(e) $Ock(t,c, \Box p)$ iff $Ock(t,c',p)$ for all c' with $t \in c'$

If these conditions hold *(TIME,<,Ock)* is said to be an Ockhamistic structure. – A formula *p* is said to be Ockham-valid if and only if *Ock(t,c,p)* for any *t* and *c* (with $t \in c$) and any Ockhamistic structure.

As we shall see, it may be doubted whether Prior's Ockhamistic system is in fact an adequate representation of the tense logical ideas propagated by William of Ockham. At least Ockham's notion of truth seems to differ from the idea used in the presentation of Prior's Ockhamistic system. On the other hand, it should be noted that Prior's Ockhamistic system appears to comprehend at least all the theorems which should be accepted according to Ockham's original ideas. Let us, for instance, consider one tense logical formula:

$$q \supset HFq$$

where *H* is defined as *~P~*. A straightforward reading of this formula would be: "if something is the case now, then it has always been going to be the case". It is obvious from the above definitions that $Ock(t,c,q \supset HFq)$ for any *t* and any *c* with $t \in c$. Therefore $q \supset HFq$ is a theorem in Prior's Ockhamistic system.

Likewise $Pq \supset \Box Pq$ (where *F* does not occur in *q*) is obviously a theorem, whereas the formula $PFq \supset \Box PFq$ is not a theorem in the system. This difference corresponds exactly to the difference between proper past (i.e. a proposition like *Pq* about the past which *q* does not depend on the future) and pseudo-past (i.e. a propostion like *PFq* which may depend on the future).

The Peircean structure can be defined in a manner very similar to Prior's Ockhamistic definition. Only the semantics of *F* would differ. According to Prior's definition of Peircean truth:

Peirce(t,c,Fp) if and only if for all c' with $t \in c'$
Peirce(t',c',p) holds for some $t' \in c'$ with $t < t'$

This definition appears to be in very good accordance with the ideas of C.S. Peirce, who rejected the very idea that statements regarding the contingent future could be true. In fact, it is evident that the Peircean system can be seen as a fragment of the Ockhamistic system. It is obvious that the Peircean *F* corresponds to the Ockhamistic $\Box F$. This means that according to the Peircean model "tomorrow" is identified with "necessarily tomorrow". As seen from natural language as well as from a common sense point of view this is rather problematic. A number of important statements in everyday discourse simply cannot be precisely represented in terms of the Peircean theory.

On the basis of the Ockhamistic view, on the other hand, it is straight forward to construct a model according to which "tomorrow", "possibly tomorrow", and "necessarily tomorrow" will be three different notions. In such a model we may not only refer what happens in some possible future, $\Diamond Fq$, and to what happens in all possible futures, $\Box Fq$, but we may also refer what is going to happen in the future, Fq, as something conceptually different from the possible as well as the necessary future.

The great achievement of the Ockhamistic system could arguably be said to be its property of making a genuine distinction between the following three types of statement:

(i) Necessarily, there will be a sea battle tomorrow.
(ii) Possibly, there will be a sea battle tomorrow.
(iii) There will be a sea battle tomorrow.

However, in the Peirce-system the type of future statement seen in (iii) will have to be interpreted as equivalent with (i). There is no ‘plain future’ in this system. Of course, that is not a consequence of sloppiness on Peirce’s side, but rather it is a deliberate and philosophically motivated choice. This is also in good accordance with the fact that the formula $q \supset HFq$ is not a theorem in Prior’s Peircean system.

It may be doubted, however, whether Prior’s Ockhamistic system is in fact an adequate representation of the tense logical ideas propagated by William of Ockham. According to Ockham, God knows the contingent future, so it seems that he would accept an idea of absolute truth, also when regarding a statement Fq about the contingent future - and not only what Prior has called ”prima-facie assignments” [1967, p.126] like *Ock(t,c,Fq)*. That is, according to Prior’s model such a proposition can be made true ‘by fiat’ simply by constructing a concrete structure which satisfies it. But Ockham would accept that Fq could be true at t without being relativised to any chronicle. This view could be called the true futurist theory. In order to represent this idea satisfactorily we need a function *TRL*, which gives the true future for any moment of time, *t,* in the branching time system. This means that *TRL* can not only be used to find the true future relative to any moment which has been, is, or will be present. The function should in fact work for any moment in the branching time system. More precisely, *TRL*(t) yields the chronicle corresponding to linear past as well as the true future of t; Belnap and Green call it “the thin red line” [1994]. But how can *TRL*(t) be specified? Belnap and Green have argued that:

$$\text{(TRL1)} \quad t \in TRL(t)$$

should hold in general. Moreover, they have also maintained that:

$$\text{(TRL2)} \quad t_1 < t_2 \supset TRL(t_1) = TRL(t_2)$$

should hold for the *TRL*-function. On the other hand, they have argued that the combination of (TRL1) and (TRL2) is inconsistent with the very idea of branching time. The reason is that if (TRL1) and (TRL2) are both accepted, it follows from $t_1 < t_2$ that $t_2 \in TRL(t_1)$, i.e. that all moments of time after t_1 would have to belong to the thin red line through t_1, which means that there will in fact be no branching at all. However, it is very hard to see why a true futurist would have to accept (TRL2), which seems to be too strong a requirement. Rather than (TRL2), the weaker condition (TRL2′) can be employed:

$$\text{(TRL2}'\text{)} \quad (t_1 < t_2 \wedge t_2 \in TRL(t_1)) \supset TRL(t_1) = TRL(t_2)$$

This seems to be much more natural in relation to the notion of a true futurist branching time logic. Belnap has later accepted that (TRL2′) is a relevant alternative to

(TRL2) (see [Belnap et al. 2001 p.169]). I have argued elsewhere [2009] that the theory based on (TRL1) and (TRL2′) can be defended against any attack formulated against it so far. It is also interesting to note that the true futurist theory may, as argued by Craig [1988, p.175], be understood as a formal version of the theory suggested by the Jesuit Luis Molina (1535-1600).

One interesting application of the Ockhamistic approach relevant within the study of user interaction with computer systems should be mentioned: A model based on Prior's Ockhamistic temporal logic may be interpreted as including a dynamic plan i.e. a plan including alternative plans corresponding to any possible choice made by the person(s) in question. This means that the model should at least include a default choice or a suggested choice (i.e. an advice to the user) whenever there are alternative futures. This plan is supposed to lead the user to the best possible outcome given the choices of the user.

5 Four Grades of Tense-Logical Involvement

Prior suggested a distinction between four possible grades of tense-logical involvement corresponding to four different views of how to relate the A-notions (past, present and future) to the B-notions ('earlier than', 'later than', 'simultaneous with'):

1. The B-notions are more fundamental than the A-notions. Therefore, in principle the A-notions have to be defined in terms of the B-notions.
2. The B-notions are just as fundamental as the A-notions. The A-notions cannot be defined in a satisfactory manner in terms of the B-notions (and vice versa). The two sets of notions have to be treated on a par.
3. The A-notions are more fundamental than the B-notions. There is also a primitive and fundamental notion of (temporal) possibility. In principle the B-notions have to be defined in terms of the A-notions and the primitive notion of temporal possibility.
4. The A-notions are more fundamental than the B-notions. In principle the B-notions have to be defined in terms of the A-notions. Even the notion of temporal possibility can be defined on terms of the A-notions.

Understood in this way, it is obvious that Prior's four grades of tense-logical involvement (see [Øhrstrøm and Hasle 1995 p.176 ff.]) represent four different views of time and also four different foundations of temporal logic.

The first grade defines tenses entirely in terms of objective instants and an earlier-later relation. For instance, a sentence such as *Fp*, 'it will be the case that *p*', is defined as a short-hand for 'there exists some instant *t* which is later than now, and *p* is true at *t*', and similarly for the past tense; these definitions are, of course:

(DF) $T(t,Fp) \equiv_{def} \exists t_1 : t<t_1 \wedge T(t_1,p)$
(DP) $T(t,Pp) \equiv_{def} \exists t_1 : t_1<t \wedge T(t_1,p)$

Tenses, then, can be considered as mere meta-linguistic abbreviations, so this is the lowest grade of tense logical involvement. That is, it is the least A-like and the most B-like theory. Prior succinctly described the first grade as follows:

> …there is a nice economy about it ... it reduces the minimal tense logic to a by-product of the introduction of four definitions into an ordinary first-order theory, and richer [tense logical] systems to byproducts of conditions imposed on a relation in that theory. [Prior 2003, p.119-20]

In the first grade, tense operators are simply a handy way of summarizing the properties of the before-after relations, which constitute the B-theory. Hence, in the first grade, B-theory concepts are seen to be determining for a proper understanding of time and reality. On this view, instants acquire an independent ontological status whereas tenses are deemed to have no independent epistemological status. As we have seen, Prior rejected the idea of temporal instants as something primitive and objective. He claimed:

> Time is not an object, but whatever is real exists and acts in time... But this earlier-later calculus is only a convenient but indirect way of expressing truths that are not really about 'events' but about things ... [Copeland 1996, p.45]

From Prior's point of view, the first grade is a reduction of reality. First of all the notion of the present, the Now, disappears. This means that the first grade represents a serious conceptual loss in expressibility in our theoretical approach to reality. This was clearly not acceptable to Prior. On the other hand, he also maintained that the conceptual construction of instants and dates is a very useful one. After a lecture, which was in fact just one in a series of lectures on temporal logic, probably held somewhere in USA, Prior wrote the following addition to the paper which he was going to read at the next lecture in the series:

> A wants me to relativise my tenses to dates. It seems to me that behind this request there is a metaphysics. Behind this request there is the idea that the whole of time is absolutely there with all these dates, and all events and processes just are, located in various parts of this giant fixed frame. I do not believe this. I think this way is to treat all time as if it were already past. I don't believe this. I don't believe that events and processes are; rather events happen (and then come to have happened) and processes go on (and then come to have gone on), and even this is an abstraction - the basic reality is things acting. But even in this flux there is a pattern, and this pattern I try to trace with my tense-logic; and it is because this pattern exists that men have been able to construct their seemingly timeless frame of dates. Dates, like classes, are a wonderful and tremendously useful invention, but they are an invention; the reality is things acting. [Bodleian Library, MS in box 6, 1 sheet, no title.]

In the second grade of tense logical involvement, tenses are not reduced into B-series notions. Rather, they are treated on a par with the earlier-later relation. This means that this grade represents something between a B-theory and a A-theory, whereas the first grade clearly is a B-theory. Specifically, a bare proposition *p* is treated as a syntactically full-fledged proposition, on a par with propositions such as *T(t,p)* ('it is true at time *t* that *p*'). The point of the second grade is that a bare proposition with no explicit temporal reference is not to be viewed as an incomplete proposition. One consequence of this is that an expression such as *T(t,T(t',p))* is also well-formed, and of the same type as *T(t,p)* and *p*. Prior showed how such a system leads to a number of theses, which relates tense logic to the earlier-later calculus and vice versa [Prior 2003, p.121]. According to Prior's view, however, this grade can like the first grade be criticized for giving the instants an unacceptable ontological status.

The third grade is A-theory. The notion of instant-propositions is introduced, and the formal language is extended with a primitive possibility operator, ◊, and its dual necessity operator, □, defined as ~◊~. In addition, standard quantification theory is assumed. As we have seen, the instant-propositions are maximal propositions with properties as stated in (I1-3) and corresponding to what the B-theory would call instants. Given the formal apparatus of the third grade it becomes possible to define *T(a,p)* as well as the before-after relation, <:

(DT) $T(a, p) \equiv_{def} \Box(a \supset p)$
(DB) $a < b \equiv_{def} \Box(a \supset Fb)$

Here *a* and *b* are arbitrary instant-propositions. Prior was able to prove that given the most basic tense-logic system K_t everything that one would expect to hold for *T* and < in a B-theory would also hold for *T* and < as defined in (DT) and (DB). This certainly makes the third grade a very attractive system as seen from an A-theoretical point of view.

Prior has thus shown how we can in fact interpret B-logic within A-logic, namely in a given modal context in which we can interpret instants as propositions and quantify over them. In this sense B-logical semantics is absorbed within an entirely A-logical axiomatics. In Prior's own words, this means "to treat the first order theory of the earlier-later relation as a mere by-product of tense logic" [Prior 2003, p.273]. He developed this view even further in his fourth grade, in which he suggested a tense logical definition of the necessity-operator such that the only primitive operators in the theory are the two tense logical ones: *P* and *F*. Prior himself favoured this fourth grade. It appears that his reasons for wanting to reduce modality to tenses were mainly metaphysical, since they have to do with his rejection of the concept of the (one) true (but still from a human point of view unknown) future. If one accepts the fourth grade of tense-logical involvement, it will turn out that something like the Peirce solution will be natural, and that we have to reject solutions, which involve the idea of a true or simple future - like the Ockhamistic theory. In all obvious models constructed in accordance with the fourth grade, "tomorrow" is identified either with "possibly tomorrow" or with "necessarily tomorrow". On the basis of Ockhamistic theory, on the other hand, it is straight forward to construct a model according to which "tomorrow", "possibly tomorrow", and "necessarily tomorrow" will be three different notions. In such a model we may not only refer what happens in some possible future, *◊Fq*, and to what happens in all possible futures, □*Fq*, but we may also refer what is going to happen in the future, *Fq*, as something different from the possible as well as the necessary future. It seems that anyone who wants to maintain the Ockhamistic view has to reject the fourth grade. From this point of view the third grade becomes the obvious choice.

6 Conclusion

We have seen that we have no convincing reasons for accepting arguments for the unreality of time based on McTaggart's paradox. The tenses appear to be crucial for a proper understanding of temporal reality. The fundamental asymmetry between past and future can be represented in a clarifying manner in terms of the models of branching time. We have also argued that the three notions of 'future' (necessary, possible, and plain) should be integrated in a branching time model. This can be done using an Ockhamistic model and even better with a true futurist theory. Finally, we have presented Prior's four grades of tense-logical involvement, and we have argued in favour of the third grade, which seems to follow given that the A-concepts are more conceptually basic than the B-concepts and that the Ockhamistic approach is more adequate than the Peircean view.

References

1. Belnap, N., Green, M.: Indeterminism and the Thin Red Line, Philosophical Perspectives. Logic and Language 8, 365–388 (1994)
2. van Benthem, J.: The Logic of Time. Synthese Library, 2nd edn., vol. 156. Kluwer Academic Publishers, London (1991)
3. Braüner, T.: Homophonic Theory of Truth for Tense Logic. Advances in Modal Logic, vol. 3, pp. 59–72. World Scientific, Singapore (2002a)
4. Braüner, T.: Modal Logic, Truth, and the Master Modality. Journal of Philosophical Logic 31, 359–386 (2002b)
5. Broad, C.D.: Examination of McTaggart's Philosophy, Cambridge, vol. ii (1938)
6. Copeland, J. (ed.): Logic and Reality. Essays on the Legacy of Arthur Prior. Clarendon Press (1996)
7. Craig, W.L.: The Problem of Divine Foreknowledge and Future Contingents from Aristotle to Suarez. E.J. Brill, New York (1988)
8. Craig, W.L.: Nice Soft Facts: Fischer on Foreknowledge. Religious Studies 25, 235–246 (1989)
9. Craig, W.L.: The Tensed Theory of Time. A Critical Examination, Synthese Library, vol. 293. Kluwer Academic Publishers, Dordrecht (2000a)
10. Craig, W.L.: The Tenseless Theory of Time. A Critical Examination, Synthese Library, vol. 294. Kluwer Academic Publishers, Dordrecht (2000b)
11. Cresswell, M.: Modality and Mellor's McTaggart. Studia Logica 49, 163–170 (1990)
12. Fraser, J.T., Haber, F.C., Müller, G.H. (eds.): The Study of Time. Springer, Berlin (1972)
13. McTaggart, J.M.E.: The Unreality of Time. Mind, 457–474 (1908)
14. McTaggart, J.M.E.: The Nature of Existence, ii: 'Time', pp. 23–34. Cambridge University Press, Cambridge (1927) Reprinted in Poidevin and MacBeath
15. Mellor, D.: Real Time, Cambridge (1981)
16. le Poidevin, R., MacBeath, M.: The Philosophy of Time. Oxford University Press, Oxford (1993)
17. Prior, A.N.: Time and Modality. Oxford University Press, Oxford (1957)
18. Prior, A.N.: Past, Present and Future. Clarendon Press, Oxford (1967)

19. Prior, A.N.: The Notion of the Present. In: Fraser, et al, pp. 320–323 (1972)
20. Prior, A.N., Fine, K.: Worlds, Times and Selves, Duckworth (1977)
21. Prior, A.N.: Papers on Time and Tense. In: Hasle, P., Øhrstrøm, P., Braüner, T., Copeland, J. (eds.). Oxford University Press, Oxford (2003)
22. Sylvan, R.: Other Withered Stumps of Time. In: Copeland, pp. 111–130 (1996)
23. Øhrstrøm, P., Hasle, P.: Temporal Logic - From Ancient Ideas to Artificial Intelligence. Kluwer Academic Publishers, Dordrecht (1995)
24. Øhrstrøm, P.: In Defence of the Thin Red Line: A Case for Ockhamism, Humana. Mente, vol. 8, pp.17–32 (2009)

Can Ontology Inform Ontologies?

Boris Wyssusek

Brisbane, Australia
wyssusek@gmail.com

Traditionally, since it was coined in the early 17th century by German school-philosophy, the word “ontology” has been used to name a field of metaphysics as well as distinct metaphysical doctrines. Since the 1990s, the word “ontology” appears increasingly in information sciences, and likewise in fields that have been subjected to ‘informatisation’ such as biology, geography, and medicine. In all these fields, however, the word “ontology” is being used with different meanings, and for the most part with meanings that are distant from its philosophical roots.

Given the obvious centrality and significance of the word “ontology” in the contemporary information sciences, both the terminological indeterminateness and the apparent semantic change are issues warranting investigation. A particular motivation for an inquiry derives from the context at hand, since the theoretical foundations of Formal Concept Analysis have already been subjected to ontological considerations, and methods of Formal Concept Analysis have been applied to the development of ontologies in the information sciences.

Starting from etymological considerations, significant developments in ontological thought in the history of occidental philosophy are summarised. Special attention is given to the separation of ontology from classical metaphysics, the emancipation of the sciences from philosophy, and the concurrent reconfiguration of the relation between science and philosophy. These developments in western knowledge constitute the condition of the possibility for the appropriation of modes of ontological thought by the information sciences and beyond.

Subsequently, a range of meanings of the word “ontology” in the information sciences is identified. It is conjectured that the multiplicity of meanings can be explained by associating selected connotations of “ontology” with the corresponding problems information scientists try to solve. This exercise discloses semantic changes of the word “ontology,” which resulted from the transformation of a philosophical term into a scientific-practical term. These semantic changes characterise the relationship between philosophy and information sciences with respect to ontology as a problem — whether (philosophical) ontology can or actually does inform the notions of ontology in the information sciences.

Concluding, some brief reflections on the relation between Formal Concept Analysis and ontology — with reference to Rudolf Wille’s proposal of a Semantology — are offered.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, p. 82, 2009.

Factor Analysis of Incidence Data via Novel Decomposition of Matrices*

Radim Belohlavek[1,2] and Vilem Vychodil[1,2]

[1] State University of New York at Binghamton
PO Box 6000, Binghamton, NY 13902–6000, USA
rbelohla@binghamton.edu, vychodil@binghamton.edu
[2] Palacky University, Olomouc
Tomkova 40, CZ-779 00 Olomouc, Czech Republic

Abstract. Matrix decomposition methods provide representations of an object-variable data matrix by a product of two different matrices, one describing relationship between objects and hidden variables or factors, and the other describing relationship between the factors and the original variables. We present a novel approach to decomposition and factor analysis of matrices with incidence data. The matrix entries are grades to which objects represented by rows satisfy attributes represented by columns, e.g. grades to which an image is red or a person performs well in a test. We assume that the grades belong to a scale bounded by 0 and 1 which is equipped with certain aggregation operators and forms a complete residuated lattice. We present an approximation algorithm for the problem of decomposition of such matrices with grades into products of two matrices with grades with the number of factors as small as possible. Decomposition of binary matrices into Boolean products of binary matrices is a special case of this problem in which 0 and 1 are the only grades. Our algorithm is based on a geometric insight provided by a theorem identifying particular rectangular-shaped submatrices as optimal factors for the decompositions. These factors correspond to formal concepts of the input data and allow for an easy interpretation of the decomposition. We present the problem formulation, basic geometric insight, algorithm, illustrative example, experimental evaluation.

1 Introduction

1.1 Problem Description

Reducing data dimensionality by mapping the data from the space of directly observable variables into a lower dimensional space of new variables is of fundamental importance for understanding and management of data. Traditional approaches achieve dimensionality reduction via matrix decomposition. In factor analysis, a decomposition of an object-variable matrix is sought into an object-factor matrix and a factor-variable matrix with the number of factors reasonably

* Supported by research plan MSM 6198959214.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 83–97, 2009.

small. This way, objects can be represented in a lower dimensional space of factors from which their representation in the space of original variables can be retrieved by a linear combination.

Recently, new methods of matrix decomposition and dimensionality reduction have been developed. One aim is to have methods which are capable of discovering possibly non-linear relationships between the original space and the lower dimensional space [26,32]. Another is driven by the need to take into account constraints imposed by the semantics of the data. An example is Boolean factor analysis in which a decomposition of a binary matrix is sought into two binary matrices [10,21,25].

In this paper, we consider decompositions of matrices I with a particular type of ordinal data. Entries I_{ij} of I are grades to which the object corresponding to i-th row has, or is incident with, the attribute corresponding to the j-th row, e.g. to which a hotel is rated as a good hotel. Typical examples of such data are results of questionnaires where respondents (rows) rate services, products, etc. according to various criteria (columns); results of performance evaluation of people or machines (rows) by various tests (columns); or binary data in which case there are only two grades, 0 (no, failure) and 1 (yes, success). Our goal is to decompose an $n \times m$ object-attribute matrix I into a product

$$I = A \circ B$$

of an $n \times k$ object-factor matrix A and a $k \times m$ factor-attribute matrix B with a reasonably small number k of factors.

The scenario is thus similar to that of ordinary factor analysis but there are important differences. First, we assume that the entries of I, i.e. the grades, as well as the entries of A and B are taken from a bounded scale L of grades. Examples of such scales are the unit interval $L = [0, 1]$, the Likert scale $L = \{1, \ldots, 5\}$ of degrees of satisfaction, or other scales used in mathematical psychology and psychophysics [18]. Second, the matrix composition operation $\circ$ used in our decompositions is not the usual matrix product. Instead, we use a so-called t-norm-based product where a t-norm is a function which we use for aggregation of grades, cf. also [9]. A Boolean matrix product is a particular case of this product in which the scale has 0 and 1 as the only grades. It is to be emphasized that we attempt to treat graded incidence data in a way which is compatible with its semantics. This need has been recognized long ago in mathematical psychology, in particular in measurement theory [18]. For example, even if we represent the grades by numbers such as $0 \sim$ strongly disagree, $\frac{1}{4} \sim$ disagree, ..., $1 \sim$ strongly agree, addition, multiplication by real numbers, and linear combination of graded incidence data may not have natural meaning. Likewise, decomposition of graded incidence matrix I into the ordinary matrix product of arbitrary real-valued matrices A and B suffers from a difficulty to interpret A and B, as well as to interpret the way I is reconstructed from, or explained by, A and B. This is not to say that the usual matrix decompositions of incidence data I are not useful. [22,31] report that decompositions of binary matrices into real-valued matrices may yield better reconstruction accuracies. Hence, as far as the dimensionality reduction aspect (the technical aspect) is concerned, ordinary decompositions may be favorable. However, when the knowledge discovery aspect plays a role, attention needs to be paid to the semantics of decomposition, and to the appropriate mathematical structure and geometry of the space of attributes, the space of factors, and the transformations between them.

Our paper is organized as follows. Section 1.2 provides and overview of related work. In Section 2, we define the decomposition problem and explain the factors we use for the decomposition and their role. Section 3 contains an illustrative example. An algorithm for decompositions of incidence matrices and its experimental evaluation is presented in Section 4.

1.2 Related Work

In case of matrices with real numbers as entries (sometimes referred to as continuous data), various methods for matrix decompositions have been developed. The best known include, in particular, factor analysis (FA), principal component analysis (PCA), and singular value decomposition (SVD) [2,13]. Results regarding optimality of such decompositions are available. However, these methods decompose a real-valued matrix into a product of matrices with possibly negative values which are generally difficult to interpret [22]. Non-negative matrix factorization [19] overcomes this problem at the expense of not minimizing the global reconstruction error. The advantage is that the decomposition describes the original data as additively composed of its easily interpretable parts. Restriction to certain values (non-negative ones) and the resulting gain in interpretability is related to our work.

There are several papers on decomposing binary matrices into non-binary matrices such as [20,27,28,30,36], see also [31] for further references.

Decompositions of binary matrices into binary matrices have been studied in a number of papers. Early work was done by Markowsky et al., see e.g. [24,25,29] which already include complexity results showing the hardness of problems related to such decompositions. Decompositions of binary matrices into a Boolean product of binary matrices using Hopfield-like associative neural networks have been studied, e.g., by Frolov et al., see [10]. This approach is a heuristic in which the factors correspond to attractors of the neural network. Other heuristic approaches to Boolean factor analysis include [15,16]. [6] shows that the decomposition of a binary matrix I to a Boolean product of binary matrices is equivalent to covering the matrix by rectangular submatrices which contain 1s and shows that formal concepts of I [11] are optimal factors for such decomposition. The problem of covering binary matrices with their submatrices containing 1s was studied in [12]. [22] presents an algorithm for finding approximate decompositions of binary matrices into Boolean product of binary matrices which is based on associations between columns of I. [33] looks at the relationship between several problems related to decomposition of binary matrices.

2 Decomposition and Factors

2.1 Decomposition

Consider an $n \times m$ object-attribute matrix I with entries I_{ij} expressing grades to which object i has attribute j. We assume that the grades are taken from a bounded scale L. In general, we assume that L is equipped with a partial order $\leq$, is bounded from below and above by elements denoted 0 and 1, and conforms to the structure of a complete lattice, i.e. infima and suprema of subsets of L exist. We do not assume the scale to be linearly ordered although in practical applications this is usually the case. Grades of ordinal scales are conveniently represented by numbers, such as the Likert scale $\{1, \ldots, 5\}$. In such a case we assume these numbers are normalized and taken from the unit interval $[0, 1]$. As an example, the Likert scale is represented by $L = \{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\}$. Due to the well-known Miller's 7 ± 2 phenomenon [23], one might argue that we should restrict ourselves to small scales but we consider arbitrary ones, including thus the unit interval $L = [0, 1]$ as well.

We want to decompose I into an $n \times k$ object-factor matrix A and a $k \times m$ factor-attribute matrix B which again have entries from the scale L. Entries A_{il} and B_{lj} are interpreted as degrees to which factor l applies to object i and to which attribute j is a manifestation of factor l, respectively. We assume that the object-attribute relationship is explained using the (hidden) factors as follows: object i has attribute j if there is a factor l which applies to i and for which j is one of its manifestations. Now, for a factor l there is a degree A_{il} to which l applies to i and a degree B_{lj} to which j is a manifestation of l. To obtain a degree a to which "l applies to i and j is a manifestation of l", we aggregate A_{il} and B_{lj} using an aggregation function $\otimes : L \times L \to L$ and put $a = A_{il} \otimes B_{lj}$, cf. [9]. This way, we obtain k degrees $A_{il} \otimes B_{lj}$, one for every factor $l = 1, \ldots, k$. Finally, we take the supremum $\bigvee$ of degrees $A_{il} \otimes B_{lj}$ (such supremum coincides with maximum if L is linearly ordered) as a result. That is, our composition operation for $I = A \circ B$ is defined by

$$(A \circ B)_{ij} = \bigvee_{l=1}^{k} A_{il} \otimes B_{lj}. \tag{1}$$

Notice that if $L = \{0, 1\}$ and $\otimes$ is the truth function of conjunction, $A \circ B$ is the Boolean matrix product. We use t-norms for aggregation functions $\otimes$. T-norms originated in K. Menger's work on statistical metric spaces [17] and are used as truth functions of conjunctions in fuzzy logic [14]. Their properties make them good candidates for aggregating graded data [8,9,17]. Note that with $\otimes$ being a t-norm, (1) is used in fuzzy set theory to define compositions of fuzzy relations [35]. Examples of $\otimes$ include the Łukasiewicz t-norm on $L = [0, 1]$ or on an equidistant subchain of $[0, 1]$ defined by $a \otimes b = \max(0, a + b - 1)$, the minimum t-norm on $L = [0, 1]$ or on a subset of $[0, 1]$ defined by $a \otimes b = \min(a, b)$, and the product t-norm $a \otimes b = a \cdot b$ on $L = [0, 1]$. Using a decomposition $I = A \circ B$ with (1), attributes are expressed by means of factors in a non-linear manner:

Example 1. With Łukasiewicz t-norm, let $I = A \circ B$ be

$$\begin{pmatrix} 0.3 & 0.0 & 0.1 \\ 0.3 & 0.7 & 0.5 \\ 0.5 & 0.8 & 0.6 \end{pmatrix} = \begin{pmatrix} 0.2 & 0.8 \\ 0.9 & 0.8 \\ 1.0 & 1.0 \end{pmatrix} \circ \begin{pmatrix} 0.4 & 0.8 & 0.6 \\ 0.5 & 0.2 & 0.3 \end{pmatrix}.$$

Then for $Q_1 = (0.6\ 0.2)$ and $Q_2 = (0.4\ 0.3)$ we have $(Q_1 + Q_2) \circ B = (1.0\ 0.5) \circ B = (0.4\ 0.8\ 0.6) \neq (0.0\ 0.6\ 0.2) = (0.0\ 0.4\ 0.2) + (0.0\ 0.2\ 0.0) = Q_1 \circ B + Q_2 \circ B$. This demonstrates non-linearity of the relationship between factors and attributes.

2.2 Factors for Decomposition

Next, we describe the factors we use for decomposition of I. For this purpose, we make use of a so-called residuum induced by the t-norm $\otimes$ [14,17], i.e. a binary function $\rightarrow$ on L defined by

$$a \rightarrow b = \max\{c \in L \,|\, a \otimes c \leq b\}.$$

Residuum satisfies an important technical condition called adjointness, namely,

$$a \otimes b \leq c \text{ iff } a \leq b \rightarrow c.$$

L together with $\otimes$ and $\rightarrow$ forms a complete residuated lattice [34]. We leave out technical details including the properties of residuated lattices and refer to [14]. The residuum induced by the Łukasiewicz t-norm is defined by $a \rightarrow b = \min(1, 1 - a + b)$.

We are going to use formal concepts associated to I as factors for a decomposition of I. Formal concepts are particular pairs $\langle C, D\rangle$ of graded sets (fuzzy sets) C of objects and D of attributes, see [4]. That is, $C : \{1, \ldots, n\} \rightarrow L$ assigns to every object i a degree $C(i) \in L$ to which C applies to i. Likewise, $D : \{1, \ldots, m\} \rightarrow L$ assigns to every attribute j a degree to which D applies to j. Denote by L^U the set of all graded (fuzzy) sets in a set U, i.e. the set of all mappings from U to L, and put $X = \{1, \ldots, n\}$ (objects) and $Y = \{1, \ldots, m\}$ (attributes).

Definition 1. *[4] A formal concept of I is any pair $\langle C, D\rangle$ for which $C^{\uparrow} = D$ and $D^{\downarrow} = C$ where ${}^{\uparrow} : L^X \rightarrow L^Y$ and ${}^{\downarrow} : L^Y \rightarrow L^X$ are operators defined by*

$$C^{\uparrow}(j) = \textstyle\bigwedge_{i\in X}(C(i) \rightarrow I_{ij}),$$
$$D^{\downarrow}(i) = \textstyle\bigwedge_{j\in Y}(D(j) \rightarrow I_{ij}).$$

In the definition, $\bigwedge$ is the infimum in L (in our case, since X and Y are finite, infimum coincides with minimum if L is linearly ordered). The set $\mathcal{B}(X, Y, I)$ of all formal concepts of I is called the concept lattice of I. Formal concepts are simple models of concepts in the sense of traditional, Port-Royal logic. If I is (a characteristic function of) an ordinary binary relation (i.e. $L = \{0, 1\}$), formal concepts of I coincide with the ordinary formal concepts of Wille [11]. C and D are called the extent and the intent of a formal concept $\langle C, D\rangle$ and represent the objects and the attributes which fall under the concept. The graded setting takes into account that empirical concepts are graded rather than clear-cut. The concept lattice equipped with a subconcept-superconcept ordering $\leq$ defined by

$$\langle C_1, D_1\rangle \leq \langle C_2, D_2\rangle \text{ iff } C_1(i) \leq C_2(i) \text{ for all } i \in X,$$

which is equivalent to $D_2(j) \leq D_1(j)$ for all $j \in Y$, is indeed a complete lattice [4]. Note that since $\rightarrow$ can be interpreted as a truth function of implication, a formal concept $\langle C, D\rangle$ can be seen as a pair of graded sets C and D such that $D(j)$ is the degree to which j is shared by all objects to which C applies, and $C(i)$ is the degree to which i shares all attributes to which D applies [4].

We are going to use formal concepts of I in the following way. For a set

$$\mathcal{F} = \{\langle C_1, D_1\rangle, \ldots, \langle C_k, D_k\rangle\}$$

of formal concepts of I, we denote by $A_{\mathcal{F}}$ an $n \times k$ matrix in which the l-th column consists of grades assigned to objects by C_l. Likewise, we denote by $B_{\mathcal{F}}$ a $k \times m$ matrix in which the l-th row consists of grades assigned to attributes by D_l. That is,

$$(A_{\mathcal{F}})_{il} = (C_l)(i) \quad \text{and} \quad (B_{\mathcal{F}})_{lj} = (D_l)(j).$$

If $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$, $\mathcal{F}$ can be seen as a set of factors which fully explain the data. In such a case, we call the formal concepts from $\mathcal{F}$ *factor concepts*. Given I, our aim is to find a small set $\mathcal{F}$ of factor concepts. Using formal concepts of I as factors is intuitively appealing because, as mentioned above, the formal concepts are in fact, simple models of human concepts according to traditional logic approach. In fact, factors are often called "(hidden) concepts" in the ordinary factor analysis. In addition, the extents and intents of the concepts, i.e. columns and rows of $A_{\mathcal{F}}$ and $B_{\mathcal{F}}$, have a straightforward interpretation: they represent the grades to which the factor concept applies to particular objects and particular attributes.

Before we turn to an illustrative example, we provide a geometric interpretation of $A \circ B$. Let $I = A \circ B$. Denote by J_l the $n \times m$ matrix defined by $(J_l)_{ij} = A_{il} \otimes B_{lj}$, $l = 1 \ldots, k$. That is, $J_l = A_{\_l} \circ B_{l\_}$ is the $\circ$-product of the l-th column of A and the l-th row of B. (1) then yields that $I = J_1 \vee \cdots \vee J_k$, i.e. I is the $\bigvee$-superposition of $J_1, \ldots, J_k$. Matrices J_l are rectangular (rectangles) in that they result as the Cartesian products of graded sets. If $L = \{0, 1\}$, rectangular matrices are just matrices where the entries containing 1s form a submatrix, i.e. tiles in the sense of [12]. We thus have:

Theorem 1. *$I = A \circ B$ for an $n \times k$ matrix A and a $k \times m$ matrix B if and only if I is a $\bigvee$-superposition of rectangular matrices $A_{\_l} \circ B_{l\_}$, $l = 1, \ldots, k$.*

Remark 1. (1) Note that due to Theorem 1, tiling databases [12] means decomposing I into $A \circ B$ where columns of A and rows of B are the characteristic vectors of the sets of objects and attributes covered by the tiles.

(2) This remains true even for arbitrary scales L: Finding a decomposition of I is equivalent to covering I by rectangular submatrices, i.e. "graded tiles", which result by the Cartesian products of graded sets of objects and attributes, and are contained in I.

(3) Let $\mathcal{F}$ be a set of factor concepts, i.e. $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$. Due to Theorem 1, for any subset $\mathcal{F}'$ of $\mathcal{F}$ we have $(A_{\mathcal{F}'} \circ B_{\mathcal{F}'})_{ij} \leq I_{ij}$. That is, for any subset $\mathcal{F}'$ of $\mathcal{F}$, $A_{\mathcal{F}'} \circ B_{\mathcal{F}'}$ approximates I from below. We will see in Sections 3 and 4 that it is usually the case that even for a small subset $\mathcal{F}' \subseteq \mathcal{F}$, matrix $A_{\mathcal{F}'} \circ B_{\mathcal{F}'}$ is a good approximation of I.

2.3 Optimality of Formal Concepts as Factors

In this section we recall two important results from [5]. The first one says that formal concepts of I are universal factors.

Theorem 2. *For every I there is $\mathcal{F} \subseteq \mathcal{B}(X, Y, I)$ such that $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$.*

The second one says that, as far as exact decompositions of I are concerned, formal concepts are optimal factors in that they provide us with decompositions of I with the least number k of factors.

Theorem 3. *If $I = A \circ B$ for $n \times k$ and $k \times m$ binary matrices A and B, there exists a set $\mathcal{F} \subseteq \mathcal{B}(X, Y, I)$ of formal concepts of I with $|\mathcal{F}| \leq k$ such that for the $n \times |\mathcal{F}|$ and $|\mathcal{F}| \times m$ matrices $A_{\mathcal{F}}$ and $B_{\mathcal{F}}$ we have $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$.*

This means that in looking for decompositions of I, one can restrict the search to the set of formal concepts instead of the set of all possible decompositions.

3 Illustrative Example

Tab. 1 (top) contains the results of top five athletes in 2004 Olympic Games decathlon in points which are obtained using the IAAF Scoring Tables for Combined Events. Note that the IAAF Scoring Tables provide us with an ordinal scale and a ranking function assigning the scale values to athletes. We are going to look at whether this data can be explained using formal concepts as factors. We first transform the data to a five-element scale

$$L = \{0.00, 0.25, 0.50, 0.75, 1.00\}$$

by a natural transformation and rounding. As a consequence, the factors then have a simple reading. Namely, the grades to which a factor applies to an athlete

Table 1. 2004 Olympic Games Decathlon

Scores of Top 5 Athletes

	10	*lj*	*sp*	*hj*	40	11	*di*	*pv*	*ja*	15
Sebrle	894	1020	873	915	892	968	844	910	897	680
Clay	989	1050	804	859	852	958	873	880	885	668
Karpov	975	1012	847	887	968	978	905	790	671	692
Macey	885	927	835	944	863	903	836	731	715	775
Warners	947	995	758	776	911	973	741	880	669	693

Incidence Data Table with Graded Attributes

	10	*lj*	*sp*	*hj*	40	11	*di*	*pv*	*ja*	15
Sebrle	0.50	1.00	1.00	1.00	0.75	1.00	0.75	0.75	1.00	0.75
Clay	1.00	1.00	0.75	0.75	0.50	1.00	0.75	0.50	1.00	0.50
Karpov	1.00	1.00	0.75	0.75	1.00	1.00	1.00	0.25	0.25	0.75
Macey	0.50	0.50	0.75	1.00	0.75	0.50	0.75	0.25	0.50	1.00
Warners	0.75	0.75	0.50	0.50	0.75	1.00	0.25	0.50	0.25	0.75

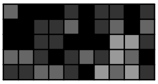
Legend: 10—100 meters sprint race; *lj*—long jump; *sp*—shot put; *hj*—high jump; 40—400 meters sprint race; 11—110 meters hurdles; *di*—discus throw; *pv*—pole vault; *ja*—javelin throw; 15—1500 meters run.

Table 2. Factor Concepts

F_i	*Extent*	*Intent*
F_1	$\{{}^{.5}/\mathrm{Sebrle}, \mathrm{Clay}, \mathrm{Karpov}, {}^{.5}/\mathrm{Macey}, {}^{.75}/\mathrm{Warners}\}$	$\{10, \mathrm{lj}, {}^{.75}/\mathrm{sp}, {}^{.75}/\mathrm{hj}, {}^{.5}/40, 11, {}^{.5}/\mathrm{di}, {}^{.25}/\mathrm{pv}, {}^{.25}/\mathrm{ja}, {}^{.5}/15\}$
F_2	$\{\mathrm{Sebrle}, {}^{.75}/\mathrm{Clay}, {}^{.25}/\mathrm{Karpov}, {}^{.5}/\mathrm{Macey}, {}^{.25}/\mathrm{Warners}\}$	$\{{}^{.5}/10, \mathrm{lj}, \mathrm{sp}, \mathrm{hj}, {}^{.75}/40, 11, {}^{.75}/\mathrm{di}, {}^{.75}/\mathrm{pv}, \mathrm{ja}, {}^{.75}/15\}$
F_3	$\{{}^{.75}/\mathrm{Sebrle}, {}^{.5}/\mathrm{Clay}, {}^{.75}/\mathrm{Karpov}, \mathrm{Macey}, {}^{.5}/\mathrm{Warners}\}$	$\{{}^{.5}/10, {}^{.5}/\mathrm{lj}, {}^{.75}/\mathrm{sp}, \mathrm{hj}, {}^{.75}/40, {}^{.5}/11, {}^{.75}/\mathrm{di}, {}^{.25}/\mathrm{pv}, {}^{.5}/\mathrm{ja}, 15\}$
F_4	$\{\mathrm{Sebrle}, {}^{.75}/\mathrm{Clay}, {}^{.75}/\mathrm{Karpov}, {}^{.5}/\mathrm{Macey}, \mathrm{Warners}\}$	$\{{}^{.5}/10, {}^{.75}/\mathrm{lj}, {}^{.5}/\mathrm{sp}, {}^{.5}/\mathrm{hj}, {}^{.75}/40, 11, {}^{.25}/\mathrm{di}, {}^{.5}/\mathrm{pv}, {}^{.25}/\mathrm{ja}, {}^{.75}/15\}$
F_5	$\{{}^{.75}/\mathrm{Sebrle}, {}^{.75}/\mathrm{Clay}, \mathrm{Karpov}, {}^{.75}/\mathrm{Macey}, {}^{.25}/\mathrm{Warners}\}$	$\{{}^{.75}/10, {}^{.75}/\mathrm{lj}, {}^{.75}/\mathrm{sp}, {}^{.75}/\mathrm{hj}, {}^{.75}/40, {}^{.75}/11, \mathrm{di}, {}^{.25}/\mathrm{pv}, {}^{.25}/\mathrm{ja}, {}^{.75}/15\}$
F_6	$\{{}^{.75}/\mathrm{Sebrle}, {}^{.5}/\mathrm{Clay}, \mathrm{Karpov}, {}^{.75}/\mathrm{Macey}, {}^{.75}/\mathrm{Warners}\}$	$\{{}^{.75}/10, {}^{.75}/\mathrm{lj}, {}^{.75}/\mathrm{sp}, {}^{.75}/\mathrm{hj}, 40, {}^{.75}/11, {}^{.5}/\mathrm{di}, {}^{.25}/\mathrm{pv}, {}^{.25}/\mathrm{ja}, {}^{.75}/15\}$
F_7	$\{\mathrm{Sebrle}, \mathrm{Clay}, {}^{.25}/\mathrm{Karpov}, {}^{.5}/\mathrm{Macey}, {}^{.25}/\mathrm{Warners}\}$	$\{{}^{.5}/10, \mathrm{lj}, {}^{.75}/\mathrm{sp}, {}^{.75}/\mathrm{hj}, {}^{.5}/40, 11, {}^{.75}/\mathrm{di}, {}^{.5}/\mathrm{pv}, \mathrm{ja}, {}^{.5}/15\}\rangle$

can be described in natural language as “not at all”, “little bit”, “half”, “quite”, “fully”, or the like. Tab. 1 (bottom) describes the athletes’ performance using the five-element scale. In addition, we use the Łukasiewicz t-norm on L.

Using shades of gray to represent grades from the five-element scale L, the matrix I corresponding to Tab. 1 (bottom) can be visualized in the following array (rows correspond to athletes, columns correspond to disciplines, the darker the array entry, the higher the score):



The algorithm described in Section 4 found a set $\mathcal{F}$ of 7 formal concepts which factorize I, i.e. for which $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$. These factor concepts are shown in Fig. 1 in the order in which they were produced by the algorithm. For example, factor concept F_1 applies to Sebrle to degree 0.5, to both Clay and Karpov to degree 1, to Macey to degree 0.5, and to Warners to degree 0.75. Furthermore, this factor concept applies to attribute 10 (100 m) to degree 1, to attribute lj (long jump) to degree 1, to attribute sp (shot put) to degree 0.75, etc. This means that an excellent performance (degree 1) in 100 m, an excellent performance in long jump, a very good performance (degree 0.75) in shot put, etc. are particular manifestations of this factor concept. On the other hand, only a relatively weak performance (degree 0.25) in javelin throw and pole vault are manifestations of this factor.

Therefore, a decomposition $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$ exists with 7 factors where:

$$A_{\mathcal{F}} = \begin{pmatrix} 0.50 & 1.00 & 0.75 & 1.00 & 0.75 & 0.75 & 1.00 \\ 1.00 & 0.75 & 0.50 & 0.75 & 0.75 & 0.50 & 1.00 \\ 1.00 & 0.25 & 0.75 & 0.75 & 1.00 & 1.00 & 0.25 \\ 0.50 & 0.50 & 1.00 & 0.50 & 0.75 & 0.75 & 0.50 \\ 0.75 & 0.25 & 0.50 & 1.00 & 0.25 & 0.75 & 0.25 \end{pmatrix},$$

$$B_{\mathcal{F}} = \begin{pmatrix} 1.00 & 1.00 & 0.75 & 0.75 & 0.50 & 1.00 & 0.50 & 0.25 & 0.25 & 0.50 \\ 0.50 & 1.00 & 1.00 & 1.00 & 0.75 & 1.00 & 0.75 & 0.75 & 1.00 & 0.75 \\ 0.50 & 0.50 & 0.75 & 1.00 & 0.75 & 0.50 & 0.75 & 0.25 & 0.50 & 1.00 \\ 0.50 & 0.75 & 0.50 & 0.50 & 0.75 & 1.00 & 0.25 & 0.50 & 0.25 & 0.75 \\ 0.75 & 0.75 & 0.75 & 0.75 & 0.75 & 0.75 & 1.00 & 0.25 & 0.25 & 0.75 \\ 0.75 & 0.75 & 0.75 & 0.75 & 1.00 & 0.75 & 0.50 & 0.25 & 0.25 & 0.75 \\ 0.50 & 1.00 & 0.75 & 0.75 & 0.50 & 1.00 & 0.75 & 0.50 & 1.00 & 0.50 \end{pmatrix}.$$

Again, using shades of gray, this decomposition can be depicted as:

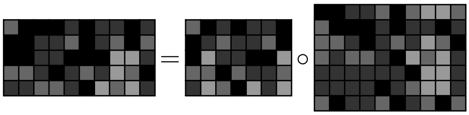


Fig. 1 shows the rectangular patterns corresponding to the factor concepts, cf. Theorem 1.

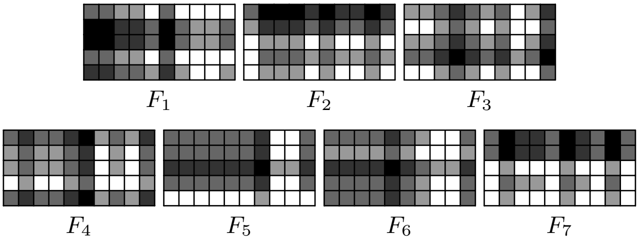


Fig. 1. Factor Concepts as Rectangular Patterns

Fig. 2 demonstrates what portion of the data matrix I is explained using just some of the factor concepts from $\mathcal{F}$. The first matrix labeled by 46% shows $A_{\mathcal{F}_1} \circ B_{\mathcal{F}_1}$ for $\mathcal{F}_1$ consisting of the first factor F_1 only. That is, the matrix is just the rectangular pattern corresponding to F_1, cf. Fig. 1. As we can see, this matrix is contained in I, i.e. approximates I from below, in that $(A_{\mathcal{F}_1} \circ B_{\mathcal{F}_1})_{ij} \leq I_{ij}$ for all entries (row i, column j). Note that Theorem 1 implies that this always needs to be the case, cf. Remark 1 (3). Label 46% indicates that 46% of the entries of $A_{\mathcal{F}_1} \circ B_{\mathcal{F}_1}$ and I are equal. In this sense, the first factor explains 46% of the data. Note however, that several of the $54\% = 100\% - 46\%$ of the other entries of $A_{\mathcal{F}_1} \circ B_{\mathcal{F}_1}$ are close to the corresponding entries of I, so a measure of closeness of $A_{\mathcal{F}_1} \circ B_{\mathcal{F}_1}$ and I which takes into account also close entries, rather than exactly equal ones only, would yield a number larger than 46%.

The second matrix in Fig. 2, with label 72%, shows $A_{\mathcal{F}_2} \circ B_{\mathcal{F}_2}$ for $\mathcal{F}_2$ consisting of F_1 and F_2. That is, the matrix demonstrates what portion of the data matrix I is explained by the first two factors. Again, $A_{\mathcal{F}_2} \circ B_{\mathcal{F}_2}$ approximates I from below and 72% of the entries of $A_{\mathcal{F}_2} \circ B_{\mathcal{F}_2}$ and I coincide now. Note again that even for the remaining 28% of entries, $A_{\mathcal{F}_2} \circ B_{\mathcal{F}_2}$ provides a reasonable approximation of I, as can be seen by comparing the matrices representing $A_{\mathcal{F}_2} \circ B_{\mathcal{F}_2}$ and I, i.e. the one labeled by 72% and the one labelled by 100%.

Similarly, the matrices labeled by 84%, 92%, 96%, 98%, and 100% represent $A_{\mathcal{F}_l} \circ B_{\mathcal{F}_l}$ for $l = 3, 4, 5, 6, 7$, for sets $\mathcal{F}_l$ of factor concepts consisting of $F_1, \ldots, F_l$. We can conclude from the visual inspection of the matrices that already the two or three first factors explain the data reasonably well.

Let us now focus on the interpretation of the factors. Fig. 1 is helpful as it shows the clusters corresponding to the factor concepts which draw together the athletes and their performances in the events.

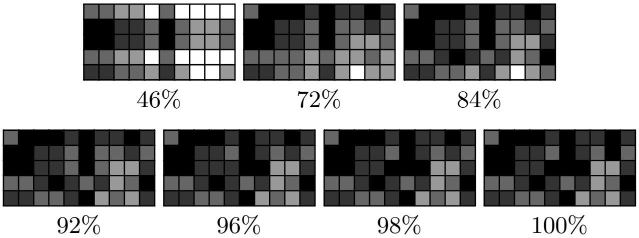


Fig. 2. $\bigvee$-superposition of Factor Concepts

Factor F_1: Manifestations of this factor with grade 1 are 100 m, long jump, 110 m hurdles. This factor can be interpreted as the ability to run fast for short distances. Note that this factor applies particularly to Clay and Karpov which is well known in the world of decathlon. Factor F_2: Manifestations of this factor with grade 1 are long jump, shot put, high jump, 110 m hurdles, javelin. F_2 can be interpreted as the ability to apply very high force in a very short term (explosiveness). F_2 applies particularly to Sebrle, and then to Clay, who are known for this ability. Factor F_3: Manifestations with grade 1 are high jump and 1500 m. This factor is typical for lighter, not very muscular athletes (too much muscles prevent jumping high and running long distances). Macey, who is evidently that type among decathletes (196 cm and 98 kg) is the athlete to whom the factor applies to degree 1. These are the most important factors behind data matrix I.

4 Algorithm and Experiments

In this section, we present a greedy approximation algorithm which takes a data matrix representing I as its input and produces a set $\mathcal{F}$ of formal concepts of I for which $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$. Due to Theorem 1, finding such $\mathcal{F}$ which is minimal in terms of the number of its elements is equivalent to finding a minimal subset of $\{C \otimes D \,|\, \langle C, D\rangle \in \mathcal{B}(X, Y, I)\}$ which covers I. Note that this can be seen as a graded version of a set covering problem which apparently has not been studied before. A further study of this problem including various versions of approximate coverings may yield useful results for processing of graded data. Now, a particular case of this problem for $L = \{0, 1\}$ is just the problem of covering a binary matrix with the smallest possible set of rectangles. This problem is known to be NP-hard, see [24,25,29] for early references, and also [6,12,33]. This indicates that we need an approximation algorithm for the problem of finding small $\mathcal{F}$ for which $I = A_{\mathcal{F}} \circ B_{\mathcal{F}}$.

An obvious approach to the design of such algorithm is to take an approximation algorithm for the (binary) set covering problem, such as the one described in [7], and modify it for the graded case. Such an algorithm would require us to compute first the set $\mathcal{B}(X, Y, I)$ of all formal concepts of I and then select candidates for factors from $\mathcal{B}(X, Y, I)$ using a greedy approach [7]. This would be time-demanding because $\mathcal{B}(X, Y, I)$ can be quite large.

Instead, we propose a different greedy algorithm. The algorithm generates maximal rectangles by looking for "promising columns". A technical property which we utilize is the fact that for each formal concept $\langle C, D\rangle$,

$$D = \bigcup_{j \in Y} \{{}^{D(j)}\!/j\}^{\downarrow\uparrow},$$

i.e. each intent D is a union of intents $\{{}^{D(j)}\!/j\}^{\downarrow\uparrow}$. Moreover, $C = D^{\downarrow}$. Here, $\{{}^{D(j)}\!/j\}$ denotes a graded singleton, i.e. the grade of j in $\{{}^{D(j)}\!/j\}$ is $D(j)$. As a consequence, we may construct any formal concept by adding sequentially $\{{}^{a}\!/j\}^{\downarrow\uparrow}$ to the empty set of attributes. Our algorithm follows a greedy approach that makes us select $j \in Y$ and degree $a \in L$ which maximize the size of

$$D \oplus_a j = \{\langle k, l\rangle \in \mathcal{U} \mid D^{+\downarrow}(k) \otimes D^{+\downarrow\uparrow}(l) \geq I_{kl}\}, \tag{2}$$

where $D^{+} = D \cup \{{}^{a}\!/j\}$ and $\mathcal{U}$ denotes the set of $\langle i, j\rangle$ of I (row i, column j) for which the corresponding entry I_{ij} is not covered yet. Note that the size of $D \oplus_a j$ is just the number of entries of I which are covered by formal concept $\langle D^{\downarrow}, D\rangle$. Therefore, instead of going through all possible formal concepts and selecting a factor from them, we just go through columns and degrees which maximize the value of the factor, i.e. the area covered by the factor, which is being constructed. The algorithm is summarized below.

```
Find-Factors($I$)
 1  $\mathcal{U} \leftarrow \{\langle i,j\rangle \mid I_{ij} \neq 0\}$
 2  $\mathcal{F} \leftarrow \emptyset$
 3  while $\mathcal{U} \neq \emptyset$
 4     do $D \leftarrow \emptyset$
 5        $V \leftarrow 0$
 6        select $\langle j,a\rangle$ that maximizes $|D \oplus_a j|$
 7        while $|D \oplus_a j| > V$
 8           do $V \leftarrow |D \oplus_a j|$
 9              $D \leftarrow (D \cup \{{}^{a}\!/j\})^{\downarrow\uparrow}$
10              select $\langle j,a\rangle$ that maximizes $|D \oplus_a j|$
11        $C \leftarrow D^{\downarrow}$
12        $\mathcal{F} \leftarrow \mathcal{F} \cup \{\langle C,D\rangle\}$
13        for $\langle i,j\rangle \in \mathcal{U}$
14           do if $I_{ij} \leq C(i) \otimes D(j)$
15                 then
16                       $\mathcal{U} \leftarrow \mathcal{U} \setminus \{\langle i,j\rangle\}$
17  return $\mathcal{F}$
```

The main loop of the algorithm (lines 3–16) is executed until all the nonzero entries of I are covered by at least one factor in $\mathcal{F}$. The code between lines 4 and 10 constructs an intent by adding the most promising columns. After such an intent D is found, we construct the corresponding factor concept and add it to $\mathcal{F}$. The loop between lines 13 and 16 ensures that all matrix entries covered by the last factor are removed from $\mathcal{U}$. Obviously, the algorithm is sound and finishes after finitely many steps with a set $\mathcal{F}$ of factor concepts.

Table 3. Exact Factorizability

k	Łukasiewicz ⊗ no. of factors	minimum ⊗ no. of factors
5	5.205 ± 0.460	6.202 ± 1.037
7	7.717 ± 0.878	10.050 ± 1.444
9	10.644 ± 1.316	13.379 ± 1.676
11	13.640 ± 1.615	15.698 ± 1.753
13	16.423 ± 1.879	17.477 ± 1.787
15	18.601 ± 2.016	18.721 ± 1.863

Experimental Evaluation. We now present experiments with exact and approximate factorization of randomly generated matrices and their evaluation. First, we observed how close is the number of factors found by the algorithm Find-Factors to a known number of factors in artificially created matrices. In this experiment, we were generating 20 × 20 matrices according to various distributions of 5 grades. These matrices were generated by multiplying $m \times k$ and $k \times n$ matrices. Therefore, the resulting matrices were factorizable with at most k factors. Then, we executed the algorithm to find $\mathcal{F}$ and observed how close is the number $|\mathcal{F}|$ of factors to k. The results are depicted in Tab. 3. We have observed that in the average case, the choice of a t-norm is not essential and all t-norms give approximately the same results. In particular, Tab. 3 describes results for Łukasiewicz and minimum t-norms which can be seen as two limit cases of t-norms [17]. Rows of Tab. 3 correspond to numbers $k = 5, 6, \ldots, 15$ denoting the known number of factors. For each k, we computed the average number of factors produced by our algorithm in 2000 k-factorizable matrices. The average values are written in the form of "average number of factors ± standard deviation".

As mentioned above, factorization and factor analysis of binary data is a special of our setting with $L = \{0, 1\}$, i.e. with the scale containing just two grades. Then, the matrix product $\circ$ given by (1) coincides with the Boolean matrix multiplication and the problem of decomposition of graded matrices coincides with the problem of decomposition of binary matrices into the Boolean product of binary matrices. We performed experiments with our algorithm in this particular case with three large binary data sets (binary matrices) from the Frequent Itemset Mining Dataset Repository, see `http://fimi.cs.helsinki.fi/data/`. In particular, we considered the CHESS, CONNECT, and MUSHROOM data sets. The results are shown in Tab. 4. The columns labeled by n and m show the numbers of rows and columns of the matrices (e.g., MUSHROOM is a 8124×119 binary matrix). The column labeled by 50% says the following: The first number is the number of factors sufficient to explain 50% of the data entries. For example, the first 5 factors explain 50% of data for CHESS data, i.e. $A_{\mathcal{F}} \circ B_{\mathcal{F}}$ covers 50% of entries of matrix I with $|\mathcal{F}| = 4$. The second number is the ratio

$$\text{number of attributes/number of factors}$$

Table 4. Factorization of Boolean Matrices

Input	Dimensions		Portion of data explained					
Database	n	m	50%		70%		90%	
CHESS	3196	75	5	15.00	13	5.77	33	2.27
CONNECT	67557	129	4	32.25	10	12.90	39	3.31
MUSHROOM	8124	119	7	17.00	19	6.26	46	2.59

which can be regarded as the coefficient of reduction of dimensionality. For example, for the MUSHROOM data set, the first 7 factors produced by our algorithm explain 50% of data and the corresponding coefficient of reduction is $119/7 = 17.00$. The columns labeled by 70% and 90% have analogous meaning.

5 Conclusions and Future Work

We presented a novel approach to factor analysis of matrices with ordinal data. The factors in this approach correspond to formal concepts in the data matrix and the relationship between the factors and original attributes is a non-linear one. One feature of the model is a transparent way of treating the grades which results in good interpretability of factors. Another feature is its feasibility regarding theoretical analysis. As an example, the factors we use are optimal in terms of their number. Furthermore, we proposed a greedy approximation algorithm for the problem of finding a small set of factors and provided results of experiments demonstrating its behavior. Future research will include the following topics:

- Comparison, both theoretical and experimental, to other methods of matrix decompositions.
- Approaches to the problem of approximate factorization of I, continuing our experiments with approximate factorization presented in this paper.
- Development of further theoretical insight focusing particularly on reducing further the space of factors to which the search for factors can be restricted. Note that decompositions of a matrix with grades into a binary matrix and a matrix with grades was studied in [3].
- Study the computational complexity aspects of the problem of approximate factorization, in particular the approximability of the problem of finding decompositions of matrix I [1].
- Explore the applications of the decompositions studied in this paper. One application area is factor analysis. The usefulness of the decompositions in this area was illustrated by the example in Section 3. Another topic which needs to be explored is the possible utilization of the dimensionality reduction provided by the decompositions.

References

1. Ausiello, G., et al.: Complexity and Approximation. Combinatorial Optimization Problems and Their Approximability Properties. Springer, Heidelberg (2003)
2. Bartholomew, D.J., Knott, M.: Latent Variable Models and Factor Analysis, 2nd edn., London, Arnold (1999)
3. Bartl, E., Belohlavek, R., Konecny, J.: Optimal decompositions of matrices with grades into binary and graded matrices. In: Proc. CLA 2008, The Sixth Intl. Conference on Concept Lattice and Their Applications, Olomouc, Czech Republic, pp. 59–70 (2008) ISBN 978–80–244–2111–7
4. Belohlavek, R.: Concept lattices and order in fuzzy logic. Annals of Pure and Applied Logic 128(1–3), 277–298 (2004)
5. Belohlavek, R.: Optimal decompositions of matrices with grades. IEEE IS 2008, Proc. Intl. IEEE Conference on Intelligent Systems, Varna, Bulgaria, pp. 15-2–15-7 (2008) IEEE Catalog Number CFP08802-PRT, ISBN 978-1-4244-1740-7
6. Belohlavek, R., Vychodil, V.: Discovery of optimal factors in binary data via a novel method of matrix decomposition. J. Computer and System Sciences (to appear)
7. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. MIT Press, Cambridge (2001)
8. Fagin, R.: Combining fuzzy information from multiple systems. J. Computer and System Sciences 58, 83–99 (1999); Preliminary version in PODS 1996, Montreal, pp. 216–226 (1996)
9. Fagin, R., Lotem, A., Naor, M.: Combining fuzzy information: an overview. SIGMOD Record 31(2), 109–118 (2002)
10. Frolov, A.A., Húsek, D., Muraviev, I.P., Polyakov, P.A.: Boolean factor analysis by Hopfield-like autoassociative memory. IEEE Transactions on Neural Networks 18(3), 698–707 (2007)
11. Ganter, B., Wille, R.: Formal Concept Analysis. Mathematical Foundations. Springer, Berlin (1999)
12. Geerts, F., Goethals, B., Mielikäinen, T.: Tiling Databases. In: Suzuki, E., Arikawa, S. (eds.) DS 2004. LNCS, vol. 3245, pp. 278–289. Springer, Heidelberg (2004)
13. Golub, G., Van Loan, C.: Matrix Computations. Johns Hopkins University Press (1996)
14. Hájek, P.: Metamathematics of Fuzzy Logic. Kluwer, Dordrecht (1998)
15. Keprt, A., Snášel, V.: Binary factor analysis with help of formal concepts. In: Proc. CLA, pp. 90–101 (2004)
16. Keprt, A., Snášel, V.: Binary Factor Analysis with Genetic Algorithms. In: Proc. IEEE WSTST, pp. 1259–1268. Springer, Heidelberg (2005)
17. Klement, E.P., Mesiar, R., Pap, E.: Triangular Norms. Kluwer, Dordrecht (2000)
18. Krantz, H.H., Luce, R.D., Suppes, P., Tversky, A.: Foundations of Measurement. vol. I (Additive and Polynomial Representations), vol. II (Geometric, Threshold, and Probabilistic Represenations), vol. III (Represenations, Axiomatization, and Invariance). Dover Edition (2007)
19. Lee, D., Seung, H.: Learning the parts of objects by non-negative matrix factorization. Nature 401, 788–791 (1999)
20. Leeuw, J.D.: Principal component analysis of binary data. Application to roll-call analysis (2003), `http://gifi.stat.ucla.edu`
21. Mickey, M.R., Mundle, P., Engelman, L.: Boolean factor analysis. In: Dixon, W.J. (ed.) BMDP statistical software manual, vol. 2, pp. 849–860. University of California Press, Berkeley (1990)

22. Miettinen, P., Mielikäinen, T., Gionis, A., Das, G., Mannila, H.: The discrete basis problem. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 335–346. Springer, Heidelberg (2006)
23. Miller, G.A.: The magical number seven, plus or minus two: Some limits on our capacity for processing information. Psychol. Rev. 63, 81–97 (1956)
24. Nau, D.S.: Specificity covering: immunological and other applications, computational complexity and other mathematical properties, and a computer program. A. M. Thesis, Technical Report CS–1976–7, Computer Sci. Dept., Duke Univ., Durham, N. C (1976)
25. Nau, D.S., Markowsky, G., Woodbury, M.A., Amos, D.B.: A Mathematical Analysis of Human Leukocyte Antigen Serology. Math. Biosciences 40, 243–270 (1978)
26. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science 290, 2323–2326 (2000)
27. Sajama, O.A.: Semi-parametric Exponential Family PCA. In: NIPS 2004 (2004)
28. Schein, A., Saul, L., Ungar, L.: A generalized linear model for principal component analysis of binary data. In: Proc. Int. Workshop on Artificial Intelligence and Statistics, pp. 14–21 (2003)
29. Stockmeyer, L.J.: The set basis problem is NP-complete. IBM Research Report RC5431, Yorktown Heights, NY (1975)
30. Tang, F., Tao, H.: Binary principal component analysis. In: Proc. British Machine Vision Conference 2006, pp. 377–386 (2006)
31. Tatti, N., Mielikäinen, T., Gionis, A., Mannila, H.: What is the dimension of your binary data? In: The 2006 IEEE Conference on Data Mining (ICDM 2006), pp. 603–612. IEEE Computer Society, Los Alamitos (2006)
32. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science 290, 2319–2323 (2000)
33. Vaidya, J., Atluri, V., Guo, Q.: The Role Mining Problem: Finding a Minimal Descriptive Set of Roles. In: ACM Symposium on Access Control Models and Technologies, pp. 175–184 (June 2007)
34. Ward, M., Dilworth, R.P.: Residuated lattices. Trans. Amer. Math. Soc. 45, 335–354 (1939)
35. Zadeh, L.A.: Fuzzy sets. Inf. Control 8, 338–353 (1965)
36. Zivkovic, Z., Verbeek, J.: Transformation invariant component analysis for binary images. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), vol. 1, pp. 254–259 (2006)

A Unified Hierarchy for Functional Dependencies, Conditional Functional Dependencies and Association Rules

Raoul Medina and Lhouari Nourine

Université Blaise Pascal – LIMOS
Campus des Cézeaux,
63173 Aubière Cedex, France
{medina,nourine}@isima.fr

Abstract. Conditional Functional Dependencies (CFDs) are Functional Dependencies (FDs) that hold on a fragment relation of the original relation. In this paper, we show the hierarchy between FDs, CFDs and Association Rules (ARs): FDs are the union of CFDs while CFDs are the union of ARs. We also show the link between Approximate Functional Dependencies (AFDs) and approximate ARs. In this paper, we show that all those dependencies are indeed structurally the same and can be unified into a single hierarchy of dependencies. A benefit of this hierarchy is that existing algorithms which discover ARs could be adapted to discover any kind of dependencies and, moreover, generate a reduced set of dependencies. We also establish the link between the problem of finding equivalent pattern tableaux of a CFD and the problem of finding keys of a relation.

1 Introduction

Dependency theory is one of the major subjects of database theory and has been traditionally used to optimize queries, prevent invalid updates and to normalize databases. Originally, dependency theory has been developed for uninterpreted data and served mainly database conception purposes [10]. The Functional Dependencies introduced by Codd were generalized to Equality Generating Dependencies (EGD) [3]. In [2], dependencies over interpreted data (i.e. equality requirements are replaced by constraints of an interpreted domain) were introduced and generalized the EGDs into Constraint-Generating-Dependencies (CGD). In [6], the authors present a particular case of CGDs: Conditional Functional Dependencies (CFDs), for data cleaning purposes. CFDs are dependencies which holds on instances of the relations. Constraint used in CFDs is the equality and allows to fix particular constant values for attributes. Basically, CFDs can be viewed as FDs which holds on a fragment relation of the original instance relation, this fragment relation being characterized by the constraints to be applied on the attributes. Those constraints represent a selection operation on the relation. All these works focused mainly on implication analysis and axiomatizability.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 98–113, 2009.

Discovery of dependencies existing in an instance of a relation received considerable interest as it allowed automatic database analysis. Knowledge discovery and data mining [1], database management ([5],[9]), reverse engineering [18] and query optimization [19] are among the main applications benefiting from efficient dependencies discovery algorithms. New dependencies on instances of relations were defined. We cite (among others): Association Rules (AR) [1] which are dependencies holding on particular values of attributes, and Approximate Functional Dependencies (AFD) [15] which are FDs which almost hold on a given relation. Note that AFDs have also applications in database design [11]. For those latter dependencies, several measures of approximateness were defined, expressing the interest of the dependency. Numerous algorithms have been proposed for such dependencies discovery, usually ad-hoc algorithms discovering a particular type of dependency. Among the most famous algorithms we can cite: A priori [1] (a level-wise frequent itemsets discovery approach) for ARs mining, Tane [15] (similar approach as A priori but for FDs and AFDs discovery), Close [17] (discovery of closed frequent itemsets). Usually, algorithms used to discover FDs can be adapted to discover ARs and reciprocally.

Recently, some algorithms have been proposed to discover CFDs: [8] and [13]. With no surprise (as our paper will show), those algorithms are adaptations of well-known algorithms for ARs or FDs discovery.

Contributions: As stated above, many dependencies have been studied and are actually used in different applications. In this paper, we show that all those dependencies are indeed structurally the same and can be unified into a single hierarchy of dependencies. A benefit of this hierarchy is that existing algorithms which discover ARs could be adapted to discover any kind of dependencies and, moreover, generate a reduced set of dependencies.

2 Background, Definitions and Preliminary Results

2.1 Definitions

Let R be a relation schema defined over a set of attributes $Attr(R)$. The domain of each attribute $A \in Attr(R)$ is denoted by $Dom(A)$. For an instance r of R, a tuple $t \in r$ and X a sequence of attributes, we use $t[X]$ to denote the projection of t onto X.

An *FD* $X \rightarrow Y$, where $X, Y \subseteq Attr(r)$, is satisfied by r, denoted by $r \models X \rightarrow Y$, if for all pairs of tuples $t_1, t_2 \in r$ we have: if $t_1[X] = t_2[X]$ then $t_1[Y] = t_2[Y]$. In other words, all pairs of tuples *agreeing* on attributes X will agree on attributes Y.

A *CFD* φ defined on R is a pair $(X \rightarrow Y, T_p)$, where (1) $X \rightarrow Y$ is a standard FD, referred to as the FD embedded in φ; and (2) T_p is a tableau with attributes in R, referred to as the pattern tableau of φ, where for each $A \in Attr(R)$ and each pattern tuple $t_p \in T_p$, $t_p[A]$ is either:

- a constant 'a' in $Dom(A)$,
- an unnamed variable $\top$ that draws values from $Dom(A)$,

- or an empty variable $\bot$ which indicates that the attribute does not contribute to the pattern (i.e. $A \notin X \cup Y$).[1]

For any constant a of an attribute we have: $\bot \ \leq a \leq \top$. We define the *pattern intersection operator* $\sqcap$ of two tuples as:

$$t_1 \sqcap t_2 = t_p \text{ such that } \forall A \in \mathit{Attr}(r), \begin{cases} t_p[A] = t_1[A], & \text{if } t_1[A] \leq t_2[A] \\ t_p[A] = t_2[A], & \text{if } t_1[A] > t_2[A] \\ t_p[A] = \bot\ , & \text{otherwise.} \end{cases}$$

We define the *pattern restriction to attributes* X of a tuple t, denoted by $t(X)$ as:

$$t(X) = t_p \text{ such that } \forall A \in \mathit{Attr}(r), \begin{cases} t_p[A] = t[A], & \text{if } A \in X \\ t_p[A] = \bot\ , & \text{otherwise.} \end{cases}$$

We define a subsumption operator $\sqsubseteq$ over pattern tuples t_1 and t_2:

$$t_1 \sqsubseteq t_2 \text{ if and only if } \forall A \in \mathit{Attr}(R), t_1[A] \leq t_2[A].$$

In other words, $t_1 \sqsubseteq t_2$ if and only if $t_1 \sqcap t_2 = t_1$. We define the special tuple *Top* as the tuple with value $\top$ on all attributes, i.e. $t_p \sqsubseteq Top$ for any pattern tuple t_p.

An instance r of R satisfies the CFD φ, denoted by $r \vDash \varphi$, if for each tuple t_p in the pattern tableau T_p of φ, and for each pair of tuples t_1, t_2 in r, if $t_1(X) = t_2(X) \sqsubseteq t_p(X)$, then $t_1(Y) = t_2(Y) \sqsubseteq t_p(Y)$. In other words, a CFD is an FD satisfied by a fragment relation.

A pattern tuple t_p defines a fragment relation of r:

$$r_{t_p} = \{t \in r \mid t_p \sqsubseteq t\}.$$

We will denote by r_{T_p} the fragment relation containing all tuples of r satisfying at least one of the patterns in T_p. Note that given a CFD $(X \to Y, T_p)$, we thus have $r_{T_p} \vDash X \to Y$ and $r - r_{T_p} \nvDash X \to Y$. For this reason, we will denote: $r_{X \to Y} = r_{T_p}$ and $r_{X \not\to Y} = r - r_{T_p}$. To ease the reading, given a CFD $\varphi = (X \to A, T_P)$, we will sometimes denote by r_φ the relation defined by r_{T_p}, i.e. $r_\varphi = r_{T_p} = r_{X \to A}$.

A pattern tableau can thus be seen as a selection query on a relation returning a fragment of the relation. Two patterns tableaux will be said *equivalent* if and only if they return the same fragment relation.

Lemma 1. *For any pattern tableau T_P there exists an equivalent pattern tableau $T_{P_{Const}}$ such that pattern tuples in $T_{P_{Const}}$ contains either constant or empty variables.*

Proof. It is straightforward since in the worst case patterns in $T_{P_{Const}}$ are exactly the tuples of the fragment relation. □

[1] In [12], empty variables are not present in the pattern tableau.

r	A	B	C	D	E	F
t_1	a_1	b_1	c_1	d_1	e_1	f_1
t_2	a_1	b_1	c_1	d_1	e_2	f_1
t_3	a_2	b_1	c_2	d_2	e_2	f_1
t_4	a_2	b_1	c_2	d_2	e_3	f_1
t_5	a_2	b_2	c_2	d_2	e_1	f_2
t_6	a_2	b_2	c_2	d_1	e_1	f_2
t_7	a_2	b_2	c_1	d_1	e_1	f_2
t_8	a_2	b_2	c_1	d_2	e_1	f_2
t_9	a_1	b_2	c_2	d_1	e_2	f_2
t_{10}	a_1	b_2	c_2	d_1	e_1	f_2

Fig. 1. An instance relation r of the schema R

Consequence of previous lemma is that, without loss of generality, we will first focus on CFDs which tableaux contain only constant or empty attributes. In Section 5 we will show how to find equivalent tableaux containing the unnamed variable $\top$.

In the same way, to simplify the discussion without losing generality, we consider, as in [12], CFDs of the form $(X \to A, T_p)$ where A is a single attribute.

Example 1 (CFDs definition)

The CFD $\varphi_1 = (AB \to C, \{(\top\ , b2, \top\ , \bot\ , \bot\ , \bot\ )\})$ is not satisfied by the relation in figure 1 because of tuples t_6 and t_7.

However, the CFDs:

- $\varphi_2 = (AB \to C, \{(a_1, b_1, c_1, \bot\ , \bot\ , \bot\ ), (a_2, b_1, c_2, \bot\ , \bot\ , \bot\ ),$ $(a_1, b_2, c_2, \bot\ , \bot\ , \bot\ )\})$
- $\varphi_3 = (AB \to C, \{(a_1, \top\ , \top\ , \bot\ , \bot\ , \bot\ ), (\top\ , b_1, \top\ , \bot\ , \bot\ , \bot\ )\})$

are satisfied by the relation. Note that φ_2 and φ_3 are equivalent in the relation. Indeed, the FD $AB \to C$ is satisfied in the fragment relation $r_{\varphi_2} = r_{\varphi_3} = \{t_1, t_2, t_3, t_4, t_9, t_{10}\}$ while it does not hold on the fragment relation $r - r_{\varphi_2}$.

There are no CFDs of the form $(AD \to C, T_p)$ in the relation.

The FDs of the relation are $\{B \to F, F \to B, ACDE \to BF, ACE \to BF\}$.

2.2 X-Complete Relations

Using the intersection operator over tuples we could build the tuples lattice of a relation. A closed tuple will thus subsume all tuples agreeing on the same values, i.e. the values of non empty variables in the closed tuple. This notion of set of tuples agreeing on the same values for a given set of attributes X has already been defined in database theory for horizontal decomposition purposes [11] or for FDs discovery [15]. We thus use their definition to define the different closure operators we use in this paper.

Definition 1 (X-complete property [11]). *The relation r is said to be* X-complete *if and only if* $\forall\, t_1, t_2 \in r$ *we have* $t_1[X] = t_2[X]$.

In other words, a relation is X-complete if all tuples agree on the attributes X. Note that they might also agree on other attributes: this constitutes the pattern of r.

Definition 2 (X-complete-pattern). *We call X-complete-pattern of an X-complete relation r, denoted by $\gamma(X,r)$, the pattern tuple on which tuples of r agree. More formally: $\gamma(X,r) = \sqcap \{t \in r\}$.*

Note that since r is X-complete, its X-complete-pattern defines at least the attributes in X (i.e. those attributes do not have the $\perp$ value). Given a relation r and a set of attributes it is always possible to horizontally decompose r in fragment relations which are X-complete.

Definition 3 (X-complete horizontal decomposition). *We denote by $R_X(r)$ the set of all X-complete fragment relations of r. More formally: $R_X(r) = \{r' \subseteq r \mid r'$ is X-complete$\}$.*

In each fragment relations, tuples agree at least on the attributes X.

Definition 4 (Set of X-patterns). *We denote by $\Gamma(X,r)$ the set of all X-complete-patterns of an X-complete decomposition. More formally: $\Gamma(X,r) = \{\gamma(X,r') \mid r' \in R_X(r)\}$.*

Attributes X are defined in all X-complete-patterns. Some other attributes might also be defined.

Definition 5 (Closure operator). *We call* closure *of X in r, denoted by $\theta(X,r)$, the set of all attributes defined in all X-complete-patterns of the relation. More formally:*

$$\theta(X,r) = \{A \in Attr(r) \mid \forall t_p \in \Gamma(X,r), t_p[A] \neq \perp \}.$$

Using the closure operator $\theta(X,r)$, we can trivially characterize FDs[2].

Property 1. Let $A \notin X$. We have $r \models X \rightarrow A$ (i.e. $X \rightarrow A$ is an FD of r) if and only if $A \in \theta(X,r)$.

Proof. $\Rightarrow$: Let $X \rightarrow A$ be an FD of r. Consider the X-complete horizontal decomposition of r. For any $r' \in R_X(r)$ and $\forall t, t' \in r'$, we have $t[X] = t'[X]$. Since $X \rightarrow A$ is an FD, $t[A] = t'[A]$. Thus, $\gamma(X,r')[A] \neq \perp$. Hence, $A \in \theta(X,r)$.

$\Leftarrow$: Consider $A \in \theta(X,A)$. Thus, $\forall t, t' \in r$ such that $t[X] = t'[X]$, we have $t[A] = t'[A]$ (since $\gamma(X,r_t)[A] \neq \perp$). As a consequence, $r \models X \rightarrow A$. □

Note that the closure operator $\theta(X,r)$ is equivalent to the closure of a set of attributes X using FD of r.

Proposition 1. *Let $r' \subseteq r$. Then r' is X-complete if and only if r' is $\theta(X,r)$-complete.*

[2] Note that this closure operator is the same that the one obtained using the agree sets [4].

$R_{a_1b_1}(r)$	A	B	C	D	E	F
t_1	a_1	b_1	c_1	d_1	e_1	f_1
t_2	a_1	b_1	c_1	d_1	e_2	f_1

$\gamma(AB, R_{a_1b_1}(r)) = (a_1, b_1, c_1, d_1, \perp, f_1)$

$R_{a_2b_1}(r)$	A	B	C	D	E	F
t_3	a_2	b_1	c_2	d_2	e_2	f_1
t_4	a_2	b_1	c_2	d_2	e_3	f_1

$\gamma(AB, R_{a_2b_1}(r)) = (a_2, b_1, c_2, d_2, \perp, f_1)$

$R_{a_2b_2}(r)$	A	B	C	D	E	F
t_5	a_2	b_2	c_2	d_2	e_1	f_2
t_6	a_2	b_2	c_2	d_1	e_1	f_2
t_7	a_2	b_2	c_1	d_1	e_1	f_2
t_8	a_2	b_2	c_1	d_2	e_1	f_2

$\gamma(AB, R_{a_2b_2}(r)) = (a_2, b_2, \perp, \perp, e_1, f_2)$

$R_{a_1b_2}(r)$	A	B	C	D	E	F
t_9	a_1	b_2	c_2	d_1	e_2	f_2
t_{10}	a_1	b_2	c_2	d_1	e_1	f_2

$\gamma(AB, R_{a_2b_1}(r)) = (a_1, b_2, c_2, d_1, \perp, f_2)$

Fig. 2. The AB-complete horizontal decomposition of relation r of figure 1 and their corresponding AB-complete patterns. The closure of AB is $\theta(AB, r) = \{A, B, F\}$.

Proof. $\Leftarrow$: trivial since $X \subseteq \theta(X, r)$.

$\Rightarrow$: Consider $t, t' \in r'$ such that $t[X] = t'[X]$. Let us demonstrate that $t[\theta(X, r)] = t'[\theta(X, r)]$.
$\forall A \in \theta(X, r) - X$, $r \models X \rightarrow A$ (i.e. $X \rightarrow A$ is an FD). Thus, $r' \models X \rightarrow A$ since $r' \subseteq r$. As a consequence, $t[A] = t'[A]$. □

A consequence is that X and $\theta(X, r)$ define the same X-complete horizontal decomposition of r. This leads to the following corollary.

Corollary 1. $\Gamma(X, r) = \Gamma(\theta(X, r), r)$.

Example 2 (X-complete patterns)

In the relation of figure 1, the fragment relation $r_{AB} = \{t_1, t_2\}$ is AB-complete. The AB-complete horizontal decomposition of r is
$R_{AB}(r) = \{\{t_1, t_2\}, \{t_3, t_4\}, \{t_5, t_6, t_7, t_8\}, \{t_9, t_{10}\}\}$. This decomposition is shown on figure 2.

The set of AB-complete patterns is $\Gamma(AB, r) = \{(a_1, b_1, c_1, d_1, \perp, f_1),$
$(a_2, b_1, c_2, d_2, \perp, f_1), (a_2, b_2, \perp, \perp, e_1, f_2), (a_1, b_2, c_2, d_1, \perp, f_2)\}$.

The closure of AB is $\theta(AB, r) = \{A, B, F\}$. Note that this is coherent with the FD $B \rightarrow F$.

3 Link between CFDs, FDs and ARs

Here we show the relation between CFDs and other usual dependencies.

FDs: An FD $X \rightarrow A$ can be seen as a CFD $(X \rightarrow A, t_p)$ where t_p is a single tuple with no constants, i.e. $t_p = Top(X \cup \{A\})$. In other words, $r \models X \rightarrow A$ if and only if $r_{X \rightarrow A} = r$ (and thus $r_{X \not\rightarrow A} = \emptyset$).

ARs: An AR is a dependency $(X_1 = b_1) \wedge \cdots \wedge (X_k = b_k) \rightarrow (A = a)$ meaning that for any tuple $t \in r$, if $t[X_1] = b_1$ and $\cdots$ and $t[X_k] = b_k$ then $t[A] = a$. Note that for this kind of dependency, contrary to FDs and CFDs, no agreement is needed among tuples. The *support* of an AR is the number of tuples of the relation satisfying the dependency. An AR can be expressed using a CFDs which tableau consists in a single pattern tuple containing only constant or empty values.

Example 3. In the relation of figure 1, the AR $(A = a_1, B = b_2 \rightarrow C = c_2)$ could be rewritten as the CFD $(AB \rightarrow C, (a_1, b_2, c_2, \perp, \perp, \perp))$. The support of this AR is 2 corresponding to the size of the fragment relation obtained using the pattern tuple $(a_1, b_2, c_2, \perp, \perp, \perp)$.

Lemma 2. *Let r be a relation, $X \subseteq Attr(r)$, such that r is X-complete. Then the following assertions are equivalent:*

1. $(X \rightarrow A, \gamma(X, r))$ *is a CFD of* r
2. $X \rightarrow A$ *is an FD of* r
3. $(X \rightarrow A, \gamma(X, r))$ *is an AR of* r

Proof. If $\mid r \mid = 1$, the lemma trivially holds. We thus consider that r contains at least two tuples t, t'. Since r is X-complete, we have r is $\theta(X, r)$-complete, $R_X(r) = \{r\}$, and $\Gamma(X, r) = \{\gamma(X, r)\}$. If $A \notin \theta(X, r)$, all assertions are false (and thus equivalent). Suppose that $A \in \theta(X, r)$. Thus, $\forall t, t' \in r$, we have $t(X \cup \{A\}) = t'(X \cup \{A\})$. □

Lemma 2 states that when a relation is X-complete, FDs, CFDs and ARs of the form $X \rightarrow A$ are equivalent. The next theorem characterizes the link between FDs, CFDs and ARs when the relation is not X-complete.

Theorem 1. *Let r be a relation, $X \subseteq Attr(r)$, $A \in Attr(r) \setminus X$ and $T_p = \{t_p \in \Gamma(X, r) \mid t_p[A] \neq \perp\}$. The following assertions are equivalent:*

1. $(X \rightarrow A, T_p)$ *is a CFD of* r
2. $X \rightarrow A$ *is an FD of* r_{T_p}
3. *For any* $r' \in R_X(r_{T_p})$, $(X \rightarrow A, \gamma(X, r'))$ *is an AR of* r

Proof. 1 ⇒ 2: We suppose that $(X \rightarrow A, T_p)$ is a CFD of r. Thus, $\forall t_p \in T_p$, we have $r_{t_p} \models X \rightarrow A$. Since $r_{T_p} = \bigcup_{t_p \in T_p} r_{t_p}$, we have $r_{T_p} \models X \rightarrow A$.

2 ⇒ 3: We suppose that $r_{T_p} \models X \rightarrow A$. We thus have $\forall r' \in R_X(r_{T_p})$, r' is X-complete. Since $r' \models X \rightarrow A$, r' is $X \cup \{A\}$-complete. Hence, $(X \rightarrow A, \gamma(X, r'))$ is an AR of r'. Since for any $t' \notin r'$ we have $t'(X) \neq \gamma(X, r')(X)$ we conclude that $(X \rightarrow A, \gamma(X, r'))$ is a CFD of r.

3 ⇒ 1: We suppose that $\forall r' \in R_X(r_{T_p})$, $X \rightarrow A$ is an AR of r. From Lemma 2 we conclude that $r' \models X \rightarrow A$. Since $r_{T_p} = \bigcup_{r' \in R_X(r_{T_p})} r'$, we have $r_{T_p} \models X \rightarrow A$. Moreover, $\forall t'_p \in \Gamma(X, r)$ such that $t'_p \notin T_p$, we have $t'_p[A] = \perp$. Thus, $r_{t'_p} \not\models X \rightarrow A$. And thus $r - r_{T_p} \not\models X \rightarrow A$. Therefore, we conclude that $(X \rightarrow A, T_p)$ is a CFD of r. □

Theorem 1 is quite important for understanding the link between ARs, CFDs and FDs. Indeed, it leads to a hierarchy among those dependencies:

- an AR $(X \rightarrow A, t_p)$ is a dependency that holds on at least one fragment relation of r which is X-complete. In other words, there exists at least one fragment relation which is X-complete and such that tuples agree on attributes X *and* A*.*
- *a CFD* $(X \rightarrow A, T_p)$ *is a dependency that holds on some fragment relations of* r *which are* X*-complete. It can thus be viewed as the union of ARs holding on those fragment relations. In other words, on each of those fragment relations where tuples agree on* X*, they also agree on* A*.*
- *an FD is a dependency that holds on all fragment relations of* r *which are* X*-complete. It can thus be viewed as the union of ARs holding on all those fragment relations. In other words, in any fragment relation which is* X*-complete, tuples agree on attribute* A*.*

Agreement or not on an attribute A for tuples in an X-complete fragment relation is thus the core condition for a dependency $X \rightarrow A$ to hold or not in this fragment relation.

Definition 6 (A-Valid and A-Invalid X-complete pattern tuples). *An* X*-complete pattern tuple* t_p *is said to be* A-valid towards A *if and only if attribute* A *is defined in* t_p*.*

We denote by $\Psi(X \rightarrow A)$ *the set of all* A*-valid* X*-complete pattern tuples. More formally:* $\Psi(X \rightarrow A) = \{t_p \in \Gamma(X, r) \mid t_p[A] \neq \perp\}$*.*

Dually, we denote by $\Psi(X \not\rightarrow A)$ *the set of all* A*-invalid* X*-complete pattern tuples:* $\Psi(X \not\rightarrow A) = \Gamma(X, r) - \Psi(X \rightarrow A)$*.*

In other words, an X-complete pattern tuple is said A-valid if in its corresponding X-complete relation all tuples agree on A. Given a set of attributes X we can thus decompose the global relation in two subsets of fragment relations: those agreeing on A (represented by the set of their X-complete pattern tuples $\Psi(X \rightarrow A)$) and those that do not agree on A (represented by the set of their X-complete pattern tuples $\Psi(X \not\rightarrow A)$).

Thus, for any dependency $X \rightarrow A$, the set $\Psi(X \rightarrow A)$ can be interpreted as the pattern tableau selecting the fragment relation of r on which the dependency holds. In other words, any CFD $(X \rightarrow A, T_p)$ could be rewritten as $(X \rightarrow A, \Psi(X \rightarrow A))$ (the CFDs are equivalent).

If $\Psi(X \rightarrow A)$ is empty, the dependency does not hold in any fragment relation (denoted by $X \not\rightarrow A$). We will say that $X \rightarrow A$ is a *pure AR* if it holds exactly on one X-complete fragment relation (i.e. $\Psi(X \rightarrow A)$ contains only one pattern tuple). We will say that a CFD is a *pure CFD* if it holds on at least two X-complete fragment relations and does not hold on at least one X-complete fragment relations (i.e. $\Psi(X \rightarrow A)$ contains at least two pattern tuples and $\Psi(X \not\rightarrow A)$ is not empty). In other words, a pure AR does not generalize into a CFD, while a pure CFD does not generalize into an FD. This hierarchy is summarized in the following table.

Dependency	Type	$\Psi(X \to A)$?	$\Psi(X \not\to A)$?
$X \not\to A$		$\Psi(X \to A) = \emptyset$	
$X \to A$	Pure AR	$\mid \Psi(X \to A) \mid = 1$	$\mid \Psi(X \not\to A) \mid \geq 1$
	Pure CFD	$\mid \Psi(X \to A) \mid \geq 2$	$\mid \Psi(X \not\to A) \mid \geq 1$
	FD		$\Psi(X \not\to A) = \emptyset$

The main consequence of this hierarchy is that any algorithm which computes ARs could be adapted in order to compute CFDs and FDs. Another interesting consequence is that, when mining ARs in a relation, one could reduce the number of generated ARs: indeed some of those ARs are in fact CFDs or FDs and thus could be regrouped into a single dependency! Algorithms could reduce the amount of generated dependency by generating only the pure ARs, the pure CFDs and FDs. Moreover, such generation could ease the expertise work since some extra information is present: a pure AR is a dependency which holds only for one precise combination of values of attributes X while, on the opposite, an FD holds for any combination of values of attributes X.

Example 4 (Link between FDs, CFDs and ARs)
The CFD $\varphi_4 = (E \to A, \{(a_2, \perp, \perp, \perp, e_3, \perp,\})$ is a pure AR of the relation of figure 1.
$\varphi_2 = (AB \to C, \{(a_1, b_1, c_1, \perp, \perp, \perp), (a_2, b_1, c_2, \perp, \perp, \perp),$
$(a_1, b_2, c_2, \perp, \perp, \perp)\})$ is a pure CFD of the same relation. Indeed, we can note that $AB \to C$ is an AR in the fragment relations $R_{a_1 b_1}(r)$, $R_{a_2 b_1}(r)$ and $R_{a_1 b_2}(r)$ (see figure 2). In the same way, $AB \to C$ is an FD for the fragment relation $r_{\varphi_2} = R_{a_1 b_1}(r) \cup R_{a_2 b_1}(r) \cup R_{a_1 b_2}(r)$. Thus $\mid \Psi(AB \to C) \mid = 3$ and $\mid \Psi(AB \not\to C) \mid = 1$: $AB \to C$ is a pure CFD or r.

Now consider attributes AD of the relation in figure 1.
$\Gamma(AD, r) = \{ (a_1, \perp, \perp, d_1, \perp, \perp), (a_2, b_2, \perp, d_1, e_1, f_2),$
$(a_2, \perp, \perp, d_2, \perp, \perp)\}.$
From this, we see that $\Psi(AD \to C) = \emptyset$. As consequence, there are no CFDs in r of the form $(AD \to C, T_p)$.

Now consider the attributes ACE.
$\Gamma(ACE, r) = \{ (a_1, b_1, c_1, d_1, e_1, f_1), (a_1, b_1, c_1, d_2, e_2, f_1),$
$(a_1, b_2, c_2, d_1, e_1, f_2), (a_1, b_2, c_2, d_1, e_2, f_2),$
$(a_2, b_2, c_1, \perp, e_1, f_2), (a_2, b_2, c_2, \perp, e_1, f_2),$
$(a_2, b_1, c_2, d_2, e_2, f_1), (a_2, b_1, c_2, d_2, e_3, f_1)\}$
Hence we have $\Psi(ACE \not\to B) = \emptyset$. Thus, $ACE \to B$ is an FD of the relation.

4 Extending the Hierarchy to Approximate FDs and Approximate ARs

An AFD [15] is an FD that almost holds: some tuples of r invalidate the FD, representing errors or exceptions to the rule. Several ways of defining the approximateness of a dependency $X \to A$ have been used. The most common accepted measure is the g_3 *error*, representing the fraction of rows that should be removed from r in order to have $r \models X \to A$. :

$$g_3(X \to A) = 1 - \frac{(max\{\mid s \mid \text{ s.t. } s \subseteq r \wedge s \models X \to A\})}{\mid r \mid}$$

We call *support* of an AFD $X \to A$, denoted by $Support(X \to A)$, the size of the maximal fragment relation s_{max} such that $s_{max} \models X \to A$. The g_3 error could thus be rewritten:

$$g_3(X \to A) = 1 - \frac{Support(X \to A)}{\mid r \mid}$$

An approximate AR is, in the same way, an AR which almost holds on r: some tuples of r invalidate the AR. Measure of approximateness of an AR $(X \to A, t_p)$ is the *confidence* $conf((X \to A, t_p))$, expressing the conditional probability for a tuple $t \in r$ of having $t[X \cup \{A\}] = t_p[X \cup \{A\}]$ knowing that $t[X] = t_p[X]$.

$$conf((X \to A, t_p)) = \frac{\mid r_{t_p(X \cup \{A\})} \mid}{\mid r_{t_p(X)} \mid}$$

Obviously, given a pure CFD $(X \to A, T_p)$, $X \to A$ is an AFD. Moreover, $r_{X \to A} \subseteq s_{max}$. Thus we have $\mid r_{T_p} \mid = \mid r_{X \to A} \mid \leq Support(X \to A)$. Missing tuples might be found in $r_{X \not\to A}$.

For any pattern tuple t_p in $\Psi(X \not\to A)$, $r_{t_p} \not\models X \to A$ by definition. However, if we consider $R_A(r_{t_p})$, the A-complete horizontal decomposition of r_{t_p}, we have: $\forall r' \in R_A(r_{t_p}), r' \models X \to A$. Thus, $r_{T_p} \cup r' \models X \to A$. If we take $r' \in R_A(r_{t_p})$ such that $\mid r' \mid$ is maximal, then $r_{T_p} \cup r' \subseteq s_{max}$.

Proposition 2. *Given a CFD $(X \to A, T_p)$:*

$$Support(X \to A) = \mid r_{T_p} \mid + \sum_{t_p \in \Psi(X \not\to A)} max(\{\mid r' \mid \ s.t.\ r' \in R_A(r_{t_p})\})$$

Proof. $\forall t_p \in \Psi(X \not\to A)$ and $\forall r' \in R_A(r_{t_p})$ we have $r' \models X \to A$. Thus, $r_{T_p} \cup r' \models X \to A$. Let $r_{max} = \bigcup_{t_p \in \Psi(X \not\to A)} r'$ such that $r' \in R_A(r_{t_p})$ and $\mid r' \mid$ is maximal. We thus have $r_{T_p} \cup r_{max} \models X \to A$ and for all $t \in (r - (r_{T_p} \cup r_{max}))$ we have $(r - (r_{T_p} \cup r_{max})) \not\models X \to A$.

As a consequence, $Support(X \to A) = \mid r_{T_p} \cup r_{max} \mid$. □

Given a CFD $(X \to A, T_p)$ and a pattern tuple t_p in $\Psi(X \not\to A)$ we have seen that $\forall r' \in R_A(r_{t_p}), r' \models X \to A$. In other words, $X \to A$ is an AR of r'. It is also an approximate AR of r which confidence is $\mid r' \mid / \mid r_{t_p} \mid$. Thus, approximate ARs with maximal confidence in each of the X-complete fragment relations of r define a fragment relation that participates in the support of the AFD.

Theorem 2

$$Support(X \to A) = \sum_{r' \in R_X(r)} \mid r'_{t_p} \mid$$

with $r'_{t_p} \in R_A(r')$ and $(X \to A, t_p)$ is an AR of r' with maximal confidence.

Proof. $\forall r' \in R_X(r)$ and $\forall r'' \in R_A(r')$ we have $r'' \models X \rightarrow A$. Thus $X \rightarrow A$ is an AR of r'' since r'' is $X \cup \{A\}$-complete.
If $r' = r''$ then $X \rightarrow A$ is an AR of r' with confidence 1.
If $r'' \subset r'$ then $X \rightarrow A$ is an approximate AR of r' with confidence < 1.
If $\mid r'' \mid$ is maximal in $R_A(r')$, then the confidence of the approximate AR $X \rightarrow A$ is maximal in r''. We conclude using previous proposition. □

In other words, an AFD can be seen as the union of ARs (one per X-complete fragment relation of the relation) with maximal confidence: this includes exact ARs and approximate ARs, while an FD is the union of all exact ARs. A consequence is that any algorithm which computes both exact and approximate ARs could be used to discover AFD and their support.

Example 5 (Link between AFDs, ARs and approximate ARs)
Consider the CFD:
$\varphi_2 = (AB \rightarrow C, \{(a_1, b_1, c_1, \perp, \perp, \perp), (a_2, b_1, c_2, \perp, \perp, \perp),$
$(a_1, b_2, c_2, \perp, \perp, \perp)\})$ of the relation in figure 1. The fragment relation satisfying the CFD φ_2 is $r_{AB\rightarrow C} = R_{a_1 b1}(r) \cup R_{a_2 b1}(r) \cup R_{a_1 b2}(r)$. Thus $\mid r_{AB\rightarrow C} \mid = 6$. (see figure 2).

If we consider $r_{AB\not\rightarrow C} = R_{a_2 b_2}(r)$, we could decompose it in ABC-complete fragment relations: $R_{a_2 b_2 c_1}(r) = \{t_5, t_6\}$ and $R_{a_2 b_2 c_2}(r) = \{t_7, t_8\}$. Each ABC-complete fragment relation satisfies the implication $AB \rightarrow C$. We can thus add one of the fragment relation to $r_{AB\rightarrow C}$ without invalidating $AB \rightarrow C$. Both fragment relations have the same number of tuples: 2. Thus, the support of the AFD $AB \rightarrow C$ is 8.

5 Constraint Satisfaction for Pattern Tableau Generation

We have seen that any CFD admits an equivalent CFD such that the pattern tableau consists only of pattern tuples with constant or empty tuples (lemma 1. Moreover, we have seen that any CFD can be rewritten into the equivalent CFD $(X \rightarrow A, \Psi(X \rightarrow A))$. The set $\Psi(X \rightarrow A)$ can be viewed as a selection query which returns the fragment relation satisfying $X \rightarrow A$. Any selection query returning a fragment relation satisfying $X \rightarrow A$ could thus be a valid pattern tableau for the CFD. The objective of this section is to study how to compute equivalent tableaux of a CFD of the form $(X \rightarrow A, \Psi(X \rightarrow A))$. We suppose that both $\Psi(X \rightarrow A)$ and $\Psi(X \not\rightarrow A)$ are known.

5.1 Generating All CFDs

The tableaux $\Psi(X \not\rightarrow A)$ is the query returning the fragment relation where $X \rightarrow A$ is not satisfied. We can thus view $\Psi(X \not\rightarrow A)$ as a set of constraints that should be satisfied in order *not* to have $X \rightarrow A$:

$$C(X \not\rightarrow A) = \bigvee_{t \in \Psi(X \not\rightarrow A)} \Big(\bigwedge_{B \in X} B = t[B] \Big)$$

If we take the complementary constraint, we thus obtain the set of constraints that should be satisfied in order to have $X \rightarrow A$ satisfied.

$$\overline{C(X \not\rightarrow A)} = \bigwedge_{t \in \Psi(X \not\rightarrow A)} \Big(\bigvee_{B \in X} B \neq t[B] \Big)$$

Theorem 3. *Given r a relation and $(X \rightarrow A, \Psi(X \rightarrow A))$ a CFD of r then for any model m of $\overline{C(X \not\rightarrow A)}$, the pair $(X \rightarrow A, m)$ defines a CFD of r.*

Proof. Consider m a model of $\overline{C(X \not\rightarrow A)}$
$\Rightarrow \forall t_p \in \Psi(X \not\rightarrow A), m \not\sqsubseteq t_p$
$\Rightarrow \forall t \in r_{X \not\rightarrow A}, m \not\sqsubseteq t$
$\Rightarrow r_m \subseteq r_{X \rightarrow A}$
$\Rightarrow r_m \models X \rightarrow A$
(Note that r_m might be empty). □

Any model of $\overline{C(X \not\rightarrow A)}$ defines a selection query such that the resulting fragment relation satisfies $X \rightarrow A$. Note that $r_m \subseteq r_{X \rightarrow A}$. Thus $(X \rightarrow A, m)$ and $(X \rightarrow A, \Psi(X \rightarrow A))$ might not be equivalent. In other words, $(X \rightarrow A, m)$ is a CFD of the relation that could be generalized into $(X \rightarrow A, \Psi(X \rightarrow A))$. Note also that any set M of models of $\overline{C(X \not\rightarrow A)}$ defines a CFD $(X \rightarrow A, M)$ of the relation. Thus, the problem of finding the CFDs of a relation can thus be expressed as finding the models of a particular constraint.

Problem 1 (All models).
Input : $\overline{C(X \not\rightarrow A)}$
Output : $M_{ALL} = \{m \mid m$ is a model of $\overline{C(X \not\rightarrow A)}\}$.

Note that the fragment relation selected by the model might be empty. In this case the CFD trivially holds. However, in order to be meaningful, it would be preferable to find models which do return a non empty fragment relation.

Problem 2 (All models in r).
Input : $\overline{C(X \not\rightarrow A)}$, relation r
Output : $M_{ALL}(r) = \{m \in M_{ALL} \mid \exists t \in r, m \sqsubseteq t\}$.

5.2 Generating CFDs Equivalent to $(X \rightarrow A, \Psi(X \rightarrow A))$

To define a model of $\overline{C(X \not\rightarrow A)}$ in the relation r, a pattern tuple t_p has to subsume at least one tuple of the relation and not subsume any A-invalid pattern tuples in $\Psi(X \not\rightarrow A)$. This condition can be simplified using pattern tuples in $\Psi(X \rightarrow A)$.

Proposition 3. *Given a relation r, $\Psi(X \rightarrow A)$ and $\Psi(X \not\rightarrow A)$, a pattern tuple m is a model of $\overline{C(X \not\rightarrow A)}$ in r if and only if $\exists t_p \in \Psi(X \rightarrow A)$ and $\forall t'_p \in \Psi(X \not\rightarrow A)$ we have $t'_p \not\sqsubseteq m$ and $m \sqsubseteq t_p$.*

Proof. A pattern tuple m is a model of $\overline{C(X \not\to A)}$
$\Leftrightarrow \forall t_p \in \Psi(X \not\to A)$, $m \not\sqsubseteq t_p$ (by definition).
We consider $Y \subseteq X$ the attributes of m which are not equal to $\perp$.
m is of a model in r if and only if m is a model and $r_m \neq \emptyset$.
$\Leftrightarrow \exists t \in r_{X \to A}$, $m \subseteq t$
$\Leftrightarrow \exists t \in r_{X \to A}$, $m[Y] = t[Y]$
$\Leftrightarrow \exists t \in r_{X \to A}$, $\exists t_p \in \Psi(X \to A)$, $t_p \sqsubset t$ and $m[Y] = t[Y]$
$\Leftrightarrow \exists t \in r_{X \to A}$, $\exists t_p \in \Psi(X \to A)$, $t_p[X] = t[X]$ and $m[Y] = t[Y]$
$\Leftrightarrow \exists t \in r_{X \to A}$, $\exists t_p \in \Psi(X \to A)$, $t_p[Y] = t[Y] = m[Y]$ (since $Y \subseteq X$)
$\Leftrightarrow \exists t \in r_{X \to A}$, $\exists t_p \in \Psi(X \to A)$, $m \sqsubseteq t_p \sqsubseteq t$
$\Leftrightarrow \exists t_p \in \Psi(X \to A)$, $m \sqsubseteq t_p$ □

Thus, any pattern tuple t_p in $\Psi(X \to A)$ defines a sub-constraint of $\overline{C(X \not\to A)}$, denoted by $\overline{C(X \not\to A)}_{t_p}$: only constant values in t_p are allowed for the models. It is easy to check that any model m of this sub-constraint is also a model of the global constraint. Moreover, all tuples subsumed by t_p are also subsumed by m. And m does not subsume tuples subsumed by pattern tuples in $\Psi(X \not\to A)$. Thus, if we pick a model m for each t_p in $\Psi(X \to A)$ we obtain a set of models M such that $(X \to A, M)$ is equivalent to $(X \to A, \Psi(X \to A))$. Note that the set of tuples subsumed by two models in M might overlap and thus some models of M might be redundant (i.e. the corresponding pattern tableau might be reduced).

While we are interested here in generating *all* equivalent tableaux of a CFD, authors in [14] are interested in finding the equivalent tableau of a CFD such that this tableau is optimal: the number of pattern tuples in the tableau is the smallest possible. They show that this problem is NP-complete.

5.3 Equivalence to the Problem of Keys Generation

We now show that finding models of $\overline{C(X \not\to A)}$ is equivalent to the problem of finding keys in a relation.

Definition 7 (Relations of models in r). *Given $t_p \in \Psi(X \to A)$, we define the relation $r_{t_p}(\Psi(X \not\to A)) = \{t_p(X) \sqcap t'_p(X) \mid t'_p \in \Psi(X \not\to A)\} \cup \{t_p(X)\}$.*

Any pattern m which subsumes $t_p(X)$ while not subsuming other tuples in the relation $r_{t_p}(\Psi(X \not\to A))$ is thus a model. In other words, keys of $r_{t_p}(\Psi(X \not\to A))$ are models of $\overline{C(X \not\to A)}$. This is formalized by next theorem.

Theorem 4. *The pattern tuple m_p defines a model of $\overline{C(X \not\to A)}$ in r if and only if there exists $t_p \in \Psi(X \to A)$ such that $\gamma(m_p, r_{t_p}(\Psi(X \not\to A))) = t_p(X)$.*

Proof. m_p model in r
$\Leftrightarrow \forall t'_p \in \Psi(X \not\to A), m_p \not\sqsubseteq t'_p$ and $r_m \neq \emptyset$
$\Leftrightarrow \forall t \in r_{t_p}(\Psi(X \not\to A))$ such that $t \neq t_p(X)$, $m_p \not\sqsubseteq t$ and $r_m \neq \emptyset$
$\Leftrightarrow \gamma(m_p, r_{t_p}(\Psi(X \not\to A))) = t_p(X)$ and $r_m \neq \emptyset$
And since $\exists t \in r$ such that $t_p(X) \sqsubset t$:
$\Leftrightarrow \gamma(m_p, r_{t_p}(\Psi(X \not\to A))) = t_p(X)$ □

Thus, to find models of $\overline{C(X \not\rightarrow A)}$, one could use any algorithm which finds keys of a relation. Finding just minimal keys could be more interesting since all keys could be inferred from minimal keys.

Problem 3 (All minimal models in r).
Input : $\overline{C(X \not\rightarrow A)}$, relation r
Output : $M_{MIN} = \{m \in M_{ALL}(r) \mid \forall m' \in M_{ALL}(r), m' \not\subseteq m\}$.

Finding all minimal models is equivalent to finding all minimal keys of the relations $r_{t_p}(\Psi(X \not\rightarrow A))$, for any $t_p \in \Psi(X \rightarrow A)$. Note that such minimal models define minimal generators of the pattern tuples in $\Psi(X \rightarrow A)$ and could thus be computed directly during the generation of the CFDs and FDs.

Example 6 (Constraint satisfaction for pattern tableaux)
Consider attributes ABC in the relation of figure 1.

$$\begin{aligned}\Gamma(ABC, r) = \{ &(a_1, b_1, c_1, d_1, \bot, f_1), (a_1, b_2, c_2, d_1, \bot, f_2),\\ &(a_2, b_1, c_2, d_2, \bot, f_1), (a_2, b_2, c_1, \bot, e_1, f_2),\\ &(a_2, b_2, c_2, \bot, e_1, f_2)\}\end{aligned}$$

We have:

$$\begin{aligned}\Psi(ABC \rightarrow D) = \{ &(a_1, b_1, c_1, d_1, \bot, f_1), (a_1, b_2, c_2, d_1, \bot, f_2),\\ &(a_2, b_1, c_2, d_2, \bot, f_1)\}\end{aligned}$$

and $\Psi(ABC \not\rightarrow D) = \{ (a_2, b_2, c_1, \bot, e_1, f_2), (a_2, b_2, c_2, \bot, e_1, f_2)\}$.

We thus obtain the following constraints:

$$\begin{aligned}\overline{C(ABC \not\rightarrow D)} = \quad &(A \neq a_2 \vee B \neq b_2 \vee C \neq c_1)\\ \wedge\ &(A \neq a_2 \vee B \neq b_2 \vee C \neq c_2)\end{aligned}$$

Consider $t_p = (a_1, b_1, c_1, d_1, \bot, f_1)$. We have $t_p(X) = (a_1, b_1, c_1, \bot, \bot, \bot)$

$$\begin{aligned}r_{t_p}(\Psi(ABC \not\rightarrow D)) = \{ &(\bot, \bot, c_1, \bot, \bot, \bot),\\ &(\bot, \bot, \bot, \bot, \bot, \bot),\\ &(a_1, b_1, c_1, \bot, \bot, \bot)\}.\end{aligned}$$

Minimal keys are $(a_1, \bot, \bot, \bot, \bot, \bot)$ and $(\bot, b_1, \bot, \bot, \bot, \bot)$. If we consider all pattern tuples in $\Psi(ABC \rightarrow D)$, we finally obtain the following CFD:

$\varphi_5 = (ABC \rightarrow D, \{\{(a_1, \top, \top, d_1, \bot, \bot), (\top, b_1, \top, \top, \bot, \bot)\}\})$.

Note that φ_5 could be simplified since $(A \rightarrow D, \{\{(a_1, \bot, \bot, d_1, \bot, \bot)\}\})$ is a CFD of r.

6 Discussion and Conclusion

The notion of CFD was originally introduced in [6]. Authors focused on implication inference and consistency. In their conclusion, they pointed out that discovering CFDs in a sample relation might be interesting for their purpose: this remark was the starting point of our work presented here. The CFDs were extended in a sequel paper [7], leading to a more expressive class of dependencies (eCFDs) which allow the use of disjonctions and inequalities in the expression of patterns. However, eCFDs are just a particular case of CGDs [2]. The notion of X-complete relation is defined in [11] and largely inspired us in our results.

In [15], the algorithm Tane uses partitions to discover FDs and AFDs. Their partitions are exactly X-complete relations. Their approach is similar to the A priori algorithm in the sense that the search space is the powerset lattice. One could easily improve the Tane algorithm by reducing the search space to the closed set lattice (in the same way as Close improves A priori). As a matter of fact, it is exactly what authors in [13] did to discover CFDs. In the same way, any algorithm which searches the closed set lattice could be used (as it has been done in [13]). Concerning efficient algorithms to discover closed sets or ARs, there are too many in the literature to put forward one. In the paper, we give link to some of them which inspired the others. The important result being that those algorithms can easily be adapted to find FDs, pure CFDs and pure ARs (exact and approximate).

The link between ARs and DFs was never (to our knowledge) formally established. It is well known however that algorithms which find ARs could be used to find FDs of a relation: such ARs discovery algorithm could simply be applied on the agree sets of the original relation [16]. In the same way, algorithms used to discover FDs could be easily adapted to discover ARs (either by a discretization of the relation or by slightly changing the comparison operator [15]). The formal link between ARs, FDs and (of course) CFDs is thus one of the main contribution of the present paper. FDs are the union of CFDs which in turn are the union of ARs, establishing a hierarchy among those dependencies. Using this hierarchy we can thus define a minimal non redundant cover of dependencies containing FDs, pure CFDs and pure ARs. In the same way, AFDs are union of ARs and approximate ARs with maximal confidence (one per horizontal decomposition of a relation according to a set of attributes).

We also establish the link between the problem of finding equivalent pattern tableaux of a CFD and the problem of finding keys of a relation.

The immediate next step to this paper is to implement our method in order to compare the number and the pertinence of the generated dependencies against the ARs generated by traditional data mining algorithms. Because of the hierarchy existing among FDs, CFDs and ARs, the number of dependencies generated should be drastically lesser. In the same way, by using different pattern tableaux for a same CFD, the work of interpreting the dependencies should be eased. Those assumptions should be validated through some experiments on real life applications.

We also intend to extend our study to eCFDs. Is there a link between ARs (or particular types of ARs) and eCFDs ? Establishing such a link would lead to an immediate application of the same approach to discover eCFDs. More challenging work is to show that any CGD could be inferred from ARs.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: VLDB, pp. 487–499 (1994)
2. Baudinet, M., Chomicki, J., Wolper, P.: Constraint-generating dependencies. J. Comput. Syst. Sci. 59(1), 94–115 (1999)

3. Beeri, C., Vardi, M.Y.: Formal systems for tuple and equality generating dependencies. SIAM J. Comput. 13(1), 76–98 (1984)
4. Beeri, C., Dowd, M., Fagin, R., Statman, R.: On the structure of armstrong relations for functional dependencies. Journal of the ACM 31, 30–46 (1984)
5. Bell, S., Brockhausen, P.: Discovery of data dependencies in relational databases. Technical Report LS-8 Report-14, University of Dortmund (1995)
6. Bohannon, P., Fan, W., Geerts, F., Jia, X., Kementsietsidis, A.: Conditional functional dependencies for data cleaning. In: ICDE, pp. 746–755 (2007)
7. Bravo, L., Fan, W., Geerts, F., Ma, S.: Increasing the expressivity of conditional functional dependencies without extra complexity. In: ICDE, pp. 516–525 (2008)
8. Chiang, F., Miller, R.: Discovering data quality rules. In: VLDB (2008)
9. Chomicki, J., Marcinkowski, J.: On the computational complexity of minimal-change integrity maintenance in relational databases. Inconsistency Tolerance, 119–150 (2005)
10. Codd, E.F.: Further normalizations of the database relational model. In: Rustin, R. (ed.) Data Base Systems, pp. 33–64. Prentice-Hall, Englewood Cliffs (1972)
11. De Bra, P., Paredaens, J.: An algorithm for horizontal decompositions. Inf. Process. Lett. 17(2), 91–95 (1983)
12. Fan, W., Geerts, F., Jia, X., Kementsietsidis, A.: Conditional functional dependencies for capturing data inconsistencies. ACM Trans. Database Syst. 33(2) (2008)
13. Fan, W., Geerts, F., Xiong, M., Lakshmanan, L.V.S.: Discovering conditional functional dependencies. In: ICDE (2009)
14. Golab, L., Karloff, H., Korn, F., Srivastava, D., Yu, B.: On generating near-optimal tableaux for conditional functional dependencies. In: VLDB (2008)
15. Huhtala, Y., Kärkkäinen, J., Porkka, P., Toivonen, H.: Tane: An efficient algorithm for discovering functional and approximate dependencies. The Computer Journal 42(2), 100–111 (1999)
16. Lopes, S., Petit, J.-M., Lakhal, L.: Discovering agree sets for database relation analysis. In: BDA (2000)
17. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering frequent closed itemsets for association rules. In: ICDT, pp. 398–416 (1999)
18. Petit, J.-M., Toumani, F., Boulicaut, J.-F., Kouloumdjian, J.: Towards the reverse engineering of denormalized relational databases. In: ICDE, pp. 218–227. IEEE Computer Society Press, Los Alamitos (1996)
19. Weddell, G.E.: Reasoning about functional dependencies generalized for semantic data models. ACM Transactions on Database Systems 17(1), 32–64 (1992)

Robust Elements in Rough Set Abstractions*

Christian Meschke

Institut für Algebra, TU Dresden, Germany
Christian.Meschke@mailbox.tu-dresden.de

Abstract. In [3] the classical rough set approach was generalized in the manner that lower and upper approximations were replaced by arbitrary kernel and closure operators respectively. Furthermore, the resulting lattices of *rough set abstractions* were described as *P-products*. This approach, though promising, needs additional research to become part of a unifying theory of Rough Set Theory and Formal Concept Analysis. For example, the role of *robust* elements and the possible existence of suitable negation operators were not addressed. We present results in these directions and on the structure of these lattices.

1 Introduction

In recent years there has been a mutual interest of bringing together Formal Concept Analysis and Rough Set Theory. Several authors made respective contributions. Many of them handle a topic they call *Rough Concept Analysis*. In [3] Ganter chose another approach. He proposed a way to generalize the classical rough set setting and described the resulting *rough set abstractions* with tools from Formal Concept Analysis.

The classical approach of Rough Set Theory [6] starts with an equivalence relation $\sim$ on a *universe* U. The pair $(U,\sim)$ is usually called an *approximation space*. Thereby $x \sim y$ is interpreted as x and y being indistinguishable. Given such an *indiscernibility* relation $\sim$ on U one defines

$$\begin{aligned} X_* &:= \{u \in U \mid [u]_\sim \subseteq X\}, \\ X^* &:= \{u \in U \mid [u]_\sim \cap X \neq \emptyset\}, \end{aligned}$$

and calls these two sets the *lower* and the *upper approximation* of $X \subseteq U$. Furthermore, the pair (X_*, X^*) is said to be the *rough set approximation* of X. A subset X is called *rough* if $X_* \subsetneq X^*$ holds. Otherwise X is called *crisp*. Hence, a subset of the universe is crisp if and only if it is the union of equivalence classes.

Ordering all approximations (X_*, X^*) by componentwise set inclusion (i.e. the canonical order of $\mathfrak{P}(U) \times \mathfrak{P}(U)$) leads to a complete lattice. These lattices are very similar to Boolean lattices, since they are direct products of chains having the length one or two. Here singleton equivalence classes correspond to chains of length one, whereas non-singleton classes correspond to chains of length two. As

* Supported by the DFG research grant no. GA 216/10-1.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 114–129, 2009.

an example take the identity on U as the indiscernibility relation. In this case all equivalence classes are singletons. Thus, the lower and upper approximations of a set equal the set itself. Hence, the lattice of all approximations is isomorphic to the power set lattice $\mathfrak{P}(U)$ which obviously is a direct product of chains having length one.

Obviously the mapping $X \mapsto X_*$ is an interior operator and $X \mapsto X^*$ is a closure operator. Ganter used this fact to generalize the notion of rough set approximations starting with so-called *kernel-closure-pairs* consisting of an arbitrary kernel and an arbitrary closure operator on U. In the underlying article we want to pick up this train of thoughts and examine properties of the resulting lattices. As proposed in [3], certain elements of the universe play a special role. We will call them *robust* elements. They generalize the elements having singleton equivalence classes in the classical setting. The robust elements are not the exclusive topic of this article. Anyhow we believe it increases their understanding. That is why the robust elements made it into the title.

We investigate basic structural properties of the lattices of rough set abstractions. With Proposition 1 we propose a perspicuous characterization of the rough set abstractions. Furthermore, we propose a simple sufficient condition under which all abstractions are generated by subsets of the universe, and propose possible negations of abstractions and their properties.

2 Rough Set Abstractions

Throughout this article $\mathcal{K}$ denotes a kernel system and $\mathcal{C}$ denotes a closure system on the universe U. The corresponding interior and closure operators are denoted by $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ respectively. Hence, for every subset X of the universe $\lfloor X\rfloor$ is the largest kernel out of $\mathcal{K}$ which is contained in X, whereas $\lceil X\rceil$ is the smallest closure from $\mathcal{C}$ that contains X. The complement $U\setminus X$ of X is denoted by $X^{\complement}$.

The question is how to generalize the notion of rough set approximations starting with such an arbitrary kernel-closure pair $(\lfloor\cdot\rfloor,\lceil\cdot\rceil)$. An obvious approach is to investigate all pairs $(\lfloor X\rfloor,\lceil X\rceil)$ with $X\subseteq U$. But this leads to unwanted differences to the classical setting of rough set theory. Firstly, the pairs $(\lfloor X\rfloor,\lceil X\rceil)$, ordered by componentwise set inclusion, do not necessarily form a lattice[1]. Secondly, even if these pairs form a lattice, infimum and supremum need not be the componentwise infimum and supremum induced by the corresponding operations in $\mathcal{K}$ (first component) and $\mathcal{C}$ (second component). In [3] these problems were solved in the following way:

Since $\mathcal{K}$ and $\mathcal{C}$, each ordered by $\subseteq$, are complete lattices, their direct product $\mathcal{K}\times\mathcal{C}$ also is a complete lattice with

$$\bigvee_{t\in T}(K_t,C_t)=\left(\bigcup_{t\in T}K_t\,,\left\lceil\bigcup_{t\in T}C_t\right\rceil\right)\quad\text{and}\quad\bigwedge_{t\in T}(K_t,C_t)=\left(\left\lfloor\bigcap_{t\in T}K_t\right\rfloor,\bigcap_{t\in T}C_t\right)$$

for $(K_t,C_t)\in\mathcal{K}\times\mathcal{C}$ $(t\in T)$. By $\mathit{\Gamma}:=\mathit{\Gamma}_{\mathcal{K},\mathcal{C}}$ we denote the complete sublattice of $\mathcal{K}\times\mathcal{C}$ that is generated by all the pairs $(\lfloor X\rfloor,\lceil X\rceil)$ with $X\subseteq U$. The

[1] See for instance Example 1 on page 127.

pairs (K, C) from Γ are called **rough set abstractions** (just **abstractions** for short). Furthermore, we say that $X \subseteq U$ **generates** the abstraction $(\lfloor X \rfloor, \lceil X \rceil)$. Abstractions that are generated by a subset of U are called **set-generated**. We shall see in Proposition 1 that this notion of a rough set abstraction is a very natural generalization of the rough set approximations in classical rough set theory. For that we need the notion of *robust* elements.

We put $R_{\langle\mathcal{K}\rangle} := \{x \in U \mid \{x\} \in \mathcal{K}\}$ and $R^{\langle\mathcal{C}\rangle} := \{x \in U \mid U \setminus \{x\} \in \mathcal{C}\}$. Elements from $R_{\langle\mathcal{K}\rangle}$ are called **kernel robust**. Dually, the elements from $R^{\langle\mathcal{C}\rangle}$ are called **closure robust**. The elements from $R := R_{\langle\mathcal{K}\rangle} \cap R^{\langle\mathcal{C}\rangle}$ are simply called **robust**. Furthermore, for $\rho = (K, C) \in \Gamma$

- we call K the **positive region** of ρ,
- we call $U \setminus C$ the **negative region** of ρ,
- we call $C \setminus K$ the **boundary** of ρ, or the **uncertain region** of ρ,
- we call $U \setminus K$ the **nonpositive region** of ρ, and
- we call C the **nonnegative region** of ρ.

Proposition 1. *A pair (K, C) consisting of a kernel $K \in \mathcal{K}$ and a closure $C \in \mathcal{C}$ is a rough set abstraction from $\Gamma_{\mathcal{K},\mathcal{C}}$ if and only if $K \subseteq C$ holds and $C \setminus K$ does not contain a robust element.*

Proof. We begin this proof with showing that any abstraction (K, C) fulfills the properties $C \subseteq K$ and $(C \setminus K) \cap R = \emptyset$. Therefore, we have to show that every set-generated abstraction fulfills the two properties. Furthermore we have to show that the two properties are preserved by suprema and infima in $\mathcal{K} \times \mathcal{C}$. Obviously, for any $A \subseteq U$ it holds that $\lfloor A \rfloor \subseteq \lceil A \rceil$. Furthermore, for any $x \in \lceil A \rceil \setminus \lfloor A \rfloor$ the premise $x \in R$ implies $x \in A$ and $x \notin A$. Let $\mathcal{R} = \{(K_j, C_j) \mid j \in J\} \subseteq \mathcal{K} \times \mathcal{C}$ be a set of pairs fulfilling the two aforesaid properties, i.e. it holds that $K_j \subseteq C_j$ and $\emptyset = R \cap (C_j \setminus K_j)$. We define

$$\bigvee \mathcal{R} = \left(\bigcup_{j \in J} K_j, \left\lceil \bigcup_{j \in J} C_j \right\rceil\right) =: (K_{\text{sup}}, C_{\text{sup}}) \qquad \text{and}$$

$$\bigwedge \mathcal{R} = \left(\left\lfloor \bigcap_{j \in J} K_j \right\rfloor, \bigcap_{j \in J} C_j\right) =: (K_{\text{inf}}, C_{\text{inf}}).$$

It is easy to see that $K_{\text{sup}} \subseteq C_{\text{sup}}$ and $K_{\text{inf}} \subseteq C_{\text{inf}}$ hold. Let us assume $x \in C_{\text{sup}} \setminus K_{\text{sup}}$ is robust. This implies $x \in \bigcup_{j \in J} C_j$ and therefore there is a $\iota \in J$ with $x \in C_\iota$. Since x is robust and (K_ι, H_ι) fulfills the two properties from above it follows $x \in K_\iota$. But this contradicts $x \notin K_{\text{sup}}$. Suppose there is a robust element $x \in C_{\text{inf}} \setminus K_{\text{inf}}$. This implies $x \notin \bigcap_{j \in J} K_j$. Hence there is a $\iota \in J$ with $x \notin K_\iota$. Analogous to things shown above it follows $x \notin C_\iota$ and therefore $x \notin C_{\text{inf}}$. A contradiction.

We now have to show that any pair (K, C) from $\mathcal{K} \times \mathcal{C}$ fulfilling the two properties from above really is an abstraction, i.e. there are pairs of the form

$(\lfloor X\rfloor, \lceil X\rceil)$ that generate (K, C). We therefore put $A := C \cap R_{\langle\mathcal{K}\rangle}$ and $B := A \setminus K$. Since B does not contain a closure robust element it follows that

$$\bigwedge_{y \in B} (\lfloor U \setminus \{y\}\rfloor, \lceil U \setminus \{y\}\rceil) \;=\; (\lfloor U \setminus B\rfloor, U) \;=\; (\lfloor A^{\complement} \cup K\rfloor, U)$$

is an abstraction. Furthermore, it holds that

$$(A, C) \;=\; \bigvee_{x \in C} (\lfloor\{x\}\rfloor, \lceil\{x\}\rceil),$$

which implies that also

$$(A \cup K, C) \;=\; (\lfloor K\rfloor, \lceil K\rceil) \vee (A, C)$$

is an abstraction. Thus,

$$(K, C) \;=\; (A \cup K, C) \;\wedge\; (\lfloor A^{\complement} \cup K\rfloor, U)$$

is an abstraction as well. □

In the remaining part of this section we analyze the basic structure of lattices of rough set abstractions. If one chooses the trivial kernel system $\{\emptyset\}$, one obtains $\Gamma_{\mathcal{K},\mathcal{C}} \cong \mathcal{C}$. Similarly, $\mathcal{C} = \{U\}$ implies $\Gamma_{\mathcal{K},\mathcal{C}} \cong \mathcal{K}$. Hence, the reader should not expect that lattices of the form $\Gamma_{\mathcal{K},\mathcal{C}}$ do have any special lattice theoretic properties. It is more advisable to investigate how the interaction of $\mathcal{K}$ and $\mathcal{C}$ mirrors the structure of $\Gamma_{\mathcal{K},\mathcal{C}}$.

Corollary 1. *For every subset X of U the following holds:*

(i) $X \subseteq R_{\langle\mathcal{K}\rangle} \;\Rightarrow\; X \in \mathcal{K}$,

(ii) $X \subseteq R^{\langle\mathcal{C}\rangle} \;\Rightarrow\; X^{\complement} \in \mathcal{C}$,

(iii) $X \subseteq R \;\Rightarrow\; X \in \mathcal{K}$ *and* $X^{\complement} \in \mathcal{C}$,

(iv) $X \cup R^{\complement}$ *is the greatest closure C, for which $(\lfloor X\rfloor, C)$ is an abstraction,*

(v) $X \cap R$ *is the smallest kernel K, for which $(K, \lceil X\rceil)$ is an abstraction,*

(vi) $(X \cap R, X \cup R^{\complement})$ *is a rough set abstraction with the boundary $R^{\complement}$.*

Proof. The first three items are trivial. From (iii) follows that $X \cup R^{\complement} = (X^{\complement} \cap R)^{\complement}$ is a closure. Furthermore, $(X \cup R^{\complement}) \setminus \lfloor X\rfloor = (X \setminus \lfloor X\rfloor) \cup (R^{\complement} \setminus \lfloor X\rfloor) \subseteq R^{\complement}$ implies that $(\lfloor X\rfloor, X \cup R^{\complement})$ is an abstraction. For $(K, C) \in \Gamma$ with $K = \lfloor X\rfloor$ it holds that $C = K \cup (C \setminus K) \subseteq \lfloor X\rfloor \cup R^{\complement} \subseteq X \cup R^{\complement}$. The fifth statement can be shown analogously, whereas (vi) holds because of $(X \cup R^{\complement}) \setminus (X \cap R) = R^{\complement}$. □

Corollary 2. *$\mathcal{K}$ as well as $\mathcal{C}$ are isomorphic to subintervals of $\Gamma_{\mathcal{K},\mathcal{C}}$. Corresponding isomorphisms are given via:*

(i) $\varphi : \mathcal{C} \to ((\emptyset, \lceil\emptyset\rceil), (R, U))$ *with* $C \mapsto (C \cap R, C)$,

(ii) $\psi : \mathcal{K} \to ((\emptyset, R^{\complement}), (\lfloor U \rfloor, U))$ *with* $K \mapsto (K, K \cup R^{\complement})$.

(iii) The intersection $((\emptyset, R^{\complement}), (R, U))$ *of the two intervals from (i) and (ii) is again a nonempty subinterval of* Γ*. It is isomorphic to the powerset lattice* $\mathfrak{P}(R)$.

Proof. It follows directly from the previous Corollary 1 that φ and ψ are well defined. Since φ (ψ) equals the identity map regarding the second (first) component it is an order embedding. Hence, it remains to show that both mappings are onto. For every $(K, C) \in \Gamma$ it holds that $C \cap R \subseteq K$. Thus, with the additional assumption $K \subseteq R$ it follows $C \cap R = K$. Analogously it always holds $C \subseteq K \cup R^{\complement}$, and thus the assumption $R^{\complement} \subseteq C$ implies $C = K \cup R^{\complement}$. Statement (iii) holds, since the interval in question consists exactly of all the rough set abstractions $(T, R^{\complement} \cup T)$ with $T \subseteq R$. □

Let $\mathcal{K}^{\vartriangle}$ be the described subinterval of Γ that is isomorphic to $\mathcal{K}$, and let $\mathcal{C}_{\triangledown}$ be the subintervall isomorphic to $\mathcal{C}$. We call $\mathcal{K}^{\vartriangle} \cup \mathcal{C}_{\triangledown}$ the *spine* of Γ. We call Γ *slim* if it equals its spine. Furthermore, the interval $\mathcal{C}_{\triangledown} \cap \mathcal{K}^{\vartriangle}$ might be interpreted as the *middle* of Γ. Statement (iii) in connection with Proposition 1 says that the rough set abstractions from the middle are exactly the rough set abstractions that have the largest occurring boundary, namely $R^{\complement}$. It is possible to generalize statement (iii) in the way that the set

$$\Gamma|_B \;:=\; \{(K, C) \in \Gamma \mid (C \setminus K) \supseteq B\}$$

of all abstractions with a boundary containing a fixed minimum boundary B is empty or is an intervall in Γ. It is easy to see that $\Gamma|_B$ is nonempty if and only if $B \subseteq R^{\complement}$ holds.

Lemma 1. *Let* $\partial(\rho)$ *denote the boundary of a given* $\rho \in \Gamma$*. For* $\rho_1, \rho_2 \in \Gamma$ *it holds that*

$$\partial(\rho_1 \wedge \rho_2) \quad \supseteq \quad \partial(\rho_1) \cap \partial(\rho_2) \quad \subseteq \quad \partial(\rho_1 \vee \rho_2).$$

Proof. For $(K_1, C_1) := \gamma_1$ and $(K_2, C_2) := \gamma_2$ it holds that

$$\begin{aligned}
\partial(\rho_1 \wedge \rho_2) &= (C_1 \cap C_2) \setminus \lfloor K_1 \cap K_2 \rfloor \\
&\supseteq (C_1 \cap C_2) \setminus (K_1 \cap K_2) \\
&\supseteq (C_1 \setminus K_1) \cap (C_2 \setminus K_2) \qquad\qquad \text{and} \\
\partial(\rho_1 \vee \rho_2) &= \lceil C_1 \cup C_2 \rceil \setminus (K_1 \cup K_2) \\
&= (\lceil C_1 \cup C_2 \rceil \setminus K_1) \cap (\lceil C_1 \cup C_2 \rceil \setminus K_2) \\
&\supseteq (C_1 \setminus K_1) \cap (C_2 \setminus K_2).
\end{aligned}$$

□

Proposition 2. *For every* $B \subseteq R^{\complement}$ *the set* $\Gamma|_B$ *is a nonempty subintervall of* Γ*. It holds that* $\Gamma|_B = \Gamma_{\mathcal{K}_B, \mathcal{C}_B}$ *where*

$$\mathcal{K}_B \;:=\; \{K \in \mathcal{K} \mid K \subseteq B^{\complement}\} \qquad \textit{and} \qquad \mathcal{C}_B \;:=\; \{C \in \mathcal{C} \mid B \subseteq C\}.$$

Proof. By Corollary 1 (iv) and (v) we get that $(\emptyset, \lceil B \rceil)$ and $(\lfloor B^{\complement} \rfloor, U)$ are abstractions. For $(K, C) \in \Gamma$ it holds that

$$\begin{aligned} (\emptyset, \lceil B \rceil) \sqsubseteq (K, C) \sqsubseteq (\lfloor B^{\complement} \rfloor, U) &\iff K \subseteq \lfloor B^{\complement} \rfloor \text{ and } \lceil B \rceil \subseteq C \\ &\iff K \subseteq B^{\complement} \quad \text{and} \quad B \subseteq C \\ &\iff C \setminus K \supseteq B \\ &\iff (K, C) \in \Gamma|_B . \end{aligned}$$

Hence, $\Gamma|_B$ indeed is a nonempty subintervall of Γ. For proving that it can be considered as the system of rough set abstractions $\Gamma_{\mathcal{K}_B, \mathcal{C}_B}$ we just have to show that in both cases ($\mathcal{K}$ & $\mathcal{C}$ vs. $\mathcal{K}_B$ & $\mathcal{C}_B$) the same robust elements occur. Thus, we have to show

$$\{r\} \in \mathcal{K} \text{ and } U \setminus \{r\} \in \mathcal{C} \iff \{r\} \in \mathcal{K}_B \text{ and } U \setminus \{r\} \in \mathcal{C}_B .$$

If $\{r\} \in \mathcal{K}$ and $U \setminus \{r\} \in \mathcal{C}$ holds, both statements $\{r\} \notin \mathcal{K}_B$ and $U \setminus \{r\} \notin \mathcal{C}_B$ each are equivalent to $r \in B$. Hence, the premise $B \subseteq R^{\complement}$ directly yields the contradiction that r is not robust (regarding to $\mathcal{K}$ and $\mathcal{C}$). The backward direction is trivial. □

An abstraction (K, C) is contained in the spine of Γ if and only if $K = C \cap R$ or $C = K \cup R^{\complement}$ holds. These are exactly the abstractions with a boundary that is already maximal, or that can be extended in at most one direction: either into the positive region, or into the negative region.

Corollary 3. *The spine is a complete sublattice of Γ.*

Proof. The spine $\mathcal{K}^{\Delta} \cup \mathcal{C}_{\nabla}$ is a sublattice of Γ since it is the union of an order filter and an order ideal. Furthermore, the spine contains the least and the greatest abstractions. Since one can split up the supremum of $\mathcal{R} \subseteq \mathcal{K}^{\Delta} \cup \mathcal{C}_{\nabla}$ into the supremum $(\mathcal{R} \cap \mathcal{K}^{\Delta}) \vee (\mathcal{R} \cap \mathcal{C}_{\nabla})$ of at most two elements from the spine, we get that it is closed under arbitrary suprema. Analogously one shows that $\bigwedge \mathcal{R}$ also belongs to the spine. □

3 Contextual Representation

The main result of [3] was describing the lattice Γ as a P-product of $\mathcal{K}$ and $\mathcal{C}$. The *P-product* of complete lattices is a subdirect product. It can be described via formal contexts called *P-fusions*; see [4]. The mentioned approach starts with describing $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ via formal contexts. Since every closure system can be described as the set of extents of a formal context, there is a formal context $\mathbb{C} = (U, M, J)$ such that $\mathcal{C} = \mathrm{Ext}(\mathbb{C})$. Dually, the kernel system $\mathcal{K}$ can be represented as the set of complements of extents of a context $\mathbb{K} = (U, M, I)$. Thus, for every $X \subseteq U$ it holds that $\lfloor X \rfloor = X^{\complement II \complement}$ and $\lceil X \rceil = X^{JJ}$. If one has chosen such two representing contexts, $\mathbb{K}$ and $\mathbb{C}$ the system $\Gamma_{\mathcal{K},\mathcal{C}}$ of all rough set abstractions will also be denoted with $\Gamma_{\mathbb{K},\mathbb{C}}$. For technical reasons we have to assume $U \cap (M \cup N) = \emptyset$. If this is not the case, one clearly can enforce it by replacing M, N or U by disjoint copies.

	U	N
M	I^d	$\perp$
U	$\overset{*}{\times}$	J

Fig. 1. The representing context of Γ

Definition 1 (see [3]). *We define $\mathbb{G}$ to be the context displayed in Figure 1, whereas I^d denotes the dual relation of I,*

$$\begin{aligned} \perp &:= \{(m,n) \in M \times N \mid m^I \cup n^J = U\}, \text{ and} \\ \overset{*}{\times} &:= (U \times U) \setminus \{(r,r) \mid r \in R\}. \end{aligned}$$

In cases where confusion about the underlying kernel and closure systems might appear, we also write $\mathbb{G}_{\mathcal{K},\mathcal{C}}$ or $\mathbb{G}_{\mathbb{K},\mathbb{C}}$ for the shorter $\mathbb{G}$. The incidence relation of $\mathbb{G}$ will be denoted by $\boxplus$, i.e. $\mathbb{G} = (M \cup U, U \cup N, \boxplus)$.

Remark 1. It is well known that $\lceil X \rceil = X^{JJ}$ is the intersection of all columns from $\mathbb{C}$ that contain X. Dually, it holds that

$$\lfloor X \rfloor = X^{\mathtt{C}II\mathtt{C}} = \bigcup\{m^{I\mathtt{C}} \mid m \in M \text{ with } m^{I\mathtt{C}} \subseteq X\}.$$

Since the kernels are exactly the unions of complements of columns from $\mathbb{K}$, it holds that r is robust if and only if $U \setminus \{r\}$ is a column in both contexts $\mathbb{K}$ and $\mathbb{C}$. Furthermore, all rough set abstractions are of the form $(S^{I\mathtt{C}}, T^J)$ with $S \subseteq M$ and $T \subseteq N$. The equivalences

$\mathbb{K}$	a	b	c	d
1	×	×		×
2	×	×		×
3		×	×	
4	×	×		
5	×	×	×	×
6	×		×	×

$\mathbb{G}$	1	2	3	4	5	6	e	f	g	h	i
a	×	×		×	×	×		×	×		×
b	×	×	×	×	×					×	
c			×		×	×				×	×
d	×	×			×	×		×	×		×
1	×	×	×	×	×	×	×			×	×
2	×	×	×	×	×	×	×	×		×	×
3	×	×		×	×	×		×	×		×
4	×	×	×	×	×	×		×	×	×	×
5	×	×	×	×	×	×			×	×	×
6	×	×	×	×	×					×	

$\mathbb{C}$	e	f	g	h	i
1	×			×	×
2	×	×		×	×
3		×	×		×
4		×	×	×	×
5			×	×	×
6				×	

Fig. 2. An example for contexts $\mathbb{K}$, $\mathbb{C}$ and the corresponding $\mathbb{G} = \mathbb{G}_{\mathbb{K},\mathbb{C}}$

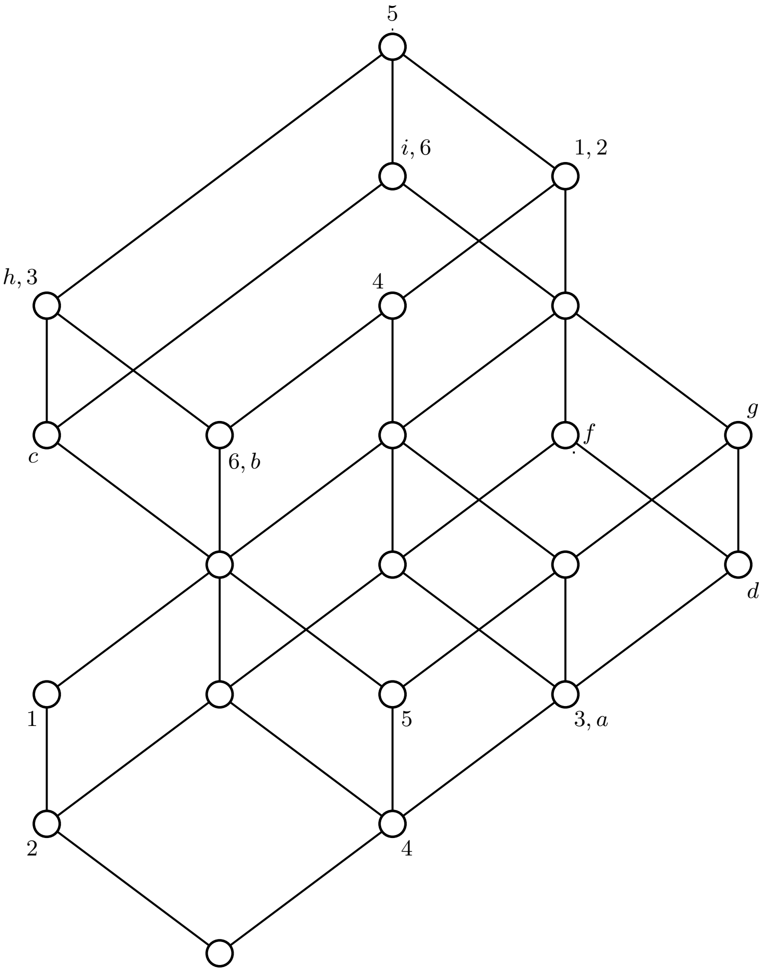


Fig. 3. The concept lattice $\underline{\mathfrak{B}}(\mathbb{G})$ to the context $\mathbb{G}$ from Figure 2

$$\begin{aligned}
S \times T \subseteq \perp \iff & \forall s \in S \forall t \in T : s^I \cup t^J = U \\
\iff & \forall s \in S \forall t \in T : s^{I\complement} \subseteq t^J \\
\iff & \bigcup_{s \in S} s^{I\complement} \subseteq \bigcap_{t \in T} t^J \\
\iff & S^{IC} \subseteq T^J.
\end{aligned}$$

show us that the *job* of $\perp$ is to *enforce* the first characteristic property of abstractions from Proposition 1. Similarly, $\overset{*}{\times}$ ensures that the second of these properties is getting fulfilled. The following Proposition 3 and the two subsequent corollaries demonstrate a possible way to show $\Gamma \cong \underline{\mathfrak{B}}(\mathbb{G})$ without using the notion of a P-Fusion. Proofs were omited since the principle result is known from [3].

Proposition 3. *$\mathbb{K}^d = (M, U, I^d)$ and $\mathbb{C} = (U, N, J)$ are both compatible subcontexts of $\mathbb{G}$.*

In the next corollary we use the following notation: for two sets X and Y we define $X_Y := X \cap Y$. This unusual notation just has the purpose of increasing the readability.

Corollary 4. *For $A \subseteq M \cup U$ and $B \subseteq U \cup N$ the following statements are equivalent:*

(a) $(A, B) \in \underline{\mathfrak{B}}(\mathbb{G})$,

(b) $(B_U, A_M) \in \underline{\mathfrak{B}}(\mathbb{K})$, $(A_U, B_N) \in \underline{\mathfrak{B}}(\mathbb{C})$, $B_U^{\mathsf{C}} \subseteq A_U$ *and* $(A_U \setminus B_U^{\mathsf{C}}) \cap R = \emptyset$.

Corollary 5. *The mapping $\varphi : \mathit{\Gamma}_{\mathcal{K},\mathcal{C}} \to \underline{\mathfrak{B}}(\mathbb{G})$ with $(K, C) \mapsto (C \cup K^{\mathsf{C}I}, K^{\mathsf{C}} \cup C^J)$ is an isomorphism.*

One can interpret the concepts (A, B) of $\mathbb{G}$ in the following way. Let $\mathcal{G}_{\text{in}} := \{m^{I\mathsf{C}} \mid m \in M\}$ and $\mathcal{G}_{\text{out}} := \{n^{J\mathsf{C}} \mid n \in N\}$ be the set of so-called *inner* and *outer granules* respectively (see [3]). Then A_U (the closure) is the complement of the union of all outer granules disjoint from A_U, which are exactly the granules $n^{J\mathsf{C}}$ with $n \in B_N$. Dually, B_U^{C} (the kernel) is the union of the $m^{I\mathsf{C}}$ with $m \in A_M$, which are exactly the inner granules contained in B_U^{C}. Hence, we are not only able to read the corresponding abstraction from (A, B), but the concept also tells us from which inner and outer granules the kernel and the closure were built.

In the case that R and $\perp$ both are empty the context $\mathbb{G}$ is of the form

$$\frac{\mathbb{K}^d \mid \emptyset}{\times \mid \mathbb{C}},$$

and hence $\mathit{\Gamma}$ is isomorphic to the *vertical sum* of $\mathcal{C} \cong \underline{\mathfrak{B}}(\mathbb{C})$ and $\mathcal{K} \cong \underline{\mathfrak{B}}(\mathbb{K}^d)$. Thus, in this case $\mathit{\Gamma}$ is slim in the sense of page 118. The following proposition shows how to read from the P-Fusion $\mathbb{G}$ if $\mathit{\Gamma}$ is slim, or not.

Proposition 4. *$\mathit{\Gamma}$ is slim if and only if for all $m \in M$ and $n \in N$ from $m \perp n$ it follows that $m^{I\mathsf{C}} \subseteq R$ or $n^{J\mathsf{C}} \subseteq R$ holds.*

Before proving this proposition we want to state one corollary.

Corollary 6. *If $\mathbb{K}$ and $\mathbb{C}$ are both attribute reduced[2], $\mathit{\Gamma}$ is slim if and only if for all $m \in M$ and $n \in N$ from $m \perp n$ it follows that $m^I = U \setminus \{x\}$ or $n^J = U \setminus \{x\}$ for some robust x. Thus, $R = \emptyset$ implies that $\mathit{\Gamma}$ is slim if and only if $\perp$ is empty.*

Proof (of Proposition 4). For $(m, n) \in \perp$ we put $A := m^I \cap n^J \cap R$. Hence, A is the set of all robust elements that are contained in the closure n^J but that are not

[2] This is (under reasonable finiteness conditions) no restriction to the choice of $\mathcal{K}$ and $\mathcal{C}$. It is just an restriction to the choice of the representing contexts $\mathbb{K}$ and $\mathbb{C}$.

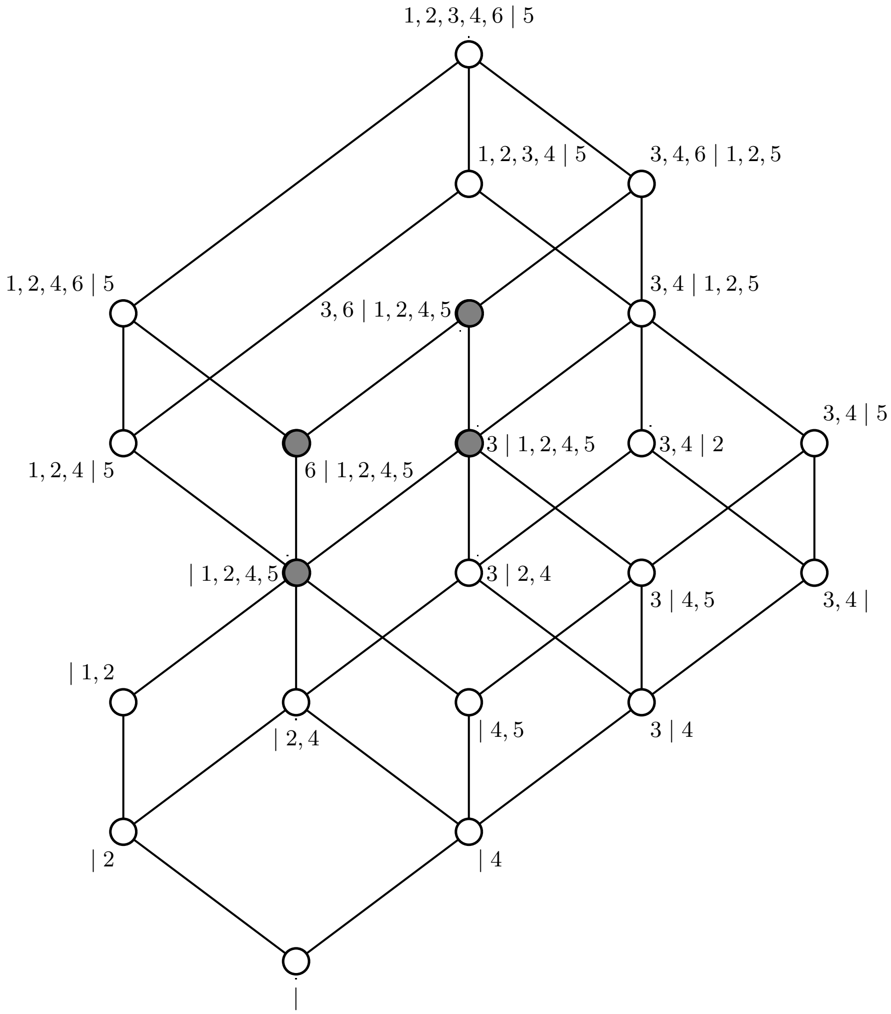


Fig. 4. The lattice $\Gamma_{\mathbb{K},\mathbb{C}}$ of rough set abstractions given through context $\mathbb{G}$ in Figure 2. In the diagram the node belonging to an abstraction (K, C) is denoted by $K \mid (H \setminus K)$ where set brackets and empty sets are omitted. Hence, we write just $\mid 2$ instead of $(\emptyset, \{2\})$ and $3, 4 \mid 5$ instead of $(\{3, 4\}, \{3, 4, 5\})$. The grey nodes correspond to the abstractions from the middle.

contained in the kernel $m^{I\complement}$. Furthermore it holds $m^{I\complement} \subseteq n^J$. Thus, $(m^{I\complement}, n^J \setminus A)$ is an abstraction. If Γ is slim, it holds that $n^J \setminus A = m^{I\complement} \cup R^{\complement}$ (iff $n^{J\complement} \cup A = m^I \cap R$) or $m^{I\complement} = (n^J \setminus A) \cap R$. Therefore, $n^{I\complement}$ or $n^{J\complement}$ is a subset of R.

Let (K, C) be an abstraction not belonging to the spine of Γ. Thus, the positive region K contains a non-robust element x and the negative region $C^{\complement}$ contains a non-robust element y. Since $K^{\complement}$ and C are extents from $\mathbb{K}$ and $\mathbb{C}$ respectively, there is a $m \in M$ and a $n \in N$ with $K^{\complement} \subseteq m^I \not\ni x$ and $C \subseteq n^J \not\ni y$.

Thus, $m^{I\mathtt{C}} \subseteq K \subseteq C \subseteq n^J$ holds which implies $m \perp n$. But it also holds that $x \in m^{I\mathtt{C}} \nsubseteq R$ and $y \in n^{J\mathtt{C}} \nsubseteq R$. □

We conclude this section with the remark that switching the roles of $\mathbb{K}$ and $\mathbb{C}$ yields dually isomorphic systems $\Gamma_{\mathbb{K},\mathbb{C}}$ and $\Gamma_{\mathbb{C},\mathbb{K}}$ of rough set abstractions. This is easy to see if one convinces oneself that this switching corresponds to transposing $\mathbb{G}$.

4 Set-Generated Abstractions

As already mentioned Γ might contain abstractions that are not generated by a subset of U. On the first sight this seems a kind of irritating. Hence, the conditions under which all abstractions are set-generated abstractions should be investigated. We are not able to give a complete characterization, but propose a sufficient condition instead.

Lemma 2. *Let (K, C) be an abstraction. If for every element x from the boundary $C \setminus K$ there is some $y \in U$ with $x \neq y$, $x^I = y^I$, and $x^J = y^J$, the abstraction (K, C) is set-generated.*

Proof. Let $B := C \setminus K$ be the boundary and let θ be the equivalence relation on U defined through $x\theta y$ iff $x^I = y^I$ and $x^J = y^J$. As explained in the introduction, we call a subset of U *θ-crisp* if it is the union of θ-equivalence classes. All closures and all kernels are θ-crisp. Hence, all regions of all abstractions are *θ-crisp*. Let S be a system of distinct representatives. Furthermore, let $S_B := S \cap B$ be the representatives from the equivalence classes contained in the boundary B. The premise implies that S_B does not contain a complete equivalence class. Thus, for every $m \in M$ $m^{I\mathtt{C}} \subseteq K$ holds if and only if $m^{I\mathtt{C}} \subseteq K \cup S_B$ holds. Hence, it follows

$$\lfloor K \cup S_B \rfloor \;=\; \bigcup \{m^{I\mathtt{C}} \mid m \in M \text{ with } m^{I\mathtt{C}} \subseteq K \cup S_B\} \;=\; \lfloor K \rfloor \;=\; K.$$

Analogously $\lceil K \cup S_B \rceil = H$ can be shown. Thus, $K \cup S_B$ generates (K, C). □

As a direct consequence we get the following. *Doubling* all objects of U that occur in boundaries induces an isomorphic lattice of abstractions in which every abstraction is set-generated. Let (O, A, Y) be a formal context and let $T \subseteq O$ be a collection of objects. We call the context $((O \setminus T) \cup (\{1, 2\} \times T), A, Y_T)$ the *T-doubled* context to (O, A, Y). Thereby, Y_T is defined via

$$gY_T m \;:\Longleftrightarrow\; \begin{cases} gYm, & \text{for } g \in O \setminus T, \\ xYm, & \text{for } g = (i, x) \in T. \end{cases}$$

Corollary 7. *Let $\bar{\mathbb{K}}$ and $\bar{\mathbb{C}}$ be the $R^{\mathtt{C}}$-doubled contexts to $\mathbb{K}$ and $\mathbb{C}$. It holds that the lattices $\Gamma_{\mathbb{K},\mathbb{C}}$ and $\bar{\Gamma} := \Gamma_{\bar{\mathbb{K}},\bar{\mathbb{C}}}$ are isomorphic. Furthermore, every abstraction from $\bar{\Gamma}$ is set-generated.*

We finish this section with the remark that the doubling of a robust element of course has an impact on the structure of Γ. The reason is: the robust element will not be robust any longer. As an example take $\mathbb{K} = \mathbb{C} = (U, U, \neq)$. The lattice of rough set abstractions is isomorphic to the powerset lattice of U, hence it is a direct product of chains having length one. After doubling all elements, the lattice is the direct product of chains having length two.

5 The Selfdual Case

In the special case that the two representing contexts $\mathbb{K}$ and $\mathbb{C}$ are equal it holds that a subset of U is a kernel if and only if its complement is a closure. If $\mathcal{K}$ and $\mathcal{C}$ fulfill this property we call it the **selfdual** case. It is equivalent to $\lfloor A \rfloor^{\mathsf{C}} = \lceil A^{\mathsf{C}} \rceil$ for every $A \subseteq U$. The selfdual case is well known from topology where it holds for the open and the closed sets. Furthermore, it includes a lot of applications involving the classical rough set setting. As we will see it allows us to define the *complement* of an abstraction. For the rest of this section we assume $\mathbb{K} = \mathbb{C}$.

5.1 Standard Complementation

The premise $\mathbb{K} = \mathbb{C}$ allows us to define $(K, C)^{\mathsf{C}} := (C^{\mathsf{C}}, K^{\mathsf{C}})$ to be the **standard complementation** (short: the **complement**) of $(K, C) \in \Gamma$. Obviously, applying the complement switches the positive and the negative region of an abstraction, whereas the boundary remains fixed. The statements (i) and (ii) of the following lemma tell us, that the standard complementation is an order involution on the self dual lattice $\Gamma = \Gamma_{\mathbb{K},\mathbb{K}}$.

Lemma 3. *For $\rho, \gamma \in \Gamma_{\mathbb{K},\mathbb{K}}$ the following statements hold:*

(i) $\rho^{\mathsf{CC}} = \rho$,

(ii) $\rho \sqsubseteq \gamma \;\Leftrightarrow\; \rho^{\mathsf{C}} \sqsupseteq \gamma^{\mathsf{C}}$,

(iii) $(\rho \vee \gamma)^{\mathsf{C}} = \rho^{\mathsf{C}} \wedge \gamma^{\mathsf{C}}$,

(iv) $(\rho \wedge \gamma)^{\mathsf{C}} = \rho^{\mathsf{C}} \vee \gamma^{\mathsf{C}}$,

(v) $\rho \wedge \rho^{\mathsf{C}} \sqsubseteq \gamma \vee \gamma^{\mathsf{C}}$.

Proof. A similar proposition can be found in [2]. The proof has been ommitted since it is elementary. □

Let us take a detailed look at statement (v). For every $(K, C) \in \Gamma$ and $B := C \backslash K$ it holds that

$$(K, C) \wedge (K, C)^{\mathsf{C}} = (\emptyset, B) \quad \text{and} \quad (K, C) \vee (K, C)^{\mathsf{C}} = (B^{\mathsf{C}}, U)$$

are the least and the greatest element of $\Gamma|_B$ (see page 2). Hence, $\rho \wedge \rho^{\mathsf{C}}$ is the abstraction having the same uncertain region as ρ (and as ρ^{C}) and having an empty positive region. Dually $\rho \vee \rho^{\mathsf{C}}$ is the abstraction having empty negative region but the same boundary as ρ.

Corollary 8. *Under the premise $\emptyset \in \mathcal{C}$ the following statements are equivalent:*

(a) every subset of U is a kernel and a closure,

(b) $R = U$,

(c) $K = C$ for all $(K, C) \in \Gamma$,

(d) the law of the excluded middle holds, i.e. $\rho \vee \rho^{\mathtt{C}} = 1$ for every $\rho \in \Gamma$,

(e) the non-contradiction law holds, i.e. $\rho \wedge \rho^{\mathtt{C}} = 0$ for every $\rho \in \Gamma$.

Thereby $0 := (\emptyset, \emptyset)$ and $1 := (U, U)$ denote the least and the greatest rough set abstraction from Γ.

Proof. The first three statements are always equivalent (for arbitrary $\mathcal{K}$ and $\mathcal{C}$). The rest directly follows from the argumentation above. □

$\approx$	1	2	3	4	5
1	×	×			
2	×	×	×		
3		×	×	×	
4			×	×	×
5				×	×

Fig. 5. A context $(U, U, \approx)$ where $\approx$ is a tolerance relation on $U = \{1, \ldots, 5\}$

5.2 Tolerance Indiscernibility

In [2] Cattaneo and Ciucci among other things presented generalized rough sets starting with so-called *preclusive spaces.* They call a pair $(U, \#)$ a *preclusive space* if $\#$ is an irreflexive, symmetric relation on U. Obviously, preclusive relations are exactly the complements of tolerance relations. Furthermore, the authors of [2] studied the approximations of the form[3] $(X^{\mathtt{C}\#\#\mathtt{C}}, X^{\#\#})$ where $\cdot^{\#}$ is the derivation operator of the formal context $(U, U, \#)$. They investigated the structures formed by these pairs as so-called *quasi-BZ-distributive lattices.* However, as we shall see in Example 1, these pairs in general do not form a lattice. And even if they do, this lattice need not be distributive. This problem does not occur in [3] since Γ is the lattice that is *generated by* these pairs.

The mentioned structures from [2] are basically bounded lattices with two additional unary operations, playing the role of negations. The first one matches our standard complementation. For the second one we will have to specialize our settings to $\mathbb{K} = \mathbb{C} = (U, U, \not\approx)$, where $\not\approx$ is the complement of a tolerance relation $\approx$ on U. As we have seen above, the standard complement does *not* fulfill the non-contradiction law $\rho \wedge \rho^{\mathtt{C}} = 0$. Under these circumstances the *canonical*

[3] Strictly speaking they took the pairs $(X^{\mathtt{C}\#\#\mathtt{C}}, X^{\#\#\mathtt{C}})$, but the second component was ordered by $\supseteq$, what makes this approach similar to that from [3].

choice of a meaningful self mapping fulfilling non-contradiction law is the so-called **Brouwer complement** (see [2])

$$(K,C)^{\not\approx} := (\lfloor C^{\not\approx} \rfloor, C^{\not\approx}) = (C^{\not\approx\mathsf{C}\not\approx\not\approx\mathsf{C}}, C^{\not\approx})$$

(for $(K,C) \in \mathit{\Gamma}$). Obviously, $(K,C)^{\not\approx}$ is the greatest abstraction having $C^{\not\approx}$ as the nonpositive region.

Lemma 4 ([2]). *For $\rho, \gamma \in \mathit{\Gamma}$ the following statements hold:*

(i) $\rho \sqsubseteq \rho^{\not\approx\not\approx}$,

(ii) $(\rho \vee \gamma)^{\not\approx} = \rho^{\not\approx} \wedge \gamma^{\not\approx}$,

(iii) $\rho \wedge \rho^{\not\approx} = 0 := (\emptyset, \emptyset)$, *and*

(iv) $\rho^{\not\approx} \sqsubseteq \rho^{\mathsf{C}}$.

Example 1. We examine the tolerance relation $\approx$ given in Figure 5. The underlying universe is $U = \{1, \ldots, 5\}$. The corresponding lattice of rough set abstractions $\mathit{\Gamma}_{\mathbb{K},\mathbb{K}}$ with $\mathbb{K} = (U, U, \not\approx)$ is displayed in Figure 6. The first thing we want to point out is that not all abstractions are set-generated. Take for instance $\rho := (\{1,2\}, \{1,2,3\})$. If there was a subset generating ρ, it has to be $\{1,2\}$ or $\{1,2,3\}$. But both of these are closed and open. Hence $\lambda_1 := (\{1,2\}, \{1,2\})$ and $\omega_1 := (\{1,2,3\}, \{1,2,3\})$ are a lower and an upper neighbor of ρ. As Figure 6 shows $\lambda_2 := (\emptyset, \{1,2,3\})$ is another lower and $\omega_2 := (\{1,2\}, U)$ is another upper neighbor of ρ. They are both generated by subsets of U (for instance by $\{1,3\}$ and by $\{1,2,4\}$ respectively).

We now take a look at the partially ordered set of the set-generated abstractions $(\lfloor X \rfloor, \lceil X \rceil)$ with $X \subseteq U$. We see that it does not form a lattice, since $\{\omega_1, \omega_2\}$ is the set of minimal upper bounds of $\{\lambda_1, \lambda_2\}$. Hence there is no supremum of λ_1 and λ_2. In this example the set R of robust elements is empty. Hence, $(\emptyset, U)$ is the only element from the middle. Furthermore, $R = \emptyset$ makes it easier to identify the abstractions from the spine. These are the ones that have an empty positive or an empty negative region. Thus, we can read from Figure 6 that λ_1, ρ and ω_1 and their three complements are the only abstractions not belonging to the spine.

In the classical setting of Rough Set Theory, i.e. when $\approx$ is an equivalence relation, it holds that $(K,C)^{\not\approx} = (C^{\mathsf{C}}, C^{\mathsf{C}})$. Thus, it holds that $(K,C)^{\not\approx}$ is the largest abstraction having C^{C} as the nonnegative region. Hence, it follows the well known fact that in the classical setting $(K,C)^{\not\approx}$ is the pseudocomplement[4] of (K,C). Unfortunately this does not hold for arbitrary tolerance relations $\approx$, as one can see in Example 1.

[4] See [1]. Let a be an element of a bounded lattice L. The element a^* is called the **pseudocomplement** of a if for all $b \in L$

$$a \wedge b = 0 \iff b \leq a^*$$

holds.

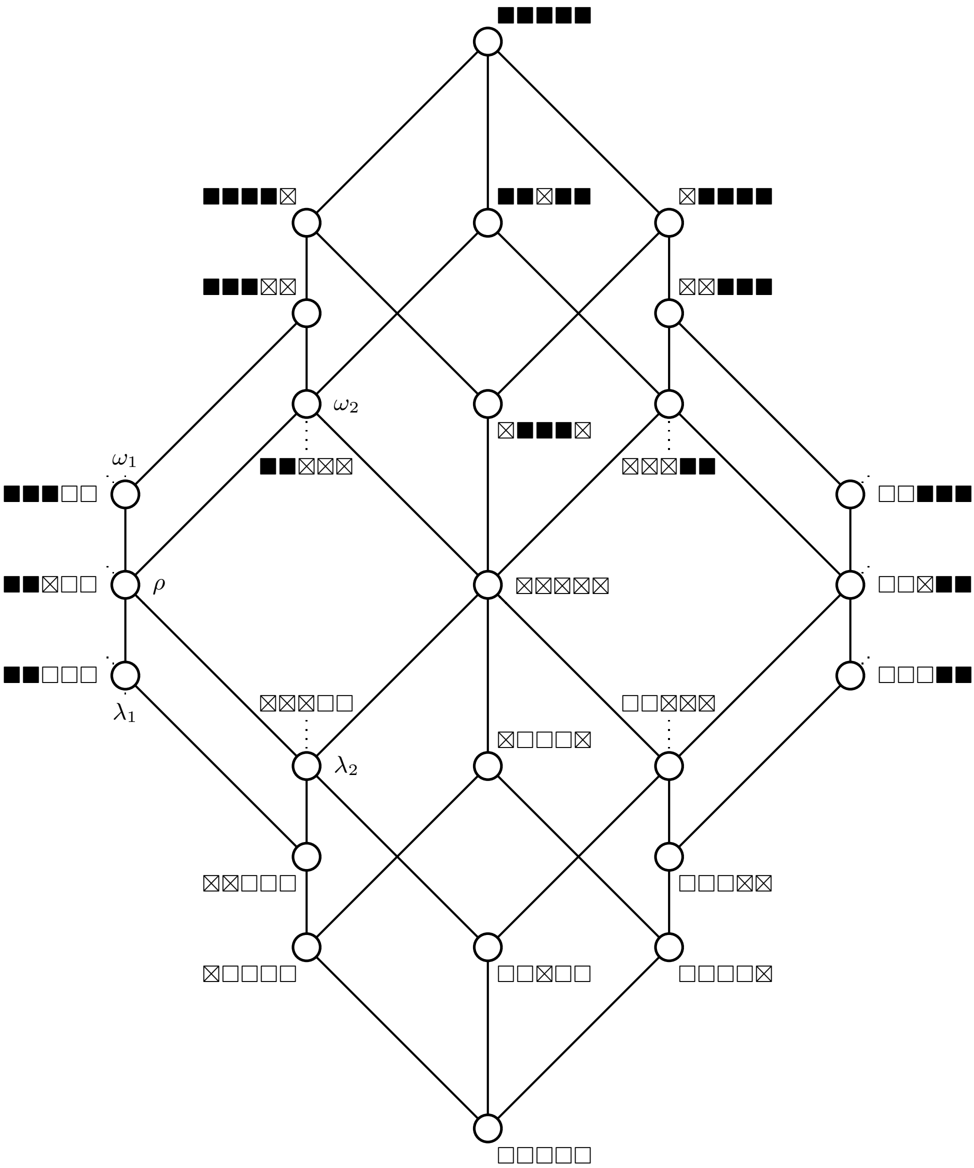


Fig. 6. The lattice $\Gamma_{\mathbb{K},\mathbb{K}}$ of rough set abstractions with $\mathbb{K} = (U, U, \not\approx)$ where $\approx$ is the tolerance relation from Figure 5. The notations are as follows. For the elements $1, 2, 3, 4, 5$ from U we write five little boxes. If i is an element of the positive/uncertain/negative region, the i-th box is ■/☒/□. For instance □□☒■■ denotes the rough set abstraction $(\{4, 5\}, \{3, 4, 5\})$. The abstractions λ_1, λ_2, ω_1 and ω_2 are set-generated, ρ is not.

6 Conclusion

Our investigations provided a clear characterization of the rough set abstractions from [3]. Furthermore, we believe this article improved the understanding of the robust elements and the lattices of abstractions. Last but not least we proposed suitable negation operators.

Future investigations could handle the following problems. What are the conditions under which the P-fusion from [3] is a tensor product? And related to this question: What are the conditions under which Γ is isomorphic to the order relation of a complete lattice? What role plays the distributivity for answering the two previous questions? Furthermore, our experience showed us that the abstractions which are not set-generated are pretty rare, even when $\mathbb{K}$ and $\mathbb{C}$ are clarified. Thus, our sufficient condition from Corollary 7 is far away from being a necessary one. Hence, there still is the open question of charaterizing the cases where all abstractions are set-generated.

References

1. Davey, B.A., Priestley, H.A.: Introduction to Lattices and Order, 2nd edn. Cambridge University Press, Cambridge (2002)
2. Cattaneo, G., Ciucci, D.: Algebraic Structures for Rough Sets. In: Peters, J.F., Skowron, A., Dubois, D., Grzymała-Busse, J.W., Inuiguchi, M., Polkowski, L. (eds.) Transactions on Rough Sets II. LNCS, vol. 3135, pp. 208–252. Springer, Heidelberg (2004)
3. Ganter, B.: Lattices of Rough Set Abstractions as P-Products. In: Medina, R., Obiedkov, S. (eds.) ICFCA 2008. LNCS (LNAI), vol. 4933, pp. 199–216. Springer, Heidelberg (2008)
4. Ganter, B., Wille, R.: Formal Concept Analysis – Mathematical Foundations. Springer, Heidelberg (1999)
5. Järvinen, J.: Lattice Theory for Rough Sets. In: Peters, J.F., Skowron, A., Düntsch, I., Grzymała-Busse, J.W., Orłowska, E., Polkowski, L. (eds.) Transactions on Rough Sets VI. LNCS, vol. 4374, pp. 400–498. Springer, Heidelberg (2007)
6. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning About Data. Kluwer Academic Publishers, Dordrecht (1991)

Some Computational Problems Related to Pseudo-intents

Barış Sertkaya[*]

TU Dresden, Germany
sertkaya@tcs.inf.tu-dresden.de

Abstract. We investigate the computational complexity of several decision, enumeration and counting problems related to pseudo-intents. We show that given a formal context and a subset of its set of pseudo-intents, checking whether this context has an additional pseudo-intent is in coNP, and it is at least as hard as checking whether a given simple hypergraph is not saturated. We also show that recognizing the set of pseudo-intents is also in coNP, and it is at least as hard as identifying the minimal transversals of a given hypergraph. Moreover, we show that if any of these two problems turns out to be coNP-hard, then unless P = NP, pseudo-intents cannot be enumerated in output polynomial time. We also investigate the complexity of finding subsets of a given Duquenne-Guigues Base from which a given implication follows. We show that checking the existence of such a subset within a specified cardinality bound is NP-complete, and counting all such minimal subsets is #P-complete.

1 Introduction

Pseudo-intents play an important rôle in Formal Concept Analysis (FCA) [8]. They form the premises of the Duquenne-Guigues Base [10], which is a minimum cardinality base for the set of implications that hold in a formal context. Computational complexity of problems related to pseudo-intents have been of major interest to the FCA community since their introduction.

One central computational problem related to pseudo-intents is determining whether a given set is a pseudo-intent of a given formal context. It has been shown in [15,16] that this problem is in coNP. However, the lower complexity bound for this problem is still open. One other natural problem is enumerating the pseudo-intents of a given formal context. The most well-known algorithm for this purpose is the *next-closure* algorithm [7]. Recently, an algorithm that computes the pseudo-intents by processing a single attribute at a single step, namely *attribute-incremental* algorithm, has been introduced in [18]. In [19], an algorithm for checking whether a set is pseudo-intent, has been presented. Another problem related to pseudo-intents is given a formal context, determining the number of its pseudo-intents. In [14], it has been shown that this counting problem is #P-hard. In addition to this, there it has also been shown that the

[*] Supported by the German Research Foundation (DFG) under grant BA 1122/12-1.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 130–145, 2009.

number of pseudo-intents of a formal context can be exponential in the size of the incidence relation of this formal context. Given this fact, it is clearly not possible to enumerate all pseudo-intents of a formal context in time polynomial in the size of this context. In complexity theory, for analyzing the performance of enumeration algorithms where the number of solutions can be exponential in the size of the input, one considers other measures. One such measure is to take into account not only the size of the input, but also the size of the output. An algorithm is said to run in *output polynomial* time [13] if it enumerates the solutions in time polynomial in the size of the input *and the output*. One advantage of an output polynomial algorithm is that it runs in polynomial time (in the size of the input) when there are only polynomially many solutions.

In the present work we investigate whether pseudo-intents can be enumerated in output polynomial time. We start with the observation that next-closure and attribute-incremental algorithms do not run in output polynomial time since their running times depend not only on the number of pseudo-intents, but also on the number of intents. We formulate two decision problems that are of significant importance for the existence of an output polynomial time algorithm. In Section 3 we work on the first problem, which is given a formal context $\mathbb{K}$ and a subset $\mathcal{P}$ of its set of pseudo-intents, the problem of checking whether $\mathbb{K}$ has an additional pseudo-intent, i.e., a pseudo-intent that does not already appear in $\mathcal{P}$. We show that this problem is in coNP, and it is at least as hard as the complement problem of checking whether a given simple hypergraph is saturated [3], which is a prominent open problem in hypergraph theory [2]. In Section 4 we work on the second problem, which is given a formal context $\mathbb{K}$ and a set $\mathcal{P}$ of subsets of its attribute set, the problem of checking whether $\mathcal{P}$ is precisely the set of pseudo-intents of $\mathbb{K}$. We show that this problem is also in coNP, and it is at least as hard as identifying the minimal transversals of a given hypergraph [3], which is also an open problem. Moreover, we show that if any of these two problems turns out to be coNP-hard, then unless P = NP, pseudo-intents cannot be enumerated in output polynomial time. In Section 5 we investigate the complexity of finding subsets of a given Duquenne-Guigues Base from which a given implication follows. We show that checking the existence of such a subset within a specified cardinality bound is NP-complete, and counting all such minimal subsets is #P-complete.

2 Preliminaries

We briefly introduce basic notions of Formal Concept Analysis [8]. Given a *formal context* $\mathbb{K} = (G, M, I)$ with the *derivation operator* $(\cdot)'$, and an *implication* $P \rightarrow Q$, where $P, Q \subseteq M$, we say that $P \rightarrow Q$ *holds* in $\mathbb{K}$ if the *objects* that have the *attributes* in P also have the attributes in Q, i.e., $P' \subseteq Q'$. A set $A \subseteq M$ *respects* an implication $P \rightarrow Q$ if $P \nsubseteq A$ or $Q \subseteq A$. An implication $P \rightarrow Q$ *follows semantically* from a set of implications $\mathcal{L}$ (written $\mathcal{L} \models P \rightarrow Q$) if each subset of M respecting the implications in $\mathcal{L}$ also respects $P \rightarrow Q$. We denote the *implicational theory* of $\mathcal{L}$, i.e, the set of all implications that follow from $\mathcal{L}$, with $Imp(\mathcal{L})$.

In [10], a minimum cardinality base, which is called the *Duquenne-Guigues Base*, of a given formal context has been characterized, and it has been shown that there cannot be another base with fewer implications. The premises of the implications in a Duquenne-Guigues Base are called the pseudo-intents of the underlying formal context. A set $P \subseteq M$ is a *pseudo-intent* if $P \neq P''$ and $Q'' \subsetneq P$ holds for every pseudo-intent $Q \subsetneq P$. Equivalently, a set $P \subseteq M$ is a pseudo-intent if $P \neq P''$, it is a quasi-intent, and for every quasi-intent $Q \subsetneq P$, $Q'' \subsetneq P$ holds, where a *quasi-intent* is defined as a set $Q \subseteq M$ that satisfies $R'' \subseteq Q$ or $R'' = Q''$ for any $R \subseteq Q$.

2.1 Hypergraphs and Related Problems

A *hypergraph* [2] $\mathcal{H} = (V, \mathcal{E})$ consists of a set of *vertices* $V = \{v_i \mid 1 \leq i \leq n\}$, and a set of nonempty *(hyper)edges* $\mathcal{E} = \{E_j \mid 1 \leq j \leq m\}$ where $E_j \subseteq V$. A set $W \subseteq V$ is called a *transversal* of $\mathcal{H}$ if it intersects all edges of $\mathcal{H}$, i.e., $\forall E \in \mathcal{E}.\ E \cap W \neq \emptyset$. A transversal is called *minimal* if no proper subset of it is a transversal. The set of all minimal transversals of $\mathcal{H}$ constitute another hypergraph on V called the *transversal hypergraph* of $\mathcal{H}$, which is denoted by $Tr(\mathcal{H})$. Generating $Tr(\mathcal{H})$ is an important problem which has applications in many fields of computer science. It is defined as follows:

Problem: TRANSVERSAL ENUMERATION (TRANS-ENUM)
Input: A hypergraph $\mathcal{H} = (V, \mathcal{E})$ on a finite set V.
Output: The edges of the transversal hypergraph $Tr(\mathcal{H})$.

The well-known decision problem associated to this computation problem is defined as follows:

Problem: TRANSVERSAL HYPERGRAPH (TRANS-HYP)
Input: Two hypergraphs $\mathcal{H} = (V, \mathcal{E}_{\mathcal{H}})$ and $\mathcal{G} = (V, \mathcal{E}_{\mathcal{G}})$.
Question: Is $\mathcal{G}$ the transversal hypergraph of $\mathcal{H}$, i.e., does $Tr(\mathcal{H}) = \mathcal{G}$ hold?

Computational complexity of these problems have now been extensively studied [3,5,6] and many important applications of these problems have been identified in logic and artificial intelligence [4], databases [17] and data mining [11]. TRANS-HYP is known to be in coNP, but so far neither a polynomial time algorithm has been found, nor has it been proved to be coNP-complete. Similarly, it is an open problem whether TRANS-ENUM can be solved in output polynomial time. We say that a decision problem Π is TRANS-HYP-hard if TRANS-HYP can be reduced to Π by a standard polynomial transformation. We say that Π is TRANS-HYP-complete if it is TRANS-HYP-hard and Π can be reduced to TRANS-HYP by a polynomial transformation.

3 Complexity of Enumerating Pseudo-intents

For enumerating pseudo-intents, unfortunately no output polynomial algorithm is known currently. The most well-known algorithm *next-closure* [7] for enumerating the pseudo-intents always enumerates the concept intents as well, i.e, its running time depends not only on the number of pseudo-intents but also on the number of concept intents. Since the number of concept intents can be exponential in the number of pseudo-intents, this algorithm in general does not run in output polynomial time. Similarly, the *attribute-incremental algorithm* in [18] has also time complexity depending on both the number of pseudo-intents and the number of concept intents. In the light of our current knowledge, it is not even clear whether there can be an algorithm at all that enumerates pseudo-intents in output polynomial time. In order to investigate this further, let us first formally define the problem.

Problem: PSEUDO-INTENT ENUMERATION (PIE)
Input: A formal context $\mathbb{K}$.
Output: The set of pseudo-intents of $\mathbb{K}$.

For solving this enumeration problem, the following decision problem has crucial importance:

Problem: ADDITIONAL PSEUDO-INTENT (API)
Input: A formal context $\mathbb{K} = (G, M, I)$, and a set $\mathcal{P}$ of pseudo-intents of $\mathbb{K}$, i.e., $\mathcal{P} \subseteq \{P \mid P \subseteq M,\ P \textit{ pseudo-intent of } \mathbb{K}\}$.
Question: Is there an additional pseudo-intent, i.e., $Q \subseteq M$ s.t. Q is a pseudo-intent of $\mathbb{K}$ and $Q \notin \mathcal{P}$?

Because, as Proposition 1 below shows, if this problem cannot be decided in polynomial time, then unless P = NP, PIE cannot be solved in output polynomial time.

Proposition 1. *If* API *cannot be decided in polynomial time, then unless* P = NP*,* PIE *cannot be solved in output-polynomial time.*

Proof. Assume that we have an algorithm $\mathcal{A}$ that solves PIE in output-polynomial time. Let its runtime be bounded by a polynomial $p(IS, OS)$ where IS denotes the size of the input context and OS denotes the size of the output, i.e., the set of all pseudo-intents of the input context.

In order to decide API for an instance given by the context $\mathbb{K}$ and a set $\mathcal{P}$ of pseudo-intents of $\mathbb{K}$, we construct another algorithm $\mathcal{A}'$ that works as follows: It runs $\mathcal{A}$ on $\mathbb{K}$ for at most $p(|\mathbb{K}|, |\mathcal{P}|)$-many steps. If $\mathcal{A}$ terminates within $p(|\mathbb{K}|, |\mathcal{P}|)$-many steps, it means that $\mathcal{P}$ contains all pseudo-intents of $\mathbb{K}$, i.e., there is no additional pseudo-intent. So $\mathcal{A}'$ returns *no*. If $\mathcal{A}$ does not terminate after $p(|\mathbb{K}|, |\mathcal{P}|)$-many steps, this implies that there is an additional pseudo-intent that is not contained in $\mathcal{P}$, so $\mathcal{A}'$ returns *yes*. It is easy to see that the runtime of $\mathcal{A}'$ is bounded by a polynomial in $|\mathbb{K}|$ and $|\mathcal{P}|$, that is $\mathcal{A}'$ decides API in time polynomial in the size of the input. □

The proposition shows that determining the complexity of API is indeed crucial for determining the complexity of PIE. In the following we show that API is in coNP, and it is at least as hard as the complement of a prominent open problem on hypergraphs. However, whether API is coNP-hard remains unfortunately open.

Proposition 2. API *is in co*NP.

Proof. Given an instance of API with the input $\mathbb{K}$ and $\mathcal{P}$, construct the set of implications $\mathcal{L} = \{P \to P'' \mid P \in \mathcal{P}\}$ and nondeterministically guess a set $Q \subseteq M$. We can verify in polynomial time that $Q \to Q''$ *does not* follow from $\mathcal{L}$, i.e., the complement of the problem is in NP, thus API is in coNP. □

Before we can continue with the proof of lower bound, we need to introduce some more notions from hypergraphs. A hypergraph $\mathcal{H} = (V, \mathcal{E})$ is called *saturated* [3] if every subset of V is contained in at least one of the edges of $\mathcal{H}$, or it contains at least one edge of $\mathcal{H}$, i.e., for every $W \subseteq V$, $W \subseteq E$ holds, or $E \subseteq W$ holds for some $E \in \mathcal{E}$. It has been shown in [3] that checking whether a hypergraph is saturated is coNP-complete. There, a special case of the problem where the given hypergraph is restricted to be simple, has also been considered. A hypergraph is called *simple* if no edge contains another edge.

Problem: SIMPLE HYPERGRAPH SATURATION (SIMPLE-H-SAT)
Input: A simple hypergraph $\mathcal{H} = (V, \mathcal{E})$, i.e., $\forall E, E' \in \mathcal{E}. E \subseteq E' \Rightarrow E = E'$.
Question: Is $\mathcal{H}$ saturated, i.e., is it true that for every $W \subseteq V$, $W \subseteq E$ holds or $E \subseteq W$ holds for some $E \in \mathcal{E}$?

It is not difficult to see that this problem is in coNP. However, up to now there has neither been a proof that it is coNP-hard, nor a proof that it is in P. It has been shown in [3] that this problem is under polynomial transformations computationally equivalent to TRANS-HYP, which as mentioned before is a prominent open problem in hypergraph theory. In the following we show that our problem API is at least as hard as the complement of SIMPLE-H-SAT:

Theorem 1. API *is co*SIMPLE-H-SAT-*hard.*

Proof. Let an instance of SIMPLE-H-SAT be given with the simple hypergraph $\mathcal{H} = (V, \mathcal{E})$ where $\mathcal{E} = \{E_1, \ldots, E_n\}$. From $\mathcal{H}$ we construct the formal context $\mathbb{K}_{\mathcal{H}} = (G, M, I)$ where $M = V$, and G and I are defined as follows: For every E_i, $1 \leq i \leq n$, we create the following objects: For every $D \subsetneq E_i$ such that $|D| = |E_i| - 1$, we create an object with the intent D. E_i has $|E_i|$-many such subsets. We name these objects as g_{ij} where $1 \leq i \leq n$ and $1 \leq j \leq |E_i|$. In total, G contains $\sum_{i=1}^{n} |E_i|$ objects. We construct $\mathcal{P}$ by just taking the edges of $\mathcal{H}$, i.e, $\mathcal{P} = \{E_1, \ldots, E_n\}$. Obviously, both $\mathbb{K}_{\mathcal{H}}$ and $\mathcal{P}$ can be constructed in time polynomial in the size of $\mathcal{H}$.

Note that $\mathbb{K}_{\mathcal{H}}$ has the following property: Since $\mathcal{H}$ is simple, no edge is contained in another edge, and obviously not in strict subsets of any other edge. Then, for every i such that $1 \leq i \leq n$, $E_i' = \emptyset$ and $E_i'' = M$. That is E_i is not closed. Moreover all its strict subsets are closed. Because for every $D \subsetneq E_i$ either there is an object whose intent is D, or there is a set of objects such that the intersection of their intents is D. This is due to the objects g_{ij}, where $1 \leq j \leq |E_i|$, whose intents are strict subsets of E_i with cardinality $|E_i| - 1$. Thus, the edges E_i are pseudo-intents of $\mathbb{K}_{\mathcal{H}}$, which means that $\mathbb{K}_{\mathcal{H}}$ and $\mathcal{P}$ indeed form an instance of API. We claim that $\mathcal{H}$ is *not* saturated if and only if $\mathbb{K}_{\mathcal{H}}$ has an additional pseudo-intent.

($\Rightarrow$) Assume $\mathcal{H}$ is not saturated. Then, there exists a $W \subseteq V$ such that for every i such that $1 \leq i \leq n$, $W \not\subseteq E_i$ holds and $E_i \not\subseteq W$ holds. Assume without loss of generality that W is minimal with respect to property $W \not\subseteq E_i$ for every $1 \leq i \leq n$. Since W is not contained in any E_i, and obviously not contained in any strict subset of any E_i, $W' = \emptyset$ and $W'' = M$. That is W is not closed. Take any $X \subsetneq W$. Since W is minimal, $X \subseteq E_i$ holds for some $1 \leq i \leq n$. We know that $E_i \not\subseteq W$, then $X = E_i$ cannot hold, thus X satisfies $X \subsetneq E_i$. Since all strict subsets of E_i are closed, X is closed. We have shown that W is not closed but all its strict subsets are closed, thus W is a pseudo-intent. Moreover, it is an additional pseudo-intent since $W \neq E_j$, for all $1 \leq j \leq n$.

($\Leftarrow$) Assume $\mathbb{K}_{\mathcal{H}}$ has an additional pseudo-intent, i.e., a pseudo-intent Q such that $Q \neq E_i$ for every $1 \leq i \leq n$. Since strict subsets of E_i are closed, Q cannot be a strict subset of any E_i. Thus $Q \not\subseteq E_i$ for every $1 \leq i \leq n$. Moreover, by definition Q contains the closure of strictly smaller pseudo-intents. We know that for every $1 \leq i \leq n$, E_i is a pseudo-intent, and $E_i'' = M$. Since Q does not strictly contain M, it cannot strictly contain any E_i either. Together with $Q \neq E_i$, this implies that $E_i \not\subseteq Q$. We have shown that there exists a $Q \subseteq V$ such that $Q \not\subseteq E_i$ and $E_i \not\subseteq Q$ for every $1 \leq i \leq n$, thus $\mathcal{H}$ is not saturated. □

The following is an immediate consequence of Theorem 1 above and Theorem 4.12 in [3]:

Corollary 1. API *is co*TRANS-HYP*-hard.*

Theorem 1 has some interesting consequences. The formal context we have constructed in the proof has a special property; namely, subsets of object intents are closed in this formal context. The proof suggests that for the formal contexts of this form, the problem API and the complement problem of SIMPLE-H-SAT are computationally equivalent problems, i.e., API is coSIMPLE-H-SAT-complete. For such formal contexts, in addition to the reduction given in the proof, one can also easily reduce API to the complement of SIMPLE-H-SAT, i.e, take an instance of API given with such a context and a set of pseudo-intents of this context, construct an instance of SIMPLE-H-SAT and show that there is an additional pseudo-intent if and only if the constructed simple hypergraph is not saturated. It would definitely be interesting to investigate whether formal contexts of this form are natural in some application domains.

One other point that should be noted here is that SIMPLE-H-SAT lies at the boundary of intractability. As mentioned before, for arbitrary graphs it is coNP-complete [3]. The proof of Theorem 1 depends on the fact that the given hypergraph is simple. Whether this restriction can be eliminated and thus the intractability result carries over to API for arbitrary formal contexts, is definitely an interesting question that should be investigated.

4 Complexity of Recognizing the Set of Pseudo-intents

Next we consider another problem about pseudo-intents, namely recognizing the set of pseudo-intents. More precisely, given a formal context $\mathbb{K} = (G, M, I)$ and a set $\mathcal{P} \subseteq \mathscr{P}(M)$ it is the problem of deciding whether $\mathcal{P}$ is precisely the set of pseudo-intents of $\mathbb{K}$. Clearly, this problem can also be formulated as: Given a formal context $\mathbb{K}$ and a set of implications $\mathcal{L}$, decide whether $\mathcal{L}$ is the Duquenne-Guigues Base of $\mathbb{K}$. In the following we are going to investigate its computational complexity. We start with defining the problem formally:

Problem: PSEUDO-INTENTS (PIS)
Input: A formal context $\mathbb{K} = (G, M, I)$, and a set $\mathcal{P} \subseteq \mathscr{P}(M)$.
Question: Is $\mathcal{P}$ precisely the set of pseudo-intents of $\mathbb{K}$?

The following proposition shows that like computational complexity of API, the complexity of PIS has also crucial importance for the solvability of PIE in output polynomial time.

Proposition 3. *If* PIS *cannot be decided in polynomial time, then unless* P = NP*,* PIE *cannot be solved in output-polynomial time.*

Proof. The proof is almost the same as the proof of Proposition 1. Again we assume that we have an algorithm $\mathcal{A}$ that solves PIE in output-polynomial time and construct another algorithm $\mathcal{A}'$ that runs $\mathcal{A}$ for at most $p(|\mathbb{K}|, |\mathcal{P}|)$-many steps. The only difference is that, if $\mathcal{A}$ terminates within $p(|\mathbb{K}|, |\mathcal{P}|)$-many steps, then $\mathcal{A}'$ first compares the output of $\mathcal{A}$ with $\mathcal{P}$ and then returns *yes* if and only if they are equal. If they are not equal, or if $\mathcal{A}$ has not yet terminated, then $\mathcal{A}'$ returns *no*. Thus if PIE can be solved in output polynomial time, PIS can be decided in polynomial time. □

In the following we show that just like in the case of API, PIS is also in coNP, and it is at least as hard as TRANS-HYP. However, whether PIS is polynomial, or it is coNP-hard also remains open.

Proposition 4. PIS *is in co*NP*.*

Proof. Given an instance with the input $\mathbb{K} = (G, M, I)$ and $\mathcal{P}$, an algorithm that decides PIS for this instance first checks whether the elements of $\mathcal{P}$ are pseudo-intents of $\mathbb{K}$. If it encounters an element that is not a pseudo-intent, it terminates and returns *no*. If every $P \in \mathcal{P}$ is a pseudo-intent, then it continues with the second step. This step is the same as the algorithm in the proof of Proposition 2. The algorithm constructs the set of implications $\mathcal{L} = \{P \to P'' | P \in \mathcal{P}\}$ and non-deterministically guesses a set $Q \subseteq M$. Obviously the implication $Q \to Q''$ holds in $\mathbb{K}$, thus if $\mathcal{L}$ is a base for $\mathbb{K}$ then $Q \to Q''$ follows from $\mathcal{L}$. Then the algorithm verifies that this is *not* the case.

It is not difficult to see that this is a coNP algorithm. In the first step the algorithm performs polynomially-many checks each of which can be done in coNP by using the algorithm in [15]. In the second step the algorithm non-deterministically guesses a Q and in polynomial time verifies that $Q \to Q''$ does *not* follow from $\mathcal{L}$, which means that $\mathcal{L}$ is *not* a base, which implies that $\mathcal{P}$ is *not* the set of all pseudo-intents of $\mathbb{K}$. This step can be performed in coNP as well, thus the whole algorithm is a coNP algorithm. □

Theorem 2. PIS *is* TRANS-HYP*-hard.*

Proof. Let an instance of TRANS-HYP be given by the hypergraphs $\mathcal{H} = (V, \mathcal{E}_\mathcal{H})$ and $\mathcal{G} = (V, \mathcal{E}_\mathcal{G})$, where $\mathcal{E}_\mathcal{H} = \{h_i \mid 1 \leq i \leq n\}$ and $\mathcal{E}_\mathcal{G} = \{g_i \mid 1 \leq i \leq m\}$. From $\mathcal{H}$ we construct the context $\mathbb{K}_\mathcal{H} = (G, M, I)$ where $M = V$, and G and I are defined as follows: For every edge $h_i \in \mathcal{E}_\mathcal{H}$, create an object whose intent is the complement of h_i, i.e., $M \setminus h_i$. Let us denote this with $\overline{h_i}$. Moreover, for each set $f \subsetneq \overline{h_i}$ such that $|f| = |\overline{h_i}| - 1$, create an object with the intent f. $\overline{h_i}$ has $|\overline{h_i}|$-many such subsets. That is, for every edge $h_i \in \mathcal{E}_\mathcal{H}$ we create $|\overline{h_i}| + 1$ objects, which means that $\mathbb{K}_\mathcal{H}$ contains $\sum_{i=1}^{n} |\overline{h_i}| + n$ objects in total. From $\mathcal{G}$ we construct $\mathcal{P}_\mathcal{G}$ by simply defining $\mathcal{P}_\mathcal{G} = \mathcal{E}_\mathcal{G}$. It is easy to see that this construction indeed creates an instance of PIS and the context $\mathbb{K}_\mathcal{H}$ as well as the set $\mathcal{P}_\mathcal{G}$ can be constructed in time polynomial in the sizes of $\mathcal{H}$ and $\mathcal{G}$. Note that $\mathbb{K}_\mathcal{H}$ has the following property: $(*)$ If $B \subseteq M$ is an object intent, then any $A \subseteq B$ is closed since every such A can be expressed as the intersection of some object intents. We claim that $\mathcal{G}$ is the transversal hypergraph of $\mathcal{H}$ if and only if $\mathcal{P}_\mathcal{G}$ is precisely the set of pseudo-intents of $\mathbb{K}_\mathcal{H}$.

($\Rightarrow$) Assume $\mathcal{G}$ is the transversal hypergraph of $\mathcal{H}$. Take an edge of $\mathcal{G}$, say g. g is a minimal transversal of $\mathcal{H}$. By definition, for every $h_i \in \mathcal{E}_\mathcal{H}$, g satisfies $g \cap h_i \neq \emptyset$, which is equivalent to $g \nsubseteq \overline{h_i}$. This means that g is not closed in $\mathbb{K}_\mathcal{H}$. Because g is not contained in any object intent, hence $g'' = M$. Now take any $f \subsetneq g$. Since g is a *minimal* transversal, f will not be a transversal. That is, for some $h_i \in \mathcal{E}_\mathcal{H}$, $f \cap h_i = \emptyset$, which is equivalent to $f \subseteq \overline{h_i}$. Due to Property $(*)$, such f are closed. This means that g is not closed in $\mathbb{K}_\mathcal{H}$, but its all proper subsets f are closed, which implies that g is a pseudo-intent of $\mathbb{K}_\mathcal{H}$. Thus we have shown that if $\mathcal{G}$ is the transversal hypergraph of $\mathcal{H}$, then $\mathcal{P}_\mathcal{G}$ is precisely the set of pseudo-intents of $\mathbb{K}_\mathcal{H}$.

($\Leftarrow$) Assume $\mathcal{P}_\mathcal{G}$ is precisely the set of pseudo-intents of $\mathbb{K}_\mathcal{H}$. Take any pseudo-intent $p \in \mathcal{P}_\mathcal{G}$. By definition, p is not closed. Due to Property $(*)$, p is not contained in any object intent, i.e., $p \nsubseteq \overline{h_i}$, and thus $p'' = M$. This means that p satisfies $p \cap h_i \neq \emptyset$ for every edge $h_i \in \mathcal{E}_\mathcal{H}$, i.e., p is a transversal of $\mathcal{H}$. Moreover, p is minimal. Assume it were not. Then there would be another transversal $q \subsetneq p$, and q would satisfy $q \nsubseteq \overline{h_i}$ for every $h_i \in \mathcal{E}_\mathcal{H}$ as well. This would mean that q is not closed in $\mathbb{K}_\mathcal{H}$ and has the same closure as p which is M. This contradicts the fact that p is a pseudo-intent. Thus, p is indeed a minimal transversal of $\mathcal{H}$. We have shown that if $\mathcal{P}_\mathcal{G}$ is precisely the set of pseudo-intents of $\mathbb{K}_\mathcal{H}$, then $\mathcal{G}$ is the transversal hypergraph of $\mathcal{H}$, which completes the proof of our claim. □

Theorem 2 has the following consequences: For the type of formal contexts used in the reduction, i.e., where subsets of object intents are also closed sets, PIS and TRANS-HYP are computationally equivalent with respect to polynomial transformations, that is PIS is TRANS-HYP-complete. One can take an instance of PIS given with such a formal context and easily reduce it to TRANS-HYP. In this case, enumerating pseudo-intents (PIE) and enumerating hypergraph transversals (TRANS-ENUM) also become computationally equivalent problems. In order to solve an instance of PIE, one can construct the corresponding hypergraph and solve TRANS-ENUM on this hypergraph for instance by using the algorithm in [6] by Fredman and Khachiyan. The minimal transversals of this hypergraph will be the pseudo-intents of the original formal context.

5 Finding Explanations in the Duquenne-Guigues Base

In the present section, we investigate the problem of *finding explanations* in a Duquenne-Guigues Base, in other words, finding subsets of a given Duquenne-Guigues Base that has a given implication as consequence. In logic, for an arbitrary set of axioms, this problem is known as *axiom pinpointing*. In [1] it has been shown that in propositional Horn logic a given consequence can have exponentially many minimal explanations, and finding a minimum cardinality explanation is NP-complete.

From a logical point of view, our implications in FCA are also propositional Horn clauses. However, here we consider the above problem when the given set of implications is not an arbitrary set of implications, but it is the Duquenne-Guigues Base of a formal context. Our motivation for considering the problem under this restriction can be explained with the following scenario: Consider a domain expert that explores a context with attribute exploration and works with the resulting Duquenne-Guigues Base as a compact representation of the implications holding in her formal context. She notices that from this base, an implication that actually is not true in her application domain follows. That is, during attribute exploration she has wrongly confirmed some implication questions. In this scenario, finding explanations for the unwanted consequences would help the domain expert to solve the problem. We start with the formal definition of an explanation in a Duquenne-Guigues Base:

Definition 1. *Let $\mathcal{L}$ be the Duquenne-Guigues Base of a formal context on the set of attributes M, and $P \to Q$ be an implication such that $\mathcal{L} \models P \to Q$. We say that a subset $\mathcal{J} \subseteq \mathcal{L}$ explains $P \to Q$ if $\mathcal{J} \models P \to Q$ is satisfied. In this case we call $\mathcal{J}$ an explanation of $P \to Q$. We say that $\mathcal{J}$ is a minimal explanation of $P \to Q$ if no proper subset of $\mathcal{J}$ explains $P \to Q$.*

In the following for a set of implications $\mathcal{L}$ we will sometimes abuse the terminology and say "the Duquenne-Guigues Base of $Imp(\mathcal{L})$" for the Duquenne-Guigues Base of the set of all implications that follow from $\mathcal{L}$. The following lemma gives a syntactic characterization of the Duquenne-Guigues Base that will later help us to recognize whether a given set of implications is a Duquenne-Guigues Base. For a set X, $\mathcal{L}(X)$ denotes the implicational closure of X under the implication set $\mathcal{L}$.

Lemma 1. *Let $\mathcal{L} = \{P_i \to Q_i \mid 1 \leq i \leq n\}$ be a set of implications such that $P_i, Q_i \subseteq M$ and $Q_i \not\subseteq P_i$. $\mathcal{L}$ is the Duquenne-Guigues Base of $Imp(\mathcal{L})$ if and only if for every $1 \leq i \leq n$ the following two conditions are satisfied:*

- *P_i is closed under $\mathcal{L} \setminus \{P_i \to Q_i\}$, and*
- *$P_i \cup Q_i$ is closed under $\mathcal{L} \setminus \{P_i \to Q_i\}$.*

Proof. ($\Rightarrow$) If $\mathcal{L}$ is the Duquenne-Guigues Base of $Imp(\mathcal{L})$, then $P_1, \dots, P_n$ are pseudo-closed sets of the closure system induced by $Imp(\mathcal{L})$. Take any P_i . By definition P_i contains the closure of all P_j such that $P_j \subsetneq P_i$. Thus P_i is closed under $\mathcal{L} \setminus \{P_i \rightarrow Q_i\}$. By the definition of Duquenne-Guigues Base, $P_i \cup Q_i$ is also closed under $\mathcal{L} \setminus \{P_i \rightarrow Q_i\}$.

($\Leftarrow$) Assume $\mathcal{L}$ is a set of implications that satisfies the two conditions. We claim that it is the Duquenne-Guigues Base of $Imp(\mathcal{L})$. In order to prove this we need to show:

i) P_i are the pseudo-closed sets of the closure system induced by $Imp(\mathcal{L})$, where $1 \leq i \leq n$, and
ii) for every $1 \leq i \leq n$, $(Imp(\mathcal{L}))(P_i) = P_i \cup Q_i$ holds.

Since for any $X \subseteq M$, $\mathcal{L}(X) = (Imp(\mathcal{L}))(X)$ holds, we are going to show these for $\mathcal{L}$.

We start with *ii*): Take any $P_i \rightarrow Q_i$ and let $\mathcal{L}' = \mathcal{L} \setminus \{P_i \rightarrow Q_i\}$. We know that $P_i \cup Q_i$ is closed under $\mathcal{L}'$, i.e., $\mathcal{L}'(P_i \cup Q_i) = P_i \cup Q_i$. Then $P_i \cup Q_i$ is also closed under $\mathcal{L}$, i.e., $\mathcal{L}(P_i \cup Q_i) = P_i \cup Q_i$. Obviously $\mathcal{L}(P_i) = \mathcal{L}(P_i \cup Q_i)$ holds for every $P_i \rightarrow Q_i \in \mathcal{L}$. Then $\mathcal{L}(P_i) = P_i \cup Q_i$ holds. Thus we have shown *ii*.

In order to show *i* we need to show that:

1. P_i is not closed, i.e., $P_i \neq \mathcal{L}(P_i)$,
2. P_i is quasi-closed, i.e., for every $R \subseteq P_i$, $\mathcal{L}(R) \subseteq P_i$ holds or $\mathcal{L}(R) = \mathcal{L}(P_i)$ holds,
3. P_i strictly contains the closure of every strictly contained quasi-closed set, i.e., for every quasi-closed set $R \subsetneq P_i$, $\mathcal{L}(R) \subsetneq P_i$ holds.

We start with 1: We are given that for every $1 \leq i \leq n$, $Q_i \not\subseteq P_i$. Then $P_i \neq \mathcal{L}(P_i)$ holds trivially. For showing 2, take any P_i and some $R \subseteq P_i$. Let $\mathcal{L}' = \mathcal{L} \setminus \{P_i \rightarrow Q_i\}$. Since implicational closure is monotone, $\mathcal{L}'(R) \subseteq \mathcal{L}'(P_i)$. We are given that P_i is closed under $\mathcal{L}'$, i.e., $\mathcal{L}'(P_i) = P_i$ hence $\mathcal{L}'(R) \subseteq P_i$. If $\mathcal{L}'(R) = P_i$, then $\mathcal{L}(R) = \mathcal{L}(P_i)$ and we are done. If $\mathcal{L}'(R) \subsetneq P_i$, then $\mathcal{L}'(R) = \mathcal{L}(R) \subsetneq P_i$ and we are done. Thus we have shown that P_i is quasi-closed.

Now we are going to show 3: Take any P_i and some quasi-closed set $R \subsetneq P_i$. Since implicational closure is extensive, $R \subseteq \mathcal{L}(R)$ holds. If $\mathcal{L}(R) = R$ then $\mathcal{L}(R) \subsetneq P_i$ and we are done. If $R \subsetneq \mathcal{L}(R)$, then there exists an implication $P_j \rightarrow Q_j$, where $1 \leq j \leq n$, such that $P_j \subseteq R$ and $Q_j \not\subseteq R$. Together with $R \subsetneq P_i$, this implies $P_j \subsetneq P_i$.

We know that P_i is closed under $\mathcal{L} \setminus \{P_i \rightarrow Q_i\}$. Since $P_j \subsetneq P_i$, this implies $Q_j \subseteq P_i$, hence $P_j \cup Q_j \subseteq P_i$. Since $Q_i \not\subseteq P_i$, $Q_i \not\subseteq P_j \cup Q_j$. We know that $P_j \cup Q_j$ is closed under $\mathcal{L} \setminus \{P_j \rightarrow Q_j\}$. If $P_j \cup Q_j = P_i$ were satisfied, then $P_j \cup Q_j$ would not be closed under $\mathcal{L} \setminus \{P_j \rightarrow Q_j\}$ since $Q_i \not\subseteq P_j \cup Q_j$. Thus, $P_j \cup Q_j \subsetneq P_i$. By using *ii*, we can rewrite it as $\mathcal{L}(P_j) \subsetneq P_i$.

We know that R is quasi-closed. Since $P_j \subseteq R$, $\mathcal{L}(P_j) \subseteq R$ holds or $\mathcal{L}(P_j) = \mathcal{L}(R)$ holds. By *ii* we know that $\mathcal{L}(P_j) = P_j \cup Q_j$. Since $Q_j \not\subseteq R$, $\mathcal{L}(P_j) \subseteq R$ cannot hold. Thus, $\mathcal{L}(P_j) = \mathcal{L}(R)$ holds. Together with $\mathcal{L}(P_j) \subsetneq P_i$ from above, this implies that $\mathcal{L}(R) \subsetneq P_i$. Thus we have shown 3, which completes the proof of *i*, which in turn completes the proof of our claim. □

Using Lemma 1, we can show that in the worst case, a given implication can have exponentially many minimal explanations in a given Duquenne-Guigues Base. The following example demonstrates this situation:

Example 1. Consider the set of implications

$$\mathcal{L} := \bigcup_{1 \leq i \leq n} \{\{x, b_{i-1}\} \rightarrow \{p_i, q_i\},\ \{y, p_i\} \rightarrow \{b_i\},\ \{y, q_i\} \rightarrow \{b_i\}\}$$

on the set of attributes $M = \{b_0, x, y, \} \cup \{b_i, p_i, q_i \mid 1 \leq i \leq n\}$. Note that none of the left handsides is contained in another left handside or in the union of left and right handsides of another implication, i.e., $\mathcal{L}$ satisfies the two conditions stated in Lemma 1 thus, $\mathcal{L}$ is the Duquenne-Guigues Base of $Imp(\mathcal{L})$.

Consider the implication $\{b_0, x, y\} \rightarrow \{b_n\}$ that follows from $\mathcal{L}$. A minimal explanation of this implication is either of the form $\{\{b_0, x\} \rightarrow \{p_1, q_1\}, \{y, p_1\} \rightarrow \{b_1\}, \ldots\}$ or, $\{\{b_0, x\} \rightarrow \{p_1, q_1\}, \{y, q_1\} \rightarrow \{b_1\}, \ldots\}$. That is at each step i, where $1 \leq i \leq n$, we have two choices since the attribute b_i can be generated either by the implication $\{y, p_i\} \rightarrow \{b_i\}$, or by the implication $\{y, q_i\} \rightarrow \{b_i\}$. This means that there are 2^n minimal explanations. Since the size of $\mathcal{L}$ is linear in n, the example shows that there can be exponentially many minimal explanations in a given Duquenne-Guigues Base.

5.1 Minimum Cardinality Explanation

Although there can be exponentially minimal explanations, given a Duquenne-Guigues Base $\mathcal{L}$ and an implication ψ that follows from it, it is not difficult to find one minimal explanation of ψ in $\mathcal{L}$. We can just start with $\mathcal{L}$, iterate over the implications in $\mathcal{L}$ and remove an implication if ψ still follows from the remaining set of implications. Clearly, this algorithm terminates since $\mathcal{L}$ is finite. It is correct since ψ still follows from the remaining set of implications and none of the implications in the remaining set can be removed without destroying this property.

However, if we want an explanation that is not only minimal w.r.t. set inclusion, but also minimal w.r.t. cardinality, the problem becomes harder. In [1] it has been shown that for an arbitrary set of implications (there called propositional Horn axioms) finding an explanation within a specified cardinality bound is NP-complete. Here we consider this problem for the case when the given set of implications is not arbitrary, but it is the Duquenne-Guigues Base of implications holding in a closure system. It turns out that under this restriction the problem does not become easier, i.e., it remains NP-complete.

Problem: MINIMUM CARDINALITY EXPLANATION (MCE)
Input: A Duquenne-Guigues Base $\mathcal{L}$, an implication $L \rightarrow R$ s.t. $\mathcal{L} \models L \rightarrow R$ and a natural number n.
Question: Is there an explanation of $L \rightarrow R$ in $\mathcal{L}$ with cardinality less than or equal to n, i.e., is there an $\mathcal{L}' \subseteq \mathcal{L}$ such that $\mathcal{L}' \models L \rightarrow R$ and $|\mathcal{L}'| \leq n$?

Theorem 3. MCE *is* NP-*complete.*

Proof. The problem is in NP. We can nondeterministically guess a subset $\mathcal{L}'$ of $\mathcal{L}$ with cardinality n, and in polynomial time check whether $\mathcal{L}' \models L \to R$. This test can indeed be done in polynomial time by checking whether $R \subseteq \mathcal{L}'(L)$.

In order to show NP-hardness, we are going to give a reduction from the NP-complete problem VERTEX COVER [9]. Recall that a vertex cover of the graph $\mathcal{G} = (V, E)$ is a set $W \subseteq V$ such that for every edge $\{u, v\} \in E$, $u \in W$ holds, or $v \in W$ holds. The problem VERTEX COVER is defined as follows:

Problem: VERTEX COVER
Input: Graph $\mathcal{G} = (V, E)$, a natural number n.
Question: Is there a vertex cover of $\mathcal{G}$ of size less than or equal to n?

Consider an instance of the VERTEX COVER problem given by $\mathcal{G} = (V, E)$, where $V = \{v_1, \ldots, v_l\}$, $E = \{e_1, \ldots, e_k\}$, and edge $e_i = \{v_{i1}, v_{i2}\}$. We construct an instance of the MCE in the following way: For every vertex $v \in V$ we introduce an attribute m_v, for every edge e_j, $1 \leq j \leq k$, we introduce an attribute m_{e_j}, and finally two more additional attributes m_a and m_b. Using these attributes we construct the following set of implications:

$$\mathcal{L} := \{\{m_v\} \to \{m_{e_j} \mid v \in e_j, 1 \leq j \leq k\} \mid v \in V\} \cup \{\{m_a, m_{e_1}, \ldots, m_{e_k}\} \to \{m_b\}\}.$$

Note that none of the implications in $\mathcal{L}$ contains the left handside of another implication in its left handside or in the union of its left and right handsides. Thus, due to Lemma 1, $\mathcal{L}$ is indeed the Duquenne-Guigues Base of $Imp(\mathcal{L})$. In addition to $\mathcal{L}$, we construct the following implication ψ that follows from $\mathcal{L}$: $\psi : \quad \{m_a\} \cup \{m_v \mid v \in V\} \to \{m_b\}$. It is not difficult to see that both $\mathcal{L}$ and ψ can be constructed in time polynomial in the size of $\mathcal{G}$, and that ψ follows from $\mathcal{L}$. We claim that $\mathcal{G}$ has a vertex cover of size less than or equal to n, where $n \leq |V|$, if and only if $\mathcal{L}$ has a subset $\mathcal{L}'$ that explains ψ, and the size of $\mathcal{L}'$ is polynomial in n.

($\Rightarrow$) Assume $W \subseteq V$ is a vertex cover of $\mathcal{G}$. Then the following set $\mathcal{L}' \subseteq \mathcal{L}$ constructed by using W is an explanation of ψ:

$$\begin{aligned}\mathcal{L}' := \{&\{m_w\} \to \{m_{e_j} \mid w \in e_j, 1 \leq j \leq k\} \mid w \in W\} \cup \\ &\{\{m_a, m_{e_1}, \ldots, m_{e_k}\} \to \{m_b\}\}.\end{aligned}$$

Since W is a vertex cover, it contains at least one vertex from every edge e_j, $1 \leq j \leq k$. Thus, $\{m_{e_1}, \ldots m_{e_k}\} \subseteq \mathcal{L}'(\{m_w \mid w \in W\})$. Since $\{m_w \mid w \in W\} \subseteq \{m_v \mid v \in V\}$, this implies that $\{m_{e_1}, \ldots m_{e_k}\} \subseteq \mathcal{L}'(\{m_v \mid v \in V\})$, which in turn implies that $\{m_b\} \subseteq \mathcal{L}'(\{m_a\} \cup \{m_v \mid v \in V\})$. Thus we have shown that $\mathcal{L}'$ is indeed an explanation of ψ, and that it contains exactly $n+1$ implications.

($\Leftarrow$) Now assume that $\mathcal{L}$ has a subset $\mathcal{L}'$ of size m that is an explanation of ψ. $\mathcal{L}'$ should contain the implication $\{m_a, m_{e_1}, \ldots, m_{e_k}\} \to \{m_b\}$, since otherwise the attribute m_b cannot be generated. Moreover, since the premise of this implication contains the attributes $m_{e_1}, \ldots, m_{e_k}$, $\mathcal{L}'$ should also contain implications of type $\{m_w\} \to \{m_{e_j} \mid w \in e_j\}$ such that every m_{e_j}, $1 \leq j \leq k$, is generated.

This means that the set W of such w is indeed a vertex cover since it intersects every edge e_j, $1 \leq j \leq k$. Thus we have shown that W is a vertex cover of $\mathcal{G}$ and it has size $m-1$. This finishes the proof of the claim that $\mathcal{G}$ has a vertex cover of size n if and only if $\mathcal{L}$ has a subset of size $n+1$ that explains ψ. □

5.2 Counting Minimal Explanations

In applications where one is interested in all explanations that are minimal w.r.t. set inclusion, it might be useful to know in advance how many of them exist. Next we consider this counting problem. It turns out that it is hard for the counting complexity class #P [20], i.e., it is intractable.

Problem: #MINIMAL EXPLANATION (#ME)
Input: A Duquenne-Guigues Base $\mathcal{L}$, and an implication $L \rightarrow R$ s.t. $\mathcal{L} \models L \rightarrow R$.
Output: The number of all minimal explanations of $L \rightarrow R$, i.e., $|\{\mathcal{L}' \subseteq \mathcal{L} \mid \mathcal{L}' \models L \rightarrow R \text{ and } \forall \mathcal{L}'' \subsetneq \mathcal{L}'.\mathcal{L}'' \not\models L \rightarrow R\}|$.

Theorem 4. #ME *is* #P*-complete.*

Proof. The problem is in #P. Given a Duquenne-Guigues Base $\mathcal{L}$, an implication $L \rightarrow R$ that follows from $\mathcal{L}$, and a set $\mathcal{L}' \subseteq \mathcal{L}$ we can in polynomial time verify whether $\mathcal{L}' \models L \rightarrow R$ just by checking whether $R \subseteq \mathcal{L}'(L)$ holds.

In order to show #P-hardness, we are going to give a parsimonious reduction from the #P-complete problem #MINIMAL VERTEX COVER, which is the problem of counting the minimal vertex covers of a graph. It has been shown to be #P-complete in [21]. In our reduction we are going to use the same construction as in the proof of Theorem 3, i.e., from a given graph $\mathcal{G}$ we construct the same Duquenne-Guigues Base $\mathcal{L}$, and the same implication ψ as in Theorem 3. What we additionally need to show here is that this construction establishes a bijection between minimal vertex covers of $\mathcal{G}$ and minimal explanations of ψ in $\mathcal{L}$.

First we show that the construction in the proof of Theorem 3 establishes an injection: Assume $W \subseteq V$ is a *minimal* vertex cover of $\mathcal{G}$, then the following set of implications is a *minimal* explanation of ψ in $\mathcal{L}$:

$$\begin{aligned}\mathcal{L}' := \{&\{m_w\} \rightarrow \{m_{e_j} \mid w \in e_j, 1 \leq j \leq k\} \mid w \in W\} \cup \\ &\{\{m_a, m_{e_1}, \ldots, m_{e_k}\} \rightarrow \{m_b\}\}.\end{aligned}$$

In the proof of Theorem 3 we have already shown that $\mathcal{L}'$ is an explanation. Here we need to show that it is minimal as well. If W is minimal, then removal of any vertex w from W will result in a $Y \subsetneq W$ such that $v_{j1} \notin Y$ and $v_{j2} \notin Y$ for some edge e_j. This implies that removal of the corresponding implication $\{m_w\} \rightarrow \{m_{e_j} \mid w \in e_j\}$ from $\mathcal{L}'$ will result in a $\mathcal{L}''$ such that the attribute m_{e_j} does not appear on the right handside of any of the implications in $\mathcal{L}''$, which means that $\mathcal{L}''$ cannot explain ψ, i.e., $\mathcal{L}'$ is minimal.

Now we show that it establishes a surjection: Assume $\mathcal{L}'$ is a minimal explanation. Then every m_{e_j}, $1 \leq j \leq k$, occurs at least once on the right handside of some implication of the form $\{m_w\} \rightarrow \{m_{e_j} \mid w \in e_j\}$, where $w \in W$, because otherwise $\mathcal{L}'$ cannot explain ψ. We have already shown in the proof of Theorem Theorem 3 that such a W is a vertex cover. Moreover, removal of any implication of this form from $\mathcal{L}'$ results in a set of implications that is not an explanation. This is because $\mathcal{L}'$ is a minimal explanation. That is, removal of any w from W results in a $Y \subsetneq W$ such that $v_{j1} \notin Y$ and $v_{j2} \notin Y$ for some $1 \leq j \leq k$, i.e., W is minimal. Thus we have shown that our construction establishes a bijection between minimal vertex covers and minimal explanations. □

5.3 Computing All Minimal Explanations

In Example 1 we have demonstrated that a given implication can have exponentially many minimal explanations in a given Duquenne-Guigues Base. Given this fact, it is clearly not possible to enumerate all minimal explanations in time polynomial in the size of the input. In this case one can investigate the existence of an output polynomial algorithm for this problem:

Problem: MINIMAL EXPLANATION ENUMERATION (MEE)
Input: A Duquenne-Guigues Base $\mathcal{L}$ and an implication $L \to R$ s.t. $\mathcal{L} \models L \to R$.
Output: The set of all minimal explanations of $L \to R$ in $\mathcal{L}$, i.e., $\{\mathcal{L}' \subseteq \mathcal{L} \mid \mathcal{L}' \models L \to R \ \ and \ \ \forall \mathcal{L}'' \subsetneq \mathcal{L}'.\ \mathcal{L}'' \not\models L \to R\}$.

In order to investigate the complexity of this enumeration problem, we need to investigate the following decision problem:

Problem: ADDITIONAL MINIMAL EXPLANATION (AME)
Input: A Duquenne-Guigues Base $\mathcal{L}$, an implication $L \to R$ s.t. $\mathcal{L} \models L \to R$, and a set of minimal explanations of $L \to R$ in $\mathcal{L}$, i.e, $\mathscr{J} = \{\mathcal{J}_i \mid \mathcal{J}_i \subseteq \mathcal{L}, \mathcal{J}_i \models L \to R \ and\ \forall \mathcal{J}' \subsetneq \mathcal{J}_i.\ \mathcal{J}' \not\models L \to R\}$
Question: Is there a minimal explanation that is not already listed in $\mathscr{J}$, i.e., $\mathcal{J} \subseteq \mathcal{L}$ such that $\mathcal{J} \models L \to R$, $\forall \mathcal{J}' \subsetneq \mathcal{J}.\ \mathcal{J}' \not\models L \to R$ and $\mathcal{J} \notin \mathscr{J}$?

Because if AME is not in P, there cannot be an algorithm that solves MEE in output polynomial time (unless P = NP). We can show it by the same argument used in the proofs of Propositions 1 and 3. It is not difficult to see that AME is in coNP. Given an instance of AME with the Duquenne-Guigues Base $\mathcal{L}$, the implication ψ and a set of minimal explanations $\mathscr{J}$, we can nondeterministically guess a minimal subset of $\mathcal{L}$ that is not already contained in $\mathscr{J}$ and in polynomial time verify that this subset *does not* explain ψ. Unfortunately we do not know the lower bound of this problem at the moment. It is definitely an interesting question whether this problem, like API and PIS, is also related to the decision problems SIMPLE-H-SAT and TRANS-HYP from hypergraph theory.

6 Concluding Remarks and Future Work

We have considered several decision, enumeration and counting problems related to pseudo-intents. Among them, PIE, the problem of enumerating pseudo-intents has been the central point of our interest. The question whether this problem can be solved in output polynomial time or not remains unfortunately open. However we have formulated two decision questions, namely API and PIS, that are crucial in determining the complexity of PIE. Some interesting consequences of our results can be summed up as follows:

- If any of the problems API, or PIS turns out to be coNP-hard, then unless P = NP, there cannot be an algorithm that solves PIE in output polynomial time (Proposition 1, Proposition 3).
- Showing that any of the problems API or PIS is polynomial implies that the open problems TRANS-HYP and SIMPLE-H-SAT are also polynomial (Theorem 1, Theorem 2, [3]).
- Even if TRANS-HYP and SIMPLE-H-SAT turn out to be polynomial, API and PIS can still be coNP-hard, thus it can still be the case that PIE is not solvable in output polynomial time.
- Even if API and PIS turn out to be polynomial, it can still be the case that PIE is not solvable in output polynomial time.

We have also investigated the complexity of finding explanations, i.e., subsets from which a given implication follows, in a given Duquenne-Guigues Base. We have shown that finding a minimum cardinality one is NP-complete, and counting minimal explanations is #P-complete.

As future work, we are going to work on determining the exact complexity of the problems API and PIS. For API, we are going to investigate whether the hardness result [3] on hypergraph saturation for arbitrary graphs carries over to API on arbitrary formal contexts. For PIS, we are going to investigate the types of formal context where PIS and TRANS-HYP (and thus PIE and TRANS-ENUM) become computationally equivalent problems, and find out whether this type of formal contexts are natural in some applications, and how often they occur in practice. One other interesting question is of course the lower complexity bound for checking whether a set is a pseudo-intent. We are going to investigate whether this problem is also related to some hypergraph problem. In addition to this, we are going to work on determining the exact complexity of counting pseudo-intents.Note that in [15,16] it has been mentioned that this problem is in #P, but this is not true. The results there only imply that this problem is in #·coNP [12], which contains #P. On the explanations side, we are going to work on determining the exact complexity of AME.

Acknowledgments. Thanks to Felix Distel for proof-reading a preliminary version of this work.

References

1. Baader, F., Peñaloza, R., Suntisrivaraporn, B.: Pinpointing in the description logic $\mathcal{EL}^+$. In: Hertzberg, J., Beetz, M., Englert, R. (eds.) KI 2007. LNCS, vol. 4667, pp. 52–67. Springer, Heidelberg (2007)
2. Berge, C.: Hypergraphs. Elsevier Science Publishers B.V, North Holland (1989)

3. Eiter, T., Gottlob, G.: Identifying the minimal transversals of a hypergraph and related problems. SIAM J. on Computing 24(6), 1278–1304 (1995)
4. Eiter, T., Gottlob, G.: Hypergraph transversal computation and related problems in logic and AI. In: Flesca, S., Greco, S., Leone, N., Ianni, G. (eds.) JELIA 2002. LNCS, vol. 2424, pp. 549–564. Springer, Heidelberg (2002)
5. Eiter, T., Gottlob, G., Makino, K.: New results on monotone dualization and generating hypergraph transversals. SIAM J. on Computing 32(2), 514–537 (2003)
6. Fredman, M.L., Khachiyan, L.: On the complexity of dualization of monotone disjunctive normal forms. J. of Algorithms 21(3), 618–628 (1996)
7. Ganter, B.: Two basic algorithms in concept analysis. Technical Report Preprint-Nr. 831, Technische Hochschule Darmstadt, Darmstadt, Germany (1984)
8. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin (1999)
9. Garey, M.R., Johnson, D.S.: Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman & Company, New York (1990)
10. Guigues, J.-L., Duquenne, V.: Familles minimales d'implications informatives resultant d'un tableau de données binaries. Mathématiques, Informatique et Sciences Humaines 95, 5–18 (1986)
11. Gunopulos, D., Khardon, R., Mannila, H., Toivonen, H.: Data mining, hypergraph transversals, and machine learning. In: Proc. of the Sixteenth Symposium on Principles of Database Systems (PODS 1997), pp. 209–216 (1997)
12. Hemaspaandra, L.A., Vollmer, H.: The satanic notations: counting classes beyond #P and other definitional adventures. ACM SIGACT-Newsletter 26(1), 2–13 (1995)
13. Johnson, D.S., Papadimitriou, C.H., Yannakakis, M.: On generating all maximal independent sets. Information Processing Letters 27(3), 119–123 (1988)
14. Kuznetsov, S.O.: On the intractability of computing the Duquenne-Guigues Base. J. of Universal Computer Science 10(8), 927–933 (2004)
15. Kuznetsov, S.O., Obiedkov, S.A.: Counting pseudo-intents and #P-completeness. In: Missaoui, R., Schmidt, J. (eds.) ICFCA 2006. LNCS, vol. 3874, pp. 306–308. Springer, Heidelberg (2006)
16. Kuznetsov, S.O., Obiedkov, S.A.: Some decision and counting problems of the duquenne-guigues basis of implications. Discrete Applied Mathematics 156(11), 1994–2003 (2008)
17. Mannila, H., Räihä, K.-J.: Design by example: An application of armstrong relations. J. of Computer and System Sciences 33(2), 126–141 (1986)
18. Obiedkov, S.A., Duquenne, V.: Attribute-incremental construction of the canonical implication basis. Annals of Mathematics and AI 49(1-4), 77–99 (2007)
19. Rudolph, S.: Some notes on pseudo-closed sets. In: Kuznetsov, S.O., Schmidt, S. (eds.) ICFCA 2007. LNCS, vol. 4390, pp. 151–165. Springer, Heidelberg (2007)
20. Valiant, L.G.: The complexity of computing the permanent. Theoretical Computer Science 8(2), 189–201 (1979)
21. Valiant, L.G.: The complexity of enumeration and reliability problems. SIAM J. on Computing 8(3), 410–421 (1979)

Exploring Finite Models in the Description Logic $\mathcal{EL}_{\mathrm{gfp}}$

Franz Baader[⋆] and Felix Distel[⋆⋆]

Theoretical Computer Science, TU Dresden, Germany
{baader,felix}@tcs.inf.tu-dresden.de

Abstract. In a previous ICFCA paper we have shown that, in the Description Logics $\mathcal{EL}$ and $\mathcal{EL}_{\mathrm{gfp}}$, the set of general concept inclusions holding in a finite model always has a finite basis. In this paper, we address the problem of how to compute this basis efficiently, by adapting methods from formal concept analysis.

1 Introduction

Description Logics (DLs) [3] are a well-investigated family of logic-based knowledge representation formalisms, which are employed in various application domains, such as natural language processing, configuration, databases, and bio-medical ontologies, but their most notable success so far is the adoption of the DL-based language OWL [11] as standard ontology language for the semantic web. From the Description Logic point of view, an ontology is a finite set of general concept inclusion axioms (GCIs) of the form $C \sqsubseteq D$, where C, D are concepts defined using an appropriate concept description language. Such a concept description language allows one to construct complex concepts out of concept names (unary predicates, interpreted as sets) and roles (binary predicates, interpreted as binary relations) using certain concept constructors. Complex concepts are again interpreted as sets. To be more precise, given an interpretation of the concept and role names, the semantics of the concept constructors determines, for every complex concept, a unique set as the extension of this concept. The GCI $C \sqsubseteq D$ states that, in a model of the ontology, the extension of the concept C must be a subset of the extension of the concept D.

When defining a DL-based ontology, one must first decide on which vocabulary (i.e., concept and role names) to use, and then define appropriate constraints on the interpretation of this vocabulary using GCIs. The work described in this paper is motivated by the fact that coming up with the right GCIs by hand is usually not an easy task. Instead, we propose an approach where the knowledge engineer is required to provide us with a finite model, which should be seen as an abstraction or approximation of the application domain to be modeled. We then automatically generate a finite basis of the GCIs holding in the model, i.e.,

[⋆] Supported by DFG under grant BA 1122/12-1.

[⋆⋆] Supported by the Cusanuswerk.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 146–161, 2009.

a finite set of GCIs that hold in this model and from which all GCIs holding in the model and expressible in the employed concept description language follow. The knowledge engineer can use the computed basis as a starting point for the definition of the ontology. She may want to weaken or even remove some of the GCIs if the chosen model was too restricted, and thus satisfies GCIs that actually do not hold in all intended models. As an example, assume that we want to define a family ontology, using the concept names $\mathsf{Male}, \mathsf{Father}, \mathsf{Female}, \mathsf{Mother}$, and the role name child. Consider a finite model with two families. The first family consists of John, Michelle, and Mackenzie, where John is male and a father (i.e., John belongs to the interpretation of the concept names Male and Father), Michelle is female and a mother, and Mackenzie is female and a child of both John and Michelle. The second family consists of Paul, Linda, and James, where Paul is male and a father, Linda is female and a mother, and James is male and a child of both Paul and Linda. In this model, the GCIs

$$\mathsf{Father} \sqsubseteq \mathsf{Male} \sqcap \exists \mathsf{child}.\top \quad \text{and} \quad \mathsf{Mother} \sqsubseteq \mathsf{Female} \sqcap \exists \mathsf{child}.\top$$

hold. The first one says that every father is male and has a child, and the second one says that every mother is female and has a child. If we had used a model consisting of only the first family, then we would have obtained the too specific GCIs $\mathsf{Father} \sqsubseteq \mathsf{Male} \sqcap \exists \mathsf{child}.\mathsf{Female}$ and $\mathsf{Mother} \sqsubseteq \mathsf{Female} \sqcap \exists \mathsf{child}.\mathsf{Female}$, where mothers and fathers always have female children.

For the approach sketched above to work, the set of GCIs holding in a finite model and expressible in the employed concept description language must have a *finite* basis. Using methods from formal concept analysis (FCA), we have shown in [5] that this is the case for the language $\mathcal{EL}$, which allows for the concept constructors $\top$ (top concept), $C \sqcap D$ (conjunction), and $\exists r.C$ (existential restriction). Though being quite inexpressive, $\mathcal{EL}$ has turned out to be very useful for representing biomedical ontologies such as SNOMED [14] and the Gene Ontology [16]. A major advantage of using an inexpressive DL like $\mathcal{EL}$ is that it allows for efficient reasoning procedures [2,7]. Because of the nice algorithmic properties of $\mathcal{EL}$, the new OWL standard will contain a profile, called OWL 2 EL, that is based on $\mathcal{EL}$.

In [5], the existence of a finite basis is actually first shown for $\mathcal{EL}_{\mathrm{gfp}}$, which extends $\mathcal{EL}$ with cyclic concept definitions interpreted with greatest fixpoint semantics. The advantage of using $\mathcal{EL}_{\mathrm{gfp}}$ rather than $\mathcal{EL}$ is that, in $\mathcal{EL}_{\mathrm{gfp}}$, every set of objects (i.e., elements of the domain of a given finite model) always has a most specific concept describing these objects. Going from a set of objects to its most specific concept corresponds to the $\cdot'$ operator in FCA, which goes from a set of objects in a formal context to the set of all attributes that these objects have in common. The existence of most specific concepts in $\mathcal{EL}_{\mathrm{gfp}}$ thus allowed us to employ methods from FCA. In a second step, we have shown in [5] that the $\mathcal{EL}_{\mathrm{gfp}}$-basis can be turned into an $\mathcal{EL}$-basis by unraveling cyclic concept definitions up to a level determined by the cardinality of the given finite model.

In [5], we concentrated on showing the *existence* of a finite basis for $\mathcal{EL}_{\mathrm{gfp}}$ and $\mathcal{EL}$. Of course, if the approach for automatically generating GCIs sketched above is to be used in practice, we also need to find efficient algorithms for computing such bases. This is the topic of the present paper. First, we show that the algorithm for computing an implication basis of a given formal context known from classical FCA can be adapted to our purposes. In contrast to the classical case, we cannot assume that all attributes of the context are known from the beginning. Instead, the set of attribute can be extended during the run of the algorithm. This is vital for obtaining an efficient algorithm. In a second step, we then extend this algorithm to an exploration algorithm. The advantage of this second algorithm is that it no longer requires the finite model to be completely represented in the computer from the beginning. As in the case of classical attribute exploration [9], the model is assumed to be "known" by an expert, who during the exploration process extends the represented part of the model in order to provide counterexamples to implication questions.

We concentrate on computing a finite $\mathcal{EL}_{\mathrm{gfp}}$-basis since this basis can be turned into an $\mathcal{EL}$-basis as described in [5]. Due to the space limitation, we cannot give complete proofs of our results. They can be found in [4]. We also assume that the reader is familiar with the basic notion and results of formal concept analysis (FCA).

2 A Finite Implication Basis for $\mathcal{EL}_{\mathrm{gfp}}$

We start by defining $\mathcal{EL}$, and show how it can be extended to $\mathcal{EL}_{\mathrm{gfp}}$. Then we define most specific concepts in $\mathcal{EL}_{\mathrm{gfp}}$, and show how they can be used to obtain a finite basis of the $\mathcal{EL}_{\mathrm{gfp}}$-GCIs holding in a finite model.

The DLs $\mathcal{EL}$ and $\mathcal{EL}_{\mathrm{gfp}}$

Because of the space restriction, we can only give a very compact introduction into these DLs (see [1] for more details). Concept descriptions of $\mathcal{EL}$ are built from a set $\mathcal{N}_c$ of concept names and a set $\mathcal{N}_r$ of role names, using the constructors top concept, conjunction, and existential restriction:

- concept names and the top concept $\top$ are $\mathcal{EL}$-concept descriptions;
- if C, D are $\mathcal{EL}$-concept descriptions and r is a role name, then $C \sqcap D$ and $\exists r.C$ are $\mathcal{EL}$-concept descriptions.

In the following, we assume that the sets $\mathcal{N}_c$ and $\mathcal{N}_r$ are finite. This assumption is reasonable since a finite ontology can contain only finitely many concept and role names.

Models of $\mathcal{EL}$ are pairs $(\Delta_i, \cdot^i)$, where Δ_i is a non-empty set, and $\cdot^i$ maps role names r to binary relations $r^i \subseteq \Delta_i \times \Delta_i$ and $\mathcal{EL}$-concept descriptions C to their *extensions* $C^i \subseteq \Delta_i$ such that

$$\top^i = \Delta_i, \qquad (C_1 \sqcap C_2)^i = C_1^i \cap C_2^i, \quad \text{and}$$
$$(\exists r.D)^i \quad = \{d \in \Delta_i \mid \exists e \in D^i \text{ such that } (d,e) \in r^i\}.$$

Subsumption and equivalence between $\mathcal{EL}$-concept descriptions is defined in the usual way, i.e., C is subsumed by D (written $C \sqsubseteq D$) iff $C^i \subseteq D^i$ for all models i, and C is equivalent to D (written $C \equiv D$) iff $C \sqsubseteq D$ and $D \sqsubseteq C$.

$\mathcal{EL}_{\text{gfp}}$ is the extension of $\mathcal{EL}$ by cyclic concept definitions interpreted with greatest fixpoint (gfp) semantics. In $\mathcal{EL}_{\text{gfp}}$, we assume that the set of concept names is partitioned into the set $\mathcal{N}_{\text{prim}}$ of primitive concepts and the set $\mathcal{N}_{\text{def}}$ of defined concepts. A *concept definition* is of the form

$$B_0 \equiv P_1 \sqcap \ldots \sqcap P_m \sqcap \exists r_1.B_1 \sqcap \ldots \sqcap \exists r_n.B_n$$

where $B_0, B_1, \ldots, B_n \in \mathcal{N}_{\text{def}}$, $P_1, \ldots, P_m \in \mathcal{N}_{\text{prim}}$, and $r_1, \ldots, r_n \in \mathcal{N}_r$. The empty conjunction (i.e., $m = 0 = n$) stands for $\top$. A *TBox* is a finite set of concept definitions such that every defined concept occurs at most once as a left-hand side of a concept definition.

Definition 1 ($\mathcal{EL}_{\text{gfp}}$-concept description). *An* $\mathcal{EL}_{\text{gfp}}$-concept description *is a tuple $(A, \mathcal{T})$ where $\mathcal{T}$ is a TBox and A is a defined concept occurring on the left-hand side of a definition in $\mathcal{T}$.*

Models of $\mathcal{EL}_{\text{gfp}}$ are of the form $i = (\Delta_i, \cdot^i)$ where Δ_i is a non-empty set, and $\cdot^i$ maps role names r to binary relations $r^i \subseteq \Delta_i \times \Delta_i$ and primitive concepts to subsets of Δ_i. The mapping $\cdot^i$ is extended to $\mathcal{EL}_{\text{gfp}}$-concept descriptions $(A, \mathcal{T})$ by interpreting the TBox $\mathcal{T}$ with gfp-semantics: consider all extensions of i to the defined concepts that satisfy the concept definitions in $\mathcal{T}$, i.e., assign the same extension to the left-hand side and the right-hand side of each definition. Among these extensions of i, the *gfp-model of $\mathcal{T}$ based on i* is the one that assigns the largest sets to the defined concepts (see [1] for a more detailed definition of gfp-semantics). The *extension $(A, \mathcal{T})^i$ of $(A, \mathcal{T})$* in i is the set assigned to A by the gfp-model of $\mathcal{T}$ based on i.

Subsumption and equivalence between $\mathcal{EL}_{\text{gfp}}$-concept descriptions is defined as in the case of $\mathcal{EL}$-concept descriptions. It is easy to see that acyclic $\mathcal{EL}_{\text{gfp}}$-concept descriptions (i.e., ones where the TBox component is acyclic) correspond exactly to $\mathcal{EL}$-concept descriptions. This shows that $\mathcal{EL}$ can indeed be seen as a sublanguage of $\mathcal{EL}_{\text{gfp}}$. In the following, we will not distinguish an acyclic $\mathcal{EL}_{\text{gfp}}$-concept description from its equivalent $\mathcal{EL}$-concept description.

Most Specific Concepts in $\mathcal{EL}_{\text{gfp}}$

In FCA, the prime operators $\cdot'$ play an important rôle. Given a set of attributes B, the set B' consists of the objects of the given context satisfying all these attributes. In DL, the operator $\cdot^i$ plays a similar rôle: given a concept description C, the set C^i consists of all objects in the model i (i.e., elements of Δ_i) satisfying C, i.e., belonging to the extension of C. In FCA, the prime operator can also be applied in the other direction: given a set of objects A, it yields the set A' of attributes common to the objects in A. This is equivalent to defining $A' = B_{\max}$, where $B_{\max}$ is the greatest subset of M such that $A \subseteq B'_{\max}$. In DL, the most specific concept plays the rôle of this $\cdot'$ operator.

Definition 2 (Most specific concept). *Let i be a finite $\mathcal{EL}_{\mathrm{gfp}}$-model and $X \subseteq \Delta_i$. The $\mathcal{EL}_{\mathrm{gfp}}$-concept description C is the* most specific $\mathcal{EL}_{\mathrm{gfp}}$-concept of X in i *if it is the least $\mathcal{EL}_{\mathrm{gfp}}$-concept description such that $X \subseteq C^i$. By* least $\mathcal{EL}_{\mathrm{gfp}}$-concept description *we mean that every other $\mathcal{EL}_{\mathrm{gfp}}$-concept description $\bar{C}$ satisfying $X \subseteq \bar{C}^i$ also satisfies $C \sqsubseteq \bar{C}$.*

Calling an $\mathcal{EL}_{\mathrm{gfp}}$-concept description satisfying the above definition *the* most specific $\mathcal{EL}_{\mathrm{gfp}}$-concept of X in i is justified by the fact that most specific concepts are obviously unique up to equivalence. In [5] it is shown that, for $\mathcal{EL}_{\mathrm{gfp}}$, the most specific concept always exists.[1]

Theorem 1. *For any finite $\mathcal{EL}_{\mathrm{gfp}}$-model i and any set $X \subseteq \Delta_i$, the most specific $\mathcal{EL}_{\mathrm{gfp}}$-concept of X in i exists and can be computed effectively.*

In the following, we denote the most specific $\mathcal{EL}_{\mathrm{gfp}}$-concept of X in i by X^i. This overloading of the notation $\cdot^i$ corresponds to the one employed in FCA for $\cdot'$. The following lemma (taken from [5]) shows that the operators $\cdot^i$ indeed behave similarly to the $\cdot'$ operators.

Lemma 1. *Let $\mathcal{L}$ be a language for which X^i exists for every $X \subseteq \Delta_i$ and every $i \in \mathcal{I}$. Let $i \in \mathcal{I}$ be an interpretation, $X, Y \in \Delta_i$ sets of objects and C, D be concept descriptions. Then the following statements hold*

1. $X \subseteq Y \Rightarrow X^i \sqsubseteq Y^i$
2. $C \sqsubseteq D \Rightarrow C^i \subseteq D^i$
3. $X \subseteq X^{ii}$
4. $C^{ii} \sqsubseteq C$
5. $X^i \equiv X^{iii}$
6. $C^i = C^{iii}$
7. $X \subseteq C^i \Leftrightarrow X^i \sqsubseteq C$.

The Set of GCIs Holding in a Finite Model and a Basis for this Set

An expression of the form $C \rightarrow D$, where C, D are $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions, is called an $\mathcal{EL}_{\mathrm{gfp}}$-GCI (or simply GCI).[2] We say that an GCI $C \rightarrow D$ *holds* in the model i iff $C^i \subseteq D^i$. Given a set of GCIs $\mathcal{B}$, we say that the GCI $C \rightarrow D$ *follows from* $\mathcal{B}$ iff $C \rightarrow D$ holds in all models in which all implications from $\mathcal{B}$ hold.

Definition 3 (Basis). *For a given finite model i we say that a set of $\mathcal{EL}_{\mathrm{gfp}}$-GCIs $\mathcal{B}$ is a* basis for the $\mathcal{EL}_{\mathrm{gfp}}$-GCIs holding in i *if $\mathcal{B}$ is*

– sound *for i, i.e., it contains only $\mathcal{EL}_{\mathrm{gfp}}$-GCIs holding in i, and*
– complete *for i, i.e., any $\mathcal{EL}_{\mathrm{gfp}}$-GCI that holds in i follows from $\mathcal{B}$.*

The following lemma, taken from [5], shows that GCIs of the form $C \rightarrow C^{ii}$ play a special rôle.

[1] Note that this is not true if we use $\mathcal{EL}$ instead of $\mathcal{EL}_{\mathrm{gfp}}$ (see [5] for an example).

[2] GCI is an abbreviation for "general concept inclusion." In DL, GCIs are usually written as $C \sqsubseteq D$. Here, we prefer to use the arrow notation to emphasize the connection to implications in FCA and to avoid confusion with subsumption statements.

Lemma 2. *Let C, D be $\mathcal{EL}_{\text{gfp}}$-concept descriptions and i a finite $\mathcal{EL}_{\text{gfp}}$-model. Then*

- *$C \to C^{ii}$ holds in i, and*
- *if $C \to D$ holds in i, then $C \to D$ follows from $\{C \to C^{ii}\}$.*

This lemma reinforces the similarity between the $\cdot'$ operators from FCA and our $\cdot^i$ operators. In fact, in FCA a basis of all implications holding in a finite context can be obtained by taking all implications $P \to P''$ where P is a so-called pseudo-intent of the context (see Section 3 below). Following the lead of FCA, we thus need to determine which $\mathcal{EL}_{\text{gfp}}$-concept descriptions can play the rôle of pseudo-intents, i.e., we want to find a *finite* set Λ_i of left-hand sides for GCI such that the set of GCIs $C \to C^{ii}$ for $C \in \Lambda_i$ is a basis for the $\mathcal{EL}_{\text{gfp}}$-GCIs holding in i.

Before we can define such a set, we need to introduce one more notation. Given a finite set U of $\mathcal{EL}_{\text{gfp}}$-concept descriptions, $\bigsqcap U := \bigsqcap_{C \in U} C$ denotes their conjunction. The set Λ_i will be obtained as the set of all such conjunctions for subsets of a basic set M_i.

Definition 4. *Let i be a finite $\mathcal{EL}_{\text{gfp}}$-model. The sets M_i, Λ_i are defined as*

$$M_i := \mathcal{N}_{\text{prim}} \cup \{\exists r.X^i \mid r \in \mathcal{N}_r \text{ and } X \subseteq \Delta_i\} \quad \text{and} \quad \Lambda_i := \{\bigsqcap U \mid U \subseteq M_i\}.$$

Since $\mathcal{N}_{\text{prim}}$, $\mathcal{N}_r$, and Δ_i are finite, M_i and Λ_i are finite as well. Thus, the basis introduced in the next theorem is finite as well.

Theorem 2. *The set of GCIs $\mathcal{B}_i := \{C \to C^{ii} \mid C \in \Lambda_i\}$ is a finite basis for the $\mathcal{EL}_{\text{gfp}}$-GCIs holding in i.*

This basis actually differs from the one defined in [5]. However, the proof that this is indeed a basis for the $\mathcal{EL}_{\text{gfp}}$-GCIs holding in i is very similar to the one given in [5] for the basis introduced there.

The definition of $\mathcal{B}_i$ also provides us with a brute-force method for computing this basis. To compute M_i, all we have to do is consider the (finitely many) subsets X of Δ_i, and compute their most specific concepts. The set Λ_i is then obtained by considering all subsets of M_i, and $\mathcal{B}_i$ is obtained from the elements C of Λ_i by first computing their extensions in i, and then building the most specific concepts of these extensions.

This brute-force approach has two disadvantages. First, up to equivalence of $\mathcal{EL}_{\text{gfp}}$-concept descriptions, the set $\{X^i \mid X \subseteq \Delta_i\}$ may be considerably smaller than the powerset of Δ_i. In fact, not every subset of Δ_i needs to be an extension of an $\mathcal{EL}_{\text{gfp}}$-concept description, and thus different subsets of Δ_i may have the same most specific concept. Second, we also want to be able to deal with a situation where the model i is not explicitly given, but rather "known" to an expert. Similar to the case of attribute exploration in FCA, we then want to elicit enough information about i from the expert to be able to compute a basis, but without having to ask too many questions. In this situation, neither all subsets of Δ_i nor their most specific concepts can be assumed to be known/computable at the beginning of the exploration process.

In order to obtain a more practical algorithm for computing a basis, we will view the set M_i as the set of attributes in a classical formal context induced by the model i. In the next section, we define this induced context and state some interesting connections between the $\cdot'$ operations in this context and the $\cdot^i$ operations defined in the present section. Basically, we want to apply to the induced context the classical FCA algorithm for computing an implication basis. However, there are two differences compared to the classical case. First, we cannot assume that all the attributes (i.e., all the elements of M_i) are known from the beginning. Second, since our attributes are $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions, we can use the known subsumption algorithm for this DL [1] to obtain background knowledge about relationships between these attributes. Thus, we use an algorithm for computing an implication basis that can handle background knowledge [15], and extend it such that it can deal with a growing set of attributes.

3 Formal Concept Analysis

Because of space constraints, we cannot give an introduction into FCA here. We thus assume that the reader is familiar with basic notions such as formal contexts; attributes and objects; the $\cdot'$ operators; intents, extents, and pseudo-intents; and implications and implication bases (see, e.g., [10]). At several points in this paper we use the so-called *Next-Closure Algorithm*, which can also be found in [10]. Recall that a total order on a finite set of attributes M induces the so-called *lectic order*, which is a total order on the powerset of M. Given a set of attributes U and a set of implications $\mathcal{B}$, the Next-Closure Algorithm computes the lectically smallest set of attributes V that is closed with respect to $\mathcal{B}$ (i.e., respects all implications in $\mathcal{B}$) and lectically greater than U.

Background Knowledge and Growing Sets of Attributes

We adopt Stumme's approach for handling background knowledge [15], where the background knowledge is given by a set of implications holding in the context under consideration. We say that a set of implications $\mathcal{B}$ is an *implication basis for the context* $\mathbb{K}$ *w.r.t. the set of background implications* $\mathcal{S}$ if $\mathcal{B} \cup \mathcal{S}$ is a sound and complete set of implications for $\mathbb{K}$. As in the case without background knowledge, pseudo-intents provide us with the left-hand sides of such a basis. Given a set $\mathcal{S}$ of background implications, the notion of a pseudo-intent is extended as follows.

Definition 5. *Let (G, M, I) be a formal context and $\mathcal{S}$ a set of implications holding in (G, M, I). The set $P \subseteq M$ is called* $\mathcal{S}$-pseudo-intent *if P respects all implications in $\mathcal{S}$ and $Q'' \subseteq P$ holds for every $\mathcal{S}$-pseudo-intent $Q \subsetneq P$.*

Stumme shows that this notion of pseudo-intents yields a minimal implication basis w.r.t. the background knowledge. To be more precise, he proves that the following holds for the set of implications

$$\mathcal{B}_{\mathcal{S}} := \{P \to P'' \mid P \text{ is } \mathcal{S}\text{-pseudo-intent in } \mathbb{K}\}$$

Algorithm 1. Construction of an implication basis w.r.t. background knowledge for the case of a growing set of attributes

```
 1: Input: $\mathbb{K}_0 = (G, M_0, I_0)$, $\mathcal{S}_0$
 2: $\Pi_0 := \emptyset$, $P_0 := \emptyset$, $k := 0$
 3: while $P_k \neq \mathsf{null}$ do
 4:    $\Pi_{k+1} := \Pi_k \cup \{P_k\}$
 5:    $k := k + 1$
 6:    Input: $\mathbb{K}_k = (G, M_k, I_k)$, $\mathcal{S}_k$
 7:    if $M_k = M_{k-1} = P_k$ then
 8:       $P_k := \mathsf{null}$
 9:    else
10:       $P_k :=$ lectically smallest set of attributes that is
                – closed with respect to $\{P_j \to P_j^{\prime\prime_k} \mid P_j \in \Pi_k\}$ and $\mathcal{S}_k$, and
                – lectically larger than $P_{k-1}$.
11:    end if
12: end while
```

- $\mathcal{B}_\mathcal{S}$ is an implication basis for $\mathbb{K}$ w.r.t. $\mathcal{S}$, and
- $\mathcal{B}_\mathcal{S}$ has minimal cardinality among all implication bases for $\mathbb{K}$ w.r.t. $\mathcal{S}$.

Algorithm 1 looks at a setting where the set of objects is fixed, while the set of attributes as well as the background knowledge can grow. It starts with a context $\mathbb{K}_0 = (G, M_0, I_0)$ and a set of background implications $\mathcal{S}_0$ that hold in $\mathbb{K}_0$. In each step, new attributes and new background implications may be added by the user, thus yielding a new context $\mathbb{K}_k = (G, M_k, I_k)$ and an new implication set $\mathcal{S}_k$. We require for all $k \geq 1$ that (i) $M_{k-1} \subseteq M_k$; (ii) I_k agrees with I_{k-1} on M_{k-1}, i.e., for all $g \in G$ and for all $m \in M_{k-1}$ we have $(g, m) \in I_k$ iff $(g, m) \in I_{k-1}$; (iii) $\mathcal{S}_{k-1} \subseteq \mathcal{S}_k$; (iv) the implications of $\mathcal{S}_k$ hold in $\mathbb{K}_k$. The Next-Closure Algorithm used in line 10 of the algorithm requires a total order on the set of attributes. We assume that the total order on M_k extends the one on M_{k-1} such that $a < b$ for all $a \in M_{k-1}$ and $b \in M_k \setminus M_{k-1}$. To make clear which context we are referring to when using the prime operators, we add the index of the context; e.g., $A^{\prime\prime_k}$ is used to denote the set obtained from A by applying the prime operator of the context $\mathbb{K}_k$ twice.

It is easy to see that Algorithm 1 terminates if, and only if, from some point on the set of attributes is no longer extended. Now, assume that the algorithm has terminated after the n-th step. We want to show that the set of implications

$$\mathcal{B}_{\mathcal{S}_n}^{(n)} := \{P_j \to P_j^{\prime\prime_n} \mid P_j \in \Pi_n\}$$

is an implication basis for the final context $\mathbb{K}_n$ w.r.t. the final set of background implications $\mathcal{S}_n$. To prove this, we first need to show that the set of left-hand sides Π_n "covers" all the quasi-closed sets of attributes for $\mathbb{K}_n$. A set of attributes U is called *quasi-closed* for a context $\mathbb{K}$ iff, for all subsets $V \subseteq U$, it holds that either $V'' \subseteq U$ or $V'' = U''$.

Lemma 3. *If Q is a set of attributes that is quasi-closed for $\mathbb{K}_n$ and respects all the background implications in $\mathcal{S}_n$, then there is some $P \in \Pi_n$ such that $P \subseteq Q$ and $P''^{n} = Q''^{n}$.*

It is a well-known fact that all pseudo-intents are quasi-closed [8]. Likewise, we can show that all $\mathcal{S}_n$-pseudo-intents are quasi-closed for $\mathbb{K}_n$ [4]. In addition, $\mathcal{S}_n$-pseudo-intents by definition respect all implications of $\mathcal{S}_n$. Thus, Stumme's result implies completeness of $\{Q \rightarrow Q''^{n} \mid Q \text{ is quasi-closed in } \mathbb{K}_n \text{ and respects all implications of } \mathcal{S}\} \cup \mathcal{S}$. Obviously, if $P \subseteq Q$ and $P''^{n} = Q''^{n}$, then the implication $P \rightarrow P''^{n}$ has the implication $Q \rightarrow Q''^{n}$ as a consequence. Thus, Lemma 3 yields completeness of $\{P \rightarrow P''^{n} \mid P \in \Pi_n\} \cup \mathcal{S}$.[3]

Theorem 3. *Assume that Algorithm 1 has terminated after the n-th step. Then $\mathcal{B}^{(n)}_{\mathcal{S}_n}$ is an implication basis for $\mathbb{K}_n$ w.r.t. $\mathcal{S}_n$.*

Note that, in contrast to the case of fixed set of attributes, in step k we must add P_k to the set of left-hand sides even if P_k is an intent of $\mathbb{K}_k$, i.e., $P_k = P_k''^{k}$. This is so because it might happen that $P_k = P_k''^{k}$, but $P_k \neq P_k''^{n}$ because the attributes in $P_k''^{n} \setminus P_k''^{k}$ have only been added at a later point.

The Induced Context

What we call induced contexts in this work are formal contexts whose attributes are concept descriptions and whose set of objects is the domain of a finite model i. In such a context, an object x has an attribute C if x is in the extension of the concept C in the model i. Similar contexts have been introduced in [12,13]. In the following, we examine the connection between the $\cdot'$-operators in the induced context and the $\cdot^{i}$-operators in the model i. Induced contexts establish the connection between the DL world and the FCA world which we need for the algorithms introduced in the next section. But let us first give a more formal definition of the induced context for the cases of $\mathcal{EL}_{\mathrm{gfp}}$.[4]

Definition 6 (induced context). *Let i be a finite $\mathcal{EL}_{\mathrm{gfp}}$-model and M a finite set of $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions. The* context induced by M and i *is the formal context $\mathbb{K} = (G, M, I)$, where $G = \Delta_i$ and $I = \{(x, C) \mid C \in M \text{ and } x \in C^{i}\}$.*

In FCA, an object is in the extension of a set of attributes U iff it has all the attributes from U. In DL terms, this means that x is in the extension of the conjunction over all elements of U. Thus, the set of attributes $U \subseteq M$ corresponds to the concept $\bigsqcap_{C \in U} C$. In the other direction, we can approximate an arbitrary concept description C by the set of all attributes $D \in M$ that subsume C. Since M in general contains only a small number of concept descriptions, this is really

[3] Note that soundness is trivial since it is well-known that implications $P \rightarrow P''$ hold in the context that defines the prime operators used.

[4] Note, however, that the definitions and results given here do not really depend on $\mathcal{EL}_{\mathrm{gfp}}$. They hold for any concept description language in which the most specific concept exists.

just an approximation, i.e., the conjunction of these concepts D may strictly subsume C.

Definition 7. *Let $\mathbb{K}$ be the context induced by M and i, C an $\mathcal{EL}_{\mathrm{gfp}}$-concept description and $U \subseteq M$. We define* $\mathrm{pr}_{\mathbb{K}}(C) := \{D \in M \mid C \sqsubseteq D\}$, *and call this the* projection of C to $\mathbb{K}$. *Conversely, we define $\bigsqcap U := \bigsqcap_{D \in U} D$, and call this the* concept defined by U. *We say that C* can be expressed in terms of M *iff there is some $V \subseteq M$ such that $C \equiv \bigsqcap V$.*

As an immediate consequence of this definition we obtain that the mappings $C \mapsto \mathrm{pr}_{\mathbb{K}}(C)$ and $U \mapsto \bigsqcap U$ are antitonic:

- $C \sqsubseteq D$ implies $\mathrm{pr}_{\mathbb{K}}(D) \subseteq \mathrm{pr}_{\mathbb{K}}(C)$,
- $U \subseteq V$ implies $\bigsqcap V \sqsubseteq \bigsqcap U$.

In general, not all $\mathcal{EL}_{\mathrm{gfp}}$-concept descriptions can be expressed in terms of M. Therefore, it is quite obvious that information is lost when we make the transformation from a concept description to the corresponding attribute set and back. This is the reason why, in the following lemma, we only have subsumption and subset relationships rather than equivalence and equality relationships.

Lemma 4. *Let $\mathbb{K}$ be the context induced by M and i, C an $\mathcal{EL}_{\mathrm{gfp}}$-concept description, and $U \subseteq M$. Then the following statements hold:*

1. $C \sqsubseteq \bigsqcap \mathrm{pr}_{\mathbb{K}}(C)$
2. $\mathrm{pr}_{\mathbb{K}}(C)'' \subseteq \mathrm{pr}_{\mathbb{K}}\left(C^{ii}\right)$
3. $U \subseteq \mathrm{pr}_{\mathbb{K}}(\bigsqcap U)$
4. $(\bigsqcap U)^{ii} \sqsubseteq \bigsqcap U''$

If a concept description is expressible in terms of M, then no information is lost by the conversion to the corresponding attribute set. This is the reason why, under additional expressibility conditions, the subsumption and subset relationships of the above lemma can be turned into equivalence and equality relationships.

Lemma 5. *Let C be an $\mathcal{EL}_{\mathrm{gfp}}$-concept description and $U \subseteq M$ a set of attributes such that both C and $(\bigsqcap U)^{ii}$ can be expressed in terms of M. Then the following statements hold:*

1. $C \equiv \bigsqcap \mathrm{pr}_{\mathbb{K}}(C)$
2. $\mathrm{pr}_{\mathbb{K}}\left(C^{ii}\right) = \mathrm{pr}_{\mathbb{K}}(C)''$
3. $\bigsqcap U'' \equiv (\bigsqcap U)^{ii}$

4 Computing a Basis for the $\mathcal{EL}_{\mathrm{gfp}}$-GCIs Holding in a Finite $\mathcal{EL}_{\mathrm{gfp}}$-Model

First, we consider the case where the finite model i is given right from the beginning. In this case, we basically apply Algorithm 1 to the context induced by M_i (see Definition 4) and i. In a second step, we extend the algorithm obtained this way to a model exploration algorithm, which can deal with the case where the model i is not explicitly given, but rather "known" to an expert.

Algorithm 2. Computing a basis for an a priori given model i

```
Input: finite model $i = (\Delta_i, \cdot^i)$
$M_0 := \mathcal{N}_{\text{prim}}$, $\mathbb{K}_0 :=$ the context induced by $M_0$ and $i$, $\mathcal{S}_0 := \emptyset$
$\Pi_0 := \emptyset$, $P_0 := \emptyset$, $k := 0$
while $P_k \neq \mathsf{null}$ do
    $\Pi_{k+1} := \Pi_k \cup \{P_k\}$
    $M_{k+1} := M_k \cup \{\exists r.(\bigsqcap P_k)^{ii} \mid r \in \mathcal{N}_r\}$
    $\mathcal{S}_{k+1} := \{\{C\} \rightarrow \{D\} \mid C, D \in M_k, C \sqsubseteq D\}$
    $k := k + 1$
    if $M_k = M_{k-1} = P_k$ then
        $P_k := \mathsf{null}$
    else
        $P_k :=$ lectically next set of attributes that respects all implications in
                $\{P_j \rightarrow P_j''^k \mid 1 \leq j < k\}$ and $\mathcal{S}_k$
    end if
end while
```

The Case of an A Priori Given Model

Let i be a finite $\mathcal{EL}_{\text{gfp}}$-model. Recall that the basis $\mathcal{B}_i$ introduced in Section 2 is the set of all implications $C \rightarrow C^{ii}$ where the left-hand sides C are of the form $C = \bigsqcap U$ for some subset U of

$$M_i = \mathcal{N}_{\text{prim}} \cup \{\exists r.X^i \mid r \in \mathcal{N}_r \text{ and } X \subseteq \Delta_i\}.$$

Therefore, it is natural to look at the induced context for the attribute set M_i. The elements of M_i are $\mathcal{EL}_{\text{gfp}}$-concept descriptions, and thus there may be subsumption relationships between them, which can be computed using the known polynomial-time subsumption algorithm for $\mathcal{EL}_{\text{gfp}}$ [1]. We will use these subsumption relationships as background knowledge. Obviously, if $C \sqsubseteq D$ for $\mathcal{EL}_{\text{gfp}}$-concept descriptions $C, D \in M_i$, then the GCI $C \rightarrow D$ holds in i, and thus the implication $\{C\} \rightarrow \{D\}$ holds in the context induced by M_i and i.

Since Algorithm 1 allows for a growing set of attributes, we do not start with the whole set M_i. Instead, we start with the set $\mathcal{N}_{\text{prim}}$ of primitive concepts, and then extend the current set of attributes by adding $\mathcal{EL}_{\text{gfp}}$-concept descriptions of the form $\exists r.X^i$ whenever a new set of objects X is obtained as the extension of a concept $\bigsqcap P$ for an already computed left-hand side P. Algorithm 2 shows the instance of Algorithm 1 obtained this way.

Algorithm 2 always terminates since there are only finitely many attributes that can be added. In fact, every attribute that is added is an element of M_i, and we have already shown in Section 2 that M_i is finite. Now, assume that Algorithm 2 has terminated after the nth step. Then the algorithm has generated a set Π_n of subsets of $M_n \subseteq M_i$. This set Π_n gives rise to the following set of GCIs:

$$\mathcal{B}_n := \{\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{ii} \mid P_k \in \Pi_n\}.$$

Theorem 4. *Assume that Algorithm 2 terminates after the n-th step. Then $\mathcal{B}_n$ is a finite basis for the $\mathcal{EL}_{\text{gfp}}$-GCIs holding in i.*

Outline of the proof: Obviously, $\mathcal{B}_n$ is finite. In addition, since $\mathcal{B}_n$ is a subset of $\mathcal{B}_i$, we know that it is sound. Thus, to show that $\mathcal{B}_n$ is a finite basis for the $\mathcal{EL}_{\mathrm{gfp}}$-GCIs holding in i, it is enough to show completeness, i.e., any $\mathcal{EL}_{\mathrm{gfp}}$-GCI that holds in i follows from $\mathcal{B}_n$. Completeness can be proved in two steps. The first step is to show that, up to equivalence, M_n contains all attributes of the form $\exists r.X^i$ for $X \subseteq \Delta_i$. The second step then uses this fact to actually prove completeness of $\mathcal{B}_n$. *Step 1* is again divided into two parts.

(a) For a set of attributes $U \subseteq M_n$, we consider its closure U''^{n} under the double-prime operator $\cdot''^{n}$ of the context $\mathbb{K}_n$. As an intent of $\mathbb{K}_n$, U''^{n} is closed under $\cdot''^{n}$, and it respects any implication that holds in $\mathbb{K}_n$. Hence it is quasi-closed and respects all the implications of $\mathcal{S}_n$. Therefore, Lemma 3 ensures that there is some $P_k \in \Pi_n$ such that $P_k \subseteq U''^{n}$ and $P_k''^{n} = U''^{n}$. After the k-th step of the algorithm, all attributes of the form $\exists r.(\bigsqcap P_l)^{ii}$, where $0 \leq l \leq k$, have been added to the set of attributes. Using Lemma 4 and 5, it is possible to prove that $(\bigsqcap P_k)^{ii} \equiv (\bigsqcap U)^{ii}$ (see [4] for details). This shows that, up to equivalence, for every set $U \subseteq M_n$ the descriptions $\exists r.(\bigsqcap U)^{ii}$ must be in M_n.

(b) The fact that M_n contains all attributes of the form $\exists r.X^i$ for $X \subseteq \Delta_i$ can now be proved by induction on the depth of X^i, where we say that X^i has depth d iff d is the least role depth of $\mathcal{EL}$-concept descriptions D such that $X^i = D^{ii}$. In [5] it is shown that this notion of a depth is indeed well-defined. The base case is easy. In fact, if X^i has depth 0, then it can be written as conjunction of primitive concepts, i.e., $X^i = (\bigsqcap U)^{ii}$ for $U \subseteq M_0 \subseteq M_n$. But then it follows from (a) that M_n contains an attribute that is equivalent to $\exists r.(\bigsqcap U)^{ii} = \exists r.X^i$. The step case is very similar, except that one has to show that every X^i of role depth d can be written as the conjunction of primitive concept names and concept descriptions of the form $\exists r.Y^i$ where Y^i has depth less than d (details can be found in [4]).

Step 2. By Theorem 3, we know that the set $\mathcal{S} \cup \{P \to P''^{n} \mid P \in \Pi_n\}$ is a basis for the implications in $\mathbb{K}_n$. Let $L \in \Lambda_i$ be a premise of some implication from the basis $\mathcal{B}_i$ that is not an intent w.r.t. i, i.e., $L \not\equiv L^{ii}$. We can show that not only L, but also L^{ii} belongs to Λ_i, and thus both can be expressed in terms of M_n, as shown in Step 1. Lemma 4 can be used to derive $\mathrm{pr}_{\mathbb{K}_n}(L) \neq \mathrm{pr}_{\mathbb{K}_n}\left(L^{ii}\right) = \mathrm{pr}_{\mathbb{K}_n}(L)''^{n}$. Consequently, $\mathrm{pr}_{\mathbb{K}_n}(L)$ is not an intent of $\mathbb{K}_n$, and hence there must be an implication $P_k \to P_k''^{n}$ for $P_k \in \Pi_n$ that $\mathrm{pr}_{\mathbb{K}_n}(L)$ does not respect, i.e., $P_k \subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$, but $P_k''^{n} \not\subseteq \mathrm{pr}_{\mathbb{K}_n}(L)$. But then Lemma 4 implies that $L \sqsubseteq \bigsqcap P_k$, but $L \not\sqsubseteq (\bigsqcap P_k)^{ii}$.

Thus, for every concept description $L \in \Lambda_i$ that is not an intent w.r.t. i, there is some $P_k \in \Pi_n$ such that $L \sqsubseteq \bigsqcap P_k$, but $L \not\sqsubseteq (\bigsqcap P_k)^{ii}$. Since $\bigsqcap P_k \to (\bigsqcap P_k)^{ii}$ belongs to $\mathcal{B}_n$, the GCI $L \to L \sqcap (\bigsqcap P_k)^{ii}$ follows from $\mathcal{B}_n$. Since $L \not\sqsubseteq (\bigsqcap P_k)^{ii}$, the concept description $L \sqcap (\bigsqcap P_k)^{ii}$ is strictly subsumed by L, and it can be shown that $L \sqcap (\bigsqcap P_k)^{ii} \in \Lambda_i$. If $L \sqcap (\bigsqcap P_k)^{ii}$ is not an intent, then we can use the same argument, and find $P_l \in \Pi_n$ such that $L \sqcap (\bigsqcap P_k)^{ii} \to L \sqcap (\bigsqcap P_k)^{ii} \sqcap (\bigsqcap P_l)^{ii}$ follows from $\mathcal{B}_n$ and $L \sqcap (\bigsqcap P_k)^{ii} \sqcap (\bigsqcap P_l)^{ii}$ belongs to Λ_i and is strictly subsumed by $L \sqcap (\bigsqcap P_k)^{ii}$, etc. Since Λ_i is finite, this cannot go on forever, and thus we must reach an intent, which can actually be shown to be equal to L^{ii} (see [4] for more details). The whole chain of implications thus implies the single implication $L \rightarrow L^{ii}$. This proves that all implications from $\mathcal{B}_i$ follow from $\mathcal{B}_n$. Because $\mathcal{B}_i$ is complete, $\mathcal{B}_n$ is also complete. □

The Exploration Algorithm

Now, we extend Algorithm 2 to a model exploration algorithm, which can deal with the case where the finite model i (called *background model* in the following) is not explicitly given, but rather "known" to an expert. We assume that, at the beginning of the exploration process, only some "parts" of the model i are given to the exploration algorithm as *working model* i_0. In the following, we assume that the model i_0 as well as its extensions i_j generated during the exploration process are *connected submodels* of i, i.e., we have $\Delta_{i_0} \subseteq \Delta_i$, $x \in A^{i_0}$ iff $x \in A^i$ for all $A \in \mathcal{N}_{\text{prim}}$ and all $x \in \Delta_{i_0}$, and Δ_{i_0} is closed under i-role successors: if $x \in \Delta_{i_0}$ and $(x, y) \in r^i$ for a role r, then $y \in \Delta_{i_0}$ and $(x, y) \in r^{i_0}$. It is easy to see that this implies $x \in C^{i_0}$ iff $x \in C^i$ for all $\mathcal{EL}_{\text{gfp}}$-concept descriptions C and all $x \in \Delta_{i_0}$.

Algorithm 3 describes our model exploration algorithm. The modification with respect to Algorithm 2 merely consists of adding a second while-loop to the algorithm. Intuitively, this loop is used to determine the proper conclusion $(\bigsqcap P_k)^{ii}$ for a given premise $\bigsqcap P_k$. Since i is not explicitly given, $(\bigsqcap P_k)^{ii}$ cannot be computed directly, but only by interacting with the expert. This is done in the following way. The implication $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j}$ is presented to the expert. If the expert refutes the implication (i.e., says that it does not hold) then she is required to provide a counter-example, i.e., a connected submodel i_{j+1} of i that extends i_j (i.e., satisfies $\Delta_{i_j} \subseteq \Delta_{i_{j+1}}$). This is repeated until the expert states that $\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_j i_j}$ holds in i.

Since the set M_i is finite, only finitely many attributes can be added by Algorithm 3. Therefore, the outer while-loop can only be entered a finite number of times. With every pass of the inner while-loop, the working model is extended. Since the working models are submodels of the finite background model, this can only happen a finite number of times. This shows that Algorithm 3 terminates after a finite number of steps. Soundness and completeness of Algorithm 3 are easy consequences of soundness and completeness of Algorithm 2.

Theorem 5. *Assume that Algorithm 3 terminates after the n-th iteration of the outer while loop and that i_ℓ is the actual working model. Then $\{\bigsqcap P_k \rightarrow (\bigsqcap P_k)^{i_\ell i_\ell} \mid P_k \in \Pi_n\}$ is a finite basis for the $\mathcal{EL}_{\text{gfp}}$-GCIs holding in i.*

An Example

We illustrate Algorithm 2 using the example from the introduction. The domain of the background model thus consists of six persons: John, Michelle and their daughter Mackenzie, as well as Paul, Linda and their son James.[5] As primitive

[5] Since this is a very simple model, it satisfies GCIs not holding in the "real world."

Algorithm 3. The model exploration algorithm

```
Input: working model $i_0$ (connected submodel of the finite background model $i$)
$M_0 := \mathcal{N}_{\text{prim}}$, $\mathbb{K}_0 :=$ the context induced by $M_0$ and $i_0$, $\mathcal{S}_0 := \emptyset$
$\Pi_0 := \emptyset$, $P_0 := \emptyset$, $k := 0$, $j := 0$
while $P_k \neq \mathsf{null}$ do
    while expert refutes $\bigsqcap P_k \to (\bigsqcap P_k)^{i_j i_j}$ do
        $j := j + 1$
        Ask the expert for a new working model $i_j$ that extends $i_{j-1}$, is a connected
          submodel of $i$, and contains a counterexample for $\bigsqcap P_k \to (\bigsqcap P_k)^{i_{j-1} i_{j-1}}$
    end while
    $\Pi_{k+1} := \Pi_k \cup \{P_k\}$
    $M_{k+1} := M_k \cup \{\exists r.(\bigsqcap P_k)^{i_j i_j} \mid r \in \mathcal{N}_r\}$
    $\mathcal{S}_{k+1} := \{\{C\} \to \{D\} \mid C, D \in M_k, C \sqsubseteq D\}$
    $k := k + 1$
    if $M_k = M_{k-1} = P_k$ then
        $P_k := \mathsf{null}$
    else
        $P_k :=$ lectically next set of attributes that respects all implications in
                 $\{P_l \to P_l''^k \mid 1 \leq l < k\}$ and $\mathcal{S}_k$
    end if
end while
```

concepts we use *Male* (M), *Female* (F), *Father* (Ft) and *Mother* (Mt), and as role *child* (c). Let us assume that the initial working model i_0 contains only the first family, i.e., Δ_{i_0} consists of John, Michelle, and Mackenzie, and we have

$$\begin{array}{ll} M^{i_0} = Ft^{i_0} = \{\text{John}\}, & Mt^{i_0} = \{\text{Michelle}\}, \\ F^{i_0} = \{\text{Michelle}, \text{Mackenzie}\}, & c^{i_0} = \{(\text{Michelle}, \text{Mackenzie}), (\text{John}, \text{Mackenzie})\}. \end{array}$$

1st Iteration: The algorithm starts with $P_0 = \emptyset$. We have $\bigsqcap P_0 = \top$ and $\top^{i_0 i_0} = \top$, and thus the expert is asked whether the GCI $\top \to \top$ holds in i. Obviously, the answer must be "yes," and we continue by computing the new set of attributes M_1 by adding $\exists c.\top$ to $M_0 = \mathcal{N}_{\text{prim}}$. The induced context $\mathbb{K}_1$ obtained this way is

	Ft	M	Mt	F	$\exists c.\top$
John	X	X			X
Michelle			X	X	X
Mackenzie			X		

where we assume that the elements of M_1 are ordered as listed in the table.

2nd Iteration: The lectically next set that is closed with respect to $\{\emptyset \to \emptyset''^1\} = \{\emptyset \to \emptyset\}$ is $\{Ft\}$. We have $Ft^{i_0 i_0} = \{\text{John}\}^{i_0} = Ft \sqcap M \sqcap \exists c.F$, which gives rise to the GCI $Ft \to Ft \sqcap M \sqcap \exists c.F$. Thus, the expert is presented with the question: "Is it true that every father is male and has a child that is female?". This is not true in the background model i since Paul is a father without daughter. The expert refutes the GCI by adding Paul as a counterexample. Note that she must also add James, because the new working model i_1 must be a connected submodel of i. Based on this model, the algorithm computes a new right-hand-side for the GCI: $Ft^{i_1 i_1} =$

$Ft \sqcap M \sqcap \exists c.\top$. The new GCI $Ft \rightarrow Ft \sqcap M \sqcap \exists c.\top$ is presented to the expert, who accepts it. Consequently, the new attribute $\exists c.(Ft \sqcap M \sqcap \exists c.\top)$ is added.

We do not look at the *next iterations* in as much detail as for the first two. The following GCIs are found:

1. $Mt \rightarrow Mt \sqcap F \sqcap \exists c.F$ (Refuted, Linda added as counterexample)
2. $Mt \rightarrow Mt \sqcap F \sqcap \exists c.\top$ (Accepted)
3. $F \sqcap M \rightarrow Aa$ (Accepted)
4. $\exists c.\top \sqcap M \rightarrow Ft \sqcap M \sqcap \exists c.\top$ (Accepted)
5. $\exists c.\top \sqcap F \rightarrow Mt \sqcap F \sqcap \exists c.\top$ (Accepted)
6. $\exists c.M \sqcap \exists c.F \rightarrow Aa$ (Accepted)
7. $\exists c.\exists c.\top \rightarrow Aa$ (Accepted)

Here Aa ("all attributes") stands for the cyclic $\mathcal{EL}_{\mathrm{gfp}}$-concept description $(\mathcal{T}, A)$ where $\mathcal{T} = \{A \equiv M \sqcap F \sqcap Mt \sqcap Ft \sqcap \exists c.A\}$. Note that Aa is subsumed by any $\mathcal{EL}_{\mathrm{gfp}}$-concept description that can be formulated using the primitive concepts M, F, Ft, Mt and the role c. As such, it is the best approximation of the bottom concept that $\mathcal{EL}_{\mathrm{gfp}}$ can come up with.

Interestingly, all the GCIs accepted during the exploration process, except for the last two (6. and 7.), hold in the "real world." The GCIs 6. and 7. are artefacts of the simple model i used for the exploration. They are due to the fact that, in i, there are no grandparents, and no one has both a son and a daughter.

5 Related and Future Work

The context induced by a finite model and a finite set of concept descriptions as attributes has been considered before (e.g., in [12,13]). However, since this previous work did not make use of the most specific concept, the authors could not show and utilize the connections between the $\cdot^i$ operators in the model and the $\cdot'$ operators in the induced context. The work whose objectives is closest to ours is [13],[6] where Rudolph considers attributes defined in the DL $\mathcal{FLE}$, which is more expressive than $\mathcal{EL}$. Given a finite $\mathcal{FLE}$-model, he considers an infinite family of induced contexts $\mathbb{K}_n$, where the finite attribute sets are obtained by considering all $\mathcal{FLE}$-concept descriptions (modulo equivalence) up to role depth n. He then applies classical attribute exploration to these induced contexts, in each step increasing the role depths until a certain termination condition applies. Rudolph shows that the implication bases of the contexts considered up to the last step contain enough information to decide, for any GCI between $\mathcal{FLE}$-concept descriptions, whether this GCI holds in the given model or not. However, these implication bases do not appear to yield a basis for all the GCIs holding in the given finite model, though it might be possible to modify Rudolph's approach such that it produces a basis in our sense. The main problem with this approach is, however, that the number of attributes grows very fast when the role depth grows (this number increases at least by one exponential in each step). In contrast

[6] see `http://relexo.ontoware.org/` for a tool that realizes this approach.

to considering all concept descriptions up to a certain role depth, our approach only adds an attribute of the form $\exists r.(\bigsqcap P)^{ii}$ if P has been generated as the left-hand side of a GCI in our basis.

The main topic for future research is to show that the approach for using attribute exploration to complete DL knowledge bases introduced in [6] can be extended to the model exploration algorithm introduced in this paper.

References

1. Baader, F.: Terminological cycles in a description logic with existential restrictions. In: Proc. of IJCAI 2003, pp. 325–330. Morgan Kaufmann, San Francisco (2003)
2. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ envelope. In: Proc. of ICJAI 2005, pp. 364–369. Morgan Kaufmann, San Francisco (2005)
3. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, Cambridge (2003)
4. Baader, F., Distel, F.: Exploring finite models in the description logic $\mathcal{EL}_{\text{gfp}}$. LTCS-Report 08-05, Chair for Automata Theory, TU Dresden (2008)
5. Baader, F., Distel, F.: A finite basis for the set of $\mathcal{EL}$-implications holding in a finite model. In: Medina, R., Obiedkov, S. (eds.) ICFCA 2008. LNCS (LNAI), vol. 4933, pp. 46–61. Springer, Heidelberg (2008)
6. Baader, F., Ganter, B., Sattler, U., Sertkaya, B.: Completing description logic knowledge bases using formal concept analysis. In: Proc. of IJCAI 2007. AAAI Press/The MIT Press (2007)
7. Baader, F., Lutz, C., Suntisrivaraporn, B.: CEL—a polynomial-time reasoner for life science ontologies. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 287–291. Springer, Heidelberg (2006)
8. Ganter, B.: Two basic algorithms in concept analysis. Preprint 831, Fachbereich Mathematik, TU Darmstadt, Darmstadt, Germany (1984)
9. Ganter, B.: Attribute exploration with background knowledge. Theoretical Computer Science 217(2), 215–233 (1999)
10. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, New York (1997)
11. Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From SHIQ and RDF to OWL: The making of a web ontology language. Journal of Web Semantics 1(1), 7–26 (2003)
12. Prediger, S.: Logical scaling in formal concept analysis. In: Delugach, H.S., Keeler, M.A., Searle, L., Lukose, D., Sowa, J.F. (eds.) ICCS 1997. LNCS, vol. 1257, pp. 332–341. Springer, Heidelberg (1997)
13. Rudolph, S.: Relational Exploration: Combining Description Logics and Formal Concept Analysis for Knowledge Specification. PhD thesis, Technische Universität Dresden (2006)
14. Spackman, K.A., Campbell, K.E., Cote, R.A.: SNOMED RT: A reference terminology for health care. J. of the American Medical Informatics Association, 640–644 (1997); Fall Symposium Supplement
15. Stumme, G.: Attribute exploration with background implications and exceptions. In: Bock, H.-H., Polasek, W. (eds.) Data Analysis and Information Systems, pp. 457–469. Springer, Berlin (1996)
16. The Gene Ontology Consortium. Gene Ontology: Tool for the unification of biology. Nature Genetics 25, 25–29 (2000)

Yet a Faster Algorithm for Building the Hasse Diagram of a Concept Lattice

Jaume Baixeries, Laszlo Szathmary, Petko Valtchev, and Robert Godin

Dépt. d'Informatique UQAM, C.P. 8888,
Succ. Centre-Ville, Montréal H3C 3P8, Canada
baixeries.jaume@uqam.ca, Szathmary.L@gmail.com, valtchev.petko@uqam.ca,
godin.robert@uqam.ca

Abstract. Formal concept analysis (FCA) is increasingly applied to data mining problems, essentially as a formal framework for mining reduced representations (bases) of target pattern families. Yet most of the FCA-based miners, closed pattern miners, would only extract the patterns themselves out of a dataset, whereas the generality order among patterns would be required for many bases. As a contribution to the topic of the (precedence) order computation on top of the set of closed patterns, we present a novel method that borrows its overall incremental approach from two algorithms in the literature. The claimed innovation consists of splitting the update of the precedence links into a large number of lower-cover list computations (as opposed to a single upper-cover list computation) that unfold simultaneously. The resulting method shows a good improvement with respect to its counterpart both on its theoretical complexity and on its practical performance. It is therefore a good starting point for the design of efficient and scalable precedence miners.

1 Introduction

Formal concept analysis (FCA) extracts knowledge from datasets represented as objects × attributes tables [4]. Concepts are key knowledge chunks that represent meaningful abstractions in the underlying domain. They are particularly useful when hierarchically ordered into the concept lattice as the structure can support various types of reasoning such as classification, clustering, implication discovery, etc. Yet the construction of the concept lattice is not a trivial problem, especially with large datasets. Indeed, the size of the lattice could grow exponentially with the number of data items, hence the need to design efficient algorithms for the task (see [3]). The existing ones may roughly be split into three categories with respect to the structure that is effectively output. Historically, the first algorithms looked at the set of all concepts [1] which was not provided with any particular structure. Later on, methods constructing the Hasse diagram of the lattice, i.e., the concept set plus the precedence relation of the lattice, have been designed [2,5]. Quite recently, the problem of ordering the set of concepts, i.e., extracting the set of precedence links out of them, has been tackled [17].

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 162–177, 2009.

The question is of both theoretical and practical significance. Indeed, on the one hand, efficient algorithms for the computation of the concept set exist which, if appropriately completed, could yield a good overall method for lattice construction. In particular, such methods can be easily crafted from the various frequent closure miners abounding in the data mining literature [9,10,12]. Yet some miners, while very efficient, are particularly hard to adapt to the simultaneous computation of the Hasse diagram. We tend to see this fact as a sufficient motivation for the design of algorithms dedicated to the latter task. On the other hand, FCA is increasingly used within the data mining community as a formal framework for the numerous reduced representations of patterns and associations. Hence the interest for the construction of the iceberg concept lattice, i.e., the ordered structure of all frequent intents. Once again, the existing closure miners only output the intents and could therefore benefit from a complementary procedure deriving precedence out of the intent set. Yet to be efficient, the target procedure should not perform operations that depend on the size of the object set.

Our current study is motivated by the need for efficient computation of the precedence links among intents without looking on the respective extents. It is a follow-up to a previous work of the third author [17] and a response to [7]. Both algorithms basically work in the same way, i.e., they use the same intuition and very similar supporting data structures. The initial approach, which boils down to an incremental top-down construction of the lattice/iceberg diagram, has been revisited and substantially improved, both on its theoretical and practical aspects. In particular, given a concept, the basic test is on its being a lower cover of specific concepts already integrated in the diagram (instead of looking for its upper covers among previously processed concepts). Moreover, a deeper insight into the concept neighborhoods within the lattice (which builds upon a property from [18]) is exploited to speedup the computation of precedence links. The overall precedence link discovery process thus unfolds gradually: at each iteration the lower cover lists of concepts already in the diagram are tentatively updated.

Our method is a clear improvement of the state of the art as it outperforms the reference algorithms. Indeed, the new approach of precedence calculations results in a lower worst-case complexity (a multiplicative factor drops out). This is empirically confirmed by the results of an experimental study involving a straightforward implementation of the method and a large set of typical datasets used in the pattern mining literature.

All in all, the contributions of the paper are three-fold. First, a handy property of neighborhoods in the Hasse diagram is proven which can be used in the design of further algorithms targeting the precedence links among concepts. Next, a concrete method exploiting that property is devised whose complexity shows a substantial improvement with respect to the reference method. Finally, an extensive empirical study on the practical performances of the new method as opposed to its counterpart is reported.

The paper is organized as follows. After a short overview of the relevant FCA notions and notations, we recall the reference method from [17] (Section 2). The structural results behind our approach are provided next (Section 3), followed by the presentation of the new method called **iPred** (Section 4). Experimental evaluations are also provided (Section 5). Finally, conclusion and future work directions are given (Section 6).

2 Previous Work

2.1 Notation

We depart from the usual FCA notation, in which we have a formal context $\mathbb{K}(G, M, I)$ plus the associated concept lattice. In this paper we focus on the lattice formed by the set of intents of a formal context, let this set be $\mathcal{C} \subseteq \wp(M)$. The lattice is this set plus an order on that set, this is: $\mathcal{L} = \langle \mathcal{C}, \leq_{\mathcal{L}} \rangle$, where $\leq_{\mathcal{L}} \subseteq \mathcal{C} \times \mathcal{C}$.

Since we are just focusing in the intents of the concept lattice, we have that the bottom of the lattice is M whereas the top is $\emptyset$, if we assume that the formal context is reduced. The **precedence** relation $\preceq$ between $c, \tilde{c} \in \mathcal{C}$ is such that $c \preceq \tilde{c}$ if and only if $\tilde{c} \subseteq c$. This relation works inverted w.r.t. the inclusion relation because of the orientation of the intents in the concept lattice.

Given two elements $c, \tilde{c} \subseteq \mathcal{C}$, the relation $\prec$ is that of being the immediate predecessor, this is, if $c \prec \tilde{c}$ we say that c is the immediate predecessor of $\tilde{c}$.

Property 1. If $c \prec \tilde{c}$, then $|c| > |\tilde{c}|$.

We also define the following metrics on the lattice $\mathcal{L}$: $\omega(\mathcal{L})$ is the **width** of $\mathcal{L}$, whereas $d(\mathcal{L})$ is the **maximal degree** of all the elements in $\mathcal{L}$.

For our convenience, we flatten the set notation when dealing with sets. Therefore, $\{ a, b, c \}$ becomes abc and $\{ \{ a, b \}, \{ b, c \} \}$ becomes $\{ ab, bc \}$.

2.2 Methods Computing the Precedence Order

The historically first algorithm to compute the set of concepts of a context, **Next-Closure** [1], does not provide the precedence of the elements in that context. Later algorithms, such as those due to Godin [5] and to Bordat [2], compute both concepts and precedence yet the two tasks are interleaved hence difficult to separate. This does not make good candidates to complete a frequent closed itemset (FCI) miner out of them.

The first algorithm dedicated to the computing of the precedence was published in [8] as a distinct and separable part of a complete algorithm for the construction of a family of open sets and its semi-lattice. Yet the corresponding method does not qualify for an efficient precedence miner either. Indeed, the core operation which, in FCA terms, corresponds to the computing of the upper covers of a concept, boils down to intersecting the concept intent with all intents of object that are not in the concept extent. Performing an operation a number of times that depends on the size of the object set in a context clearly hurts the scalability of a method and hence limits its data mining potential as in realistic settings the number of the objects is orders of magnitude higher than the number of attributes. Nothewortily, transposing the context matrix would not help here as the cost of the algoritm in [8] depends on both dimensions of the context. Thus, whenever one of these is huge, its computation performances will invariably suffer.

The method in [17] is, to the best of our knowledge, the first attempt to address the precedence computation problem with data mining concerns in mind. In fact, the method only considers the set of all (frequent) intents and organizes them into a graph representing the Hasse diagram of the (iceberg) lattice. To that end, the intents are processed sizewise while at each step, the current intent is integrated into the already constructed part of the lattice graph (an upper set thereof for that matters) by recognizing its upper covers among the vertices of that partial graph. More precisely, the target concepts are pinpointed among a larger set of candidates, themselves generated by intersecting the current concept intent with the intents of all the current minimal concepts of the partial graph (see next subsection). The method has been recently rediscovered (see [7]) yet this new version shows greatly improved practical performance.

The idea of computing the order among frequent closed itemsets (FCI, alias frequent concept intents) has made its way into the data mining literature. For instance, the **Charm-L** algorithm [13] tackles that composite problem and its performance is very satisfactory (see [16]). Yet **Charm-L**, like the aforementioned lattice algorithms mixes concept computing with precedence detection, hence it is not a good choice for a mere precedence miner to adjoin to an existing FCI miner.

Recently, an approach for computing the precedence link out of FCIs and frequent generators has been proposed (see [14]). The corresponding method, **Snow**, fully qualifies for the task of completing existing FCI miners in a generic way. This has been extensively argued on in [16]. Yet **Snow** comes with a price: the minimal generators of frequent intents must be known as well as their respective closures (among the frequent intents). Although this is often the case that FCI miners output frequent minimal generators as byproduct, such practice is not a must in the field, so an overhead for computing the generators and for associating them to their closures must be provided for.

In our current study, we push further the ideas from [17], i.e., computing the precedence incrementally by incorporating the current intent into the already constructed subgraph. The novelty is a completely reshuffled procedure for establishing the links among concepts. In what follows, we first present the original algorithm and then the new one. Their respective worst-case complexity functions are established to theoretically ground the claimed improvement: Here a whole factor from the original formula vanishes. This is experimentally confirmed by the results of our comparative study on the practical performances of both methods.

2.3 The BorderAlg Algorithm

We depart from the algorithms in [17] and in [7], which we generically call **BorderAlg**. In general terms, the approach of those algorithms is to find the upper cover of the elements that are in the lattice. In order to achieve that, the algorithm sorts all the elements in the lattice sizewise and then, it proceeds to process each element one by one. At a given point of the algorithm, the element that is to be processed is intersected with all the elements in the border. The **border** set is the set of maximal elements of the set of already processed elements. Those intersections form the **candidate** set of upper covers for the current element. It is clear that those elements in the candidate set exist in the lattice, because it is closed under intersection. However, not all of them may necessarily be immediate predecessors. In order to find those predecessors, the maximal elements of this candidate set must be found, which we call the **cover** set. This is the set of upper covers for the current element. The algorithm proceeds to add the connections between the current element and all the elements in the cover set, updates the border set and proceeds with the next element.

Intuitively, we can see that for each element, we are sure that all those elements in the lattice that are of size strictly smaller (Property 1), have been processed. Since an element can only be in the upper cover if the size is strictly minor, we know that all the elements that potentially are immediate predecessors have already been processed. Some of them are in the border set, and the rest will result from the intersection with all the elements in cover.

```
  Input: $\mathcal{C} = \{ c_1, c_2, \ldots, c_l \}$
  Output: $\mathcal{L} = \langle \mathcal{C}, \leq_{\mathcal{L}} \rangle$
1 Sort($\mathcal{C}$);
2 Border $\leftarrow \{ c_1 \}$;
3 foreach $i \in \{ 2, l \}$ do
4     $Candidate \leftarrow \{ c_i \cap \tilde{c} \mid \tilde{c} \in Border \}$;
5     $Cover \leftarrow Maxima(Candidate)$;
6     $\leq_{\mathcal{L}} \leftarrow \leq_{\mathcal{L}} \cup \{ (c_i, \tilde{c}) \mid \tilde{c} \in Cover \}$;
7     $Border \leftarrow (Border - Cover) \cup c_i$;
8 end
```

Algorithm 1. The **BorderAlg** algorithm

To illustrate the **BorderAlg** algorithm, we have a formal context with its associate concept lattice (where only the intents are shown) in Figure 1. Let us assume that this algorithm has sorted the sets in the concept lattice (line 1) in the following way:

$$\{ \emptyset, c, d, a, bc, cd, de, abc, bcd, ade, cde \}$$

and that, at a certain point, all the elements up to de have been processed. Therefore, we have that the lattice has been constructed up to the iceberg in Figure 2.

	a	b	c	d	e
1	X	X	X		
2		X	X	X	
3	X			X	X
4			X	X	X
5	X				
6			X		

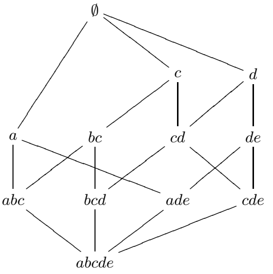


Fig. 1. Formal context and its concept lattice, in which only the intents are present

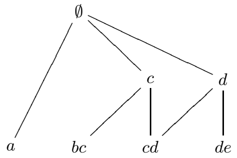


Fig. 2. Iceberg of the already processed elements

And that the next element to be processed is *abc*. The border set will therefore contain the elements a, bc, cd, de, and the candidate set computed in line 4 will yield the following set: $\{\, a, bc, c, \emptyset \,\}$, which is the intersection of *abc* with the border set. As it can be seen, the candidate set contains the elements of the upper cover of *abc* (i.e. $\{\, a, bc \,\}$) that are the immediate predecessors of *abc*, as well as other sets which are not in that upper cover: (i.e $\{\, c, \emptyset \,\}$). Precisely, the function $Maxima$ in line 5 computes the $Cover$ set from the $Candidate$ set. The resulting set is $\{\, a, bc \,\}$, and then, the connections (a, abc) and (bc, abc) are added to $\leq_{\mathcal{L}}$ in line 6, the sets a and bc are removed from the $Border$ set, and *abc* is added to that same set, so that the final $Border$ set after this iteration of the algorithm is cd, de, abc.

The complexity of **BorderAlg** is ([17]):

$$|\,\mathcal{C}\,| \times \omega(\mathcal{L}) \times |M|^2$$

More detailed information on the complexity analysis of **BorderAlg** can be found in Appendix A.

3 Theoretical Background of iPred

We have seen that the dominant factors in the complexity of Algorithm **BorderAlg** are the size of the input set $|\,\mathcal{C}\,|$ and the computation of the maximal elements of $Candidate$ (the set $Cover$) which is $\omega(\mathcal{L}) \times |M| \times d(\mathcal{L})$. Since the first factor can't be avoided, the efforts to reduce the complexity of this algorithm should be directed towards the reduction of the computation of the *Cover* set. In this paper we present our new algorithm, called **iPred**, which improves the second factor by assuming a new strategy.

If we focus on the second factor of complexity in Algorithm 1, we see that for any given element c_i we test which elements of the *Candidate* set also belong to the *Cover* set. This computation is performed by choosing the maximal elements of the *Candidate* set, since the following condition is met:

Property 2. An element $\tilde{c}$ is in the *Candidate* set of an element $c_i \in \mathcal{C}$ if and only if $c_i \preceq \tilde{c}$.

That means that all potential elements that can be immediate predecessors of c_i are all in the *Candidate* set. Therefore, the algorithm only needs to choose among the *Candidate* set all those elements that are **immediate** predecessors of c_i, i.e:

$$\{ \tilde{c} \in Candidate(c_i) \mid c_i \prec \tilde{c} \}$$

This is what the function *Maxima* in line 5 of Algorithm 1 does, since it chooses the maximal elements of the *Candidate* set.

The algorithm **iPred** that we propose in this paper takes a different strategy: instead of checking if an element $\tilde{c}$ of the *Candidate* set is in the upper cover of the element c_i which is currently being processed, we check if the element c_i belongs to the lower cover of $\tilde{c}$. Although in principle, it may seem that both strategies should yield a similar (not to say the same) complexity, we prove in this paper that **iPred** improves in a factor of $d(\mathcal{L})$ the complexity of previous algorithms. Since $|M|$ is an upper bound of $d(\mathcal{L})$, the improvement, in the best of the cases, is that of $|M|$.

In this section we present the mathematical background on which **iPred** is based. We define an enumeration of $\mathcal{C}$:

Definition 1. *An* ***enumeration*** *of* $\mathcal{C}$ *is the set:*

$$\text{enum}(\mathcal{C}) = \{ c_1, c_2, \ldots, c_n \}$$

such that $\forall i, j \leq n : i \leq j \implies |c_i| \leq |c_j|$.

An enumeration is simply a sizewise sorting of the elements of a set. We now define the **face** ([11]) and the **set of faces** of an element:

Definition 2. *The* face *of an element* $c \in \mathcal{C}$ *w.r.t. an immediate successor* $\tilde{c}$ *is the difference between those two sets. The* ***set of faces*** *is:*

$$faces(c) = \{ \tilde{c} - c \mid \tilde{c} \prec c \}$$

For instance, according to the concept lattice in Figure 1, the face of a w.r.t. abc is bc, and $faces(a) = \{ bc, de \}$. We now define a partial union of the faces of an element of the lattice:

Definition 3. *We define the* ***accumulation of faces*** *of an element* $c \in \mathcal{C}$ *w.r.t. an enumeration* $\text{enum}(\mathcal{C})$ *as:*

$$\Delta_c^i = \bigcup \{\, c_j - c \mid c_j \in \text{enum}(\mathcal{C}) \textit{ and } c_j \prec c \textit{ and } j < i \,\}$$

The accumulation of faces of $c \subseteq M$ up to i is simply the faces of c in the iceberg lattice formed by the elements of the enumeration $\text{enum}(\mathcal{C})$ up to i. Following Figure 1, we have that if the enumeration of the lattice is

$$\{\, \emptyset, c, d, a, bc, cd, de, abc, bcd, ade, cde \,\}$$

then, $\Delta_a^9 = bc$ and $\Delta_a^{11} = bcde$. This accumulation of faces is a handy way to test whether an element of the lattice is in the lower cover of another element of the lattice, as the next proposition shows:

Proposition 1. *$c_i \prec c$ if and only if $c_i \cap \Delta_c^i = \emptyset$ and $c_i \preceq c$.*

Proof. $\Rightarrow$ $c_i \cap \Delta_c^i = \emptyset$ and $c_i \preceq c$ implies that $c_i \prec c$. By the way of contradiction, let us assume that $c_i \not\prec c$. Since $c_i \preceq c$, there is a $\tilde{c}$ such that $c_i \preceq \tilde{c} \prec c$. Therefore, by Definition 1 of sequence, and by Definition 3 of accumulation of faces, $\tilde{c} - c \subseteq \Delta_c^i$, and since $c_i \preceq \tilde{c}$, then, $c_i \cap \Delta_c^i \neq \emptyset$, which is a contradiction.

$\Leftarrow$ $c_i \prec c$ implies that $c_i \cap \Delta_c^i = \emptyset$ and $c_i \preceq c$. If $c_i \prec c$, by Definition 1 of sequence, all the immediate predecessors of c of size smaller or equal than $|c_i|$ are incomparable with c_i, meaning that $c_i \cap \Delta_c^i = \emptyset$. □

This proposition basically states that, given an enumeration of the elements of a lattice, in order to know if between two elements $c, \tilde{c}$ of the lattice we have that $c \prec \tilde{c}$, we only need to test if the accumulation of faces of $\tilde{c}$ has an empty intersection with c.

In order to compute the connections in a lattice according to Proposition 1, the following must be performed:

1. Sort the elements of the lattice into an enumeration.
2. For each element in the lattice, the candidate set must be computed.
3. It must be checked if the element currently being processed belongs to the lower set of all the elements of the candidate set.

The first two steps are as in **BorderAlg**, and the difference appears in the third step. In order to test if the current element is in the lower cover of an element of the candidate set, we must compute the accumulation of faces of the latter. This step will reduce the complexity of **BorderAlg** in a factor of $|M|$ as we will see in the next section.

4 The iPred Algorithm

In this section we present our new algorithm called **iPred**, we examine its correctness and complexity, and we also provide a running example.

4.1 The Algorithm

The algorithm is based on Proposition 1, and it computes the sets *Border*, *Candidate* as in Algorithm 1, but the set *Cover* is not needed any more. There is an extra structure, Δ, in which we store the accumulation of faces for all the elements of the lattice, and we choose the notation $\Delta[c]$ to show the access to the accumulated faces of the set of attributes c. In terms of complexity, if this structure is implemented with a trie, the access to an element would be linear on the number of attributes: $|M|$. The **iPred** algorithm is as follows:

```
   Input: $\mathcal{C} = \{ c_1, c_2, \ldots, c_l \}$
   Output: $\mathcal{L} = \langle \mathcal{C}, \leq_{\mathcal{L}} \rangle$
1  Sort($\mathcal{C}$);
2  foreach $i \in \{ 2, l \}$ do
3      $\Delta[c_i] \leftarrow \emptyset$;
4  end
5  Border $\leftarrow \{ c_1 \}$;
6  foreach $i \in \{ 2, l \}$ do
7      $Candidate \leftarrow \{ c_i \cap \tilde{c} \mid \tilde{c} \in Border \}$;
8      foreach $\tilde{c} \in Candidate$ do
9          if $\Delta[\tilde{c}] \cap c_i = \emptyset$ then
10             $\leq_{\mathcal{L}} \leftarrow \leq_{\mathcal{L}} \cup (c_i, \tilde{c})$;
11             $\Delta[\tilde{c}] = \Delta[\tilde{c}] \cup (c_i - \tilde{c})$;
12             $Border \leftarrow Border - \tilde{c}$;
13         end
14     end
15     $Border \leftarrow Border \cup c_i$;
16 end
```

Algorithm 2. The **iPred** algorithm

The algorithm works as follows:

1. It sorts the elements of the lattice by size (line 1). This sequence is now an enumeration as in Definition 1.
2. All the $\Delta[c_i]$ in each element of the input set is initialized to the empty set. This $\Delta[c_i]$ will contain the accumulation of faces for each element (lines 2–4).
3. The first element in the border is the first element in the sequence (line 5).
4. All remaining elements in the input sequence are processed in the order in which they appear in the enumeration (lines 6–16).
5. The candidate set is computed by intersecting the current element c_i with all the elements in the border (line 7).
6. We check if the current element belongs to the upper set of the elements that are in the candidate set (lines 8–14). This is done by checking Proposition 1 (line 9).

7. If the test result is positive, by Proposition 1 we know that $c_i \prec \tilde{c}$, so we can add this connection to the output set (line 10), then we add that face to the set of accumulated faces of $\tilde{c}$ (line 11) and finally, we remove $\tilde{c}$ from the Border (line 12).
8. Before the next element is processed, we make sure that c_i is added to the border (line 15).

The correctness of this algorithm follows from the following facts:

1. $\mathcal{C}$ is a valid enumeration, according to Definition 1.
2. $\Delta[\tilde{c}]$ has at each step of the algorithm the accumulation of faces $\Delta^i_{\tilde{c}}$. At the beginning of the algorithm, for any element $\tilde{c}$ of the lattice, its accumulation is the empty set, this is $\Delta^0_{\tilde{c}}$. Every time a new element $\tilde{\tilde{c}}$ such that $\tilde{\tilde{c}} \prec \tilde{c}$ is found, $\Delta^i_{\tilde{c}}$ is updated conveniently in line 11, which guarantees that at the loop i of the algorithm, $\Delta[\tilde{c}] = \Delta^i_{\tilde{c}}$.
3. A connection is added if and only if the test in Proposition 1 is positive. It should be noted that since all the elements of the *Candidate* set are the intersection of c_i with the elements of the *Border* set, we are sure that, for each element $c \in \mathit{Candidate}$, $c \subseteq c_i$ (i.e. $c_i \preceq c$) holds, which is one of the conditions in Proposition 1.
4. *Border* always contains the maximal elements of the set of processed elements. This border is updated in line 12, where the element $\tilde{c}$ is removed. This is valid since we are sure that this can be done because c_i will be added to the border (line 15) and, at the same time, we know that $c_i \prec \tilde{c}$ since we are in the positive case of the test in line 9. If the test is negative, c_i is also added to the border, but no elements are removed.

Therefore, we conclude that Algorithm 2 finishes (since we assume that the set $|\,\mathcal{C}\,|$ is finite), and correctly computes the connections in the lattice set, since it correctly tests the condition in Proposition 1.

4.2 Complexity Analysis

The complexity of the previous algorithm is based on the following factors:

1. The sort in line 1 can be performed in linear time w.r.t. the size of the set, this is, $|\,\mathcal{C}\,| \times |M|$ (as in Algorithm 1). The cost of line 2 is exactly the same.
2. The loop in lines 6–16 is done $|\,\mathcal{C}\,|$ times (as in Algorithm 1).
3. The complexity of the computation of the candidate set in line 7 is $\omega(\mathcal{L}) \times |M|$ (as in Algorithm 1).
4. The loop in lines 8–14 is performed $\omega(\mathcal{L})$ times (as in Algorithm 1).
5. The cost of checking the condition in line 9 is $|M|$ if Δ is a trie.
6. The cost of line 10 is $|M|$ since it consists in adding a pair of size M to $\leq_{\mathcal{L}}$.
7. The cost of line 11 is $|M|$ because it consists in updating an element in a trie.
8. As for line 12, it consists in removing an element from a set. If this set is also implemented with a trie, then the cost is also $|M|$.

We condense the costs of lines 9, 10, 11 and 12 into $|M|$ and, therefore, the total cost of this algorithm is:

$$\text{line:}\quad \underbrace{|\mathcal{C}| \times |M|}_{1\text{–}2} + \underbrace{|\mathcal{C}|}_{6\text{–}16} \times \Big(\underbrace{\omega(\mathcal{L}) \times |M|}_{7} + \underbrace{\omega(\mathcal{L})}_{8\text{–}14} \times \underbrace{|M|}_{9\text{–}12} \Big)$$

Since the cost of line 7 ($\omega(\mathcal{L}) \times |M|$) subsumes the cost of lines 8–14 and 9–12 (it is just the same), and since the factor in lines 1–2 is subsumed by the rest of the formula, the complexity of Algorithm 2 is finally of order:

$$|\mathcal{C}| \times \omega(\mathcal{L}) \times |M|$$

Compared with the complexity of **BorderAlg**, we can see that the factor $|M|^2$ is now $|M|$, which means that we should expect an improvement of the performance of the algorithm by a factor linear on the size of the attribute set.

4.3 Running Example

Let us see how the algorithm would perform according to Figure 1. We list the following variables:

1. The element currently being processed (line 6).
2. The candidate set (line 7).
3. The output $\leq_{\mathcal{L}}$ (line 10).
4. The accumulation of faces, only for those that are changed (line 11).
5. The border (line 12).

At the beginning of the algorithm, the elements of the input set are sorted sizewise, let us assume that one of the possible orderings is:

$$\{\, \emptyset, c, d, a, bc, cd, de, abc, bcd, ade, cde \,\}$$

All the accumulation of faces are set to the empty set (lines 2 - 4) and Border has the first element of the sorted input sequence, this is $\emptyset$. We now list how the precedent sets change according to each loop in the algorithm.

1 Current element $:= c$
Candidate set $:= \{\, \emptyset \,\}$
$\leq_{\mathcal{L}}$: added $(\emptyset, c)$
$\mathit{\Delta}[\emptyset] = c$
Border $:= \{\, c \,\}$

2 Current element $:= d$
Candidate set $:= \{\, \emptyset \,\}$
$\leq_{\mathcal{L}}$: added $(\emptyset, d)$
$\mathit{\Delta}[\emptyset] = cd$
Border $:= \{\, c, d \,\}$

7 Current element $:= abc$
Candidate set $:= \{\, \emptyset, a, c, bc \,\}$
$\leq_{\mathcal{L}}$: added $(a, abc), (bc, abc)$
$\mathit{\Delta}[a] := bc$, $\mathit{\Delta}[bc] = a$
Border $:= \{\, cd, de, abc \,\}$

8 Current element $:= bcd$
Candidate set $:= \{\, \emptyset, d, bc, cd \,\}$
$\leq_{\mathcal{L}}$: added $(bc, bcd), (cd, bcd)$
$\mathit{\Delta}[bc] = ad$,$\mathit{\Delta}[cd] = b$
Border $:= \{\, de, abc, bcd \,\}$

3 Current element := a
Candidate set := $\{\, \emptyset \,\}$
$\leq_{\mathcal{L}}$: added $(\emptyset, a)$
$\Delta[\emptyset] = acd$
Border := $\{\, a, c, d \,\}$

4 Current element := bc
Candidate set := $\{\, \emptyset, c \,\}$
$\leq_{\mathcal{L}}$: added (c, bc),
$\Delta[c] = b$
Border := $\{\, a, bc, d \,\}$

5 Current element := cd
Candidate set := $\{\, \emptyset, c, d \,\}$
$\leq_{\mathcal{L}}$: added $(c, cd), (d, cd)$
$\Delta[c] := bd, \Delta[d] := c$
Border := $\{\, a, bc, cd \,\}$

6 Current element := $\{d, e\}$
Candidate set := $\{\, \emptyset, d \,\}$
$\leq_{\mathcal{L}}$: added (d, de)
$\Delta[d] := ce$
Border := $\{\, a, bc, cd, de \,\}$

9 Current element := ade
Candidate set := $\{\, \emptyset, a, d, de \,\}$
$\leq_{\mathcal{L}}$: added $(a, ade), (de, ade)$
$\Delta[a] = bcde$,$\Delta[de] = a$
Border := $\{\, abc, bcd, ade \,\}$

9 Current element := cde
Candidate set := $\{\, \emptyset, c, cd, de \,\}$
$\leq_{\mathcal{L}}$: added $(cd, cde), (de, cde)$
$\Delta[cd] = be$,$\Delta[de] = ac$
Border := $\{\, abc, bcd, ade, cde \,\}$

10 Current element := $abcde$
Candidate set := $\{\, abc, bcd, ade, cde \,\}$
$\leq_{\mathcal{L}}$: added $(abc, abcde)$,$(bcd, abcde)$
$(ade, abcde)$,$(cde, abcde)$
$\Delta[abc] = de$,$\Delta[bcd] = ae$
$\Delta[ade] = bc$,$\Delta[cde] = ab$
Border := $\{\, abcde \,\}$

As an example, let us see what happens in step 7. We add the element abc. The *Candidate* set is $\{\, \emptyset, a, c, bc \,\}$, which results from the intersection with the *Border* set, which is $\{\, a, bc, cd, de \,\}$. We point out the fact that the *Candidate* set contains the upper set of abc in the lattice $\mathcal{L}$. The accumulated faces of the elements of the *Candidate* set are then tested one by one with abc (line 9). The first intersection is with the accumulated face of $\emptyset$, which is acd according to its last update in loop 3. Since the intersection is not void, then, $\emptyset$ is not considered as an immediate predecessor of abc. As it has been previously explained, the reason is that the pairs $(\emptyset, a)$ and $(\emptyset, bc)$ have already been added to $\leq_{\mathcal{L}}$ and, therefore, the differences between those sets and $\emptyset$ is in $\Delta^7_{\emptyset}$. The next element to be checked is the accumulated faces of a, which is void and, hence, the intersection is also void. It means that $abc \prec a$, and the following operations are performed in lines 10–12: the pair (a, abc) is added to $\leq_{\mathcal{L}}$, and a is removed from the *Border* set. The accumulated faces of a is updated accordingly, and we have now that $\Delta[a] := bc$. We now check the intersection of accumulated faces of the next element, this is element$\Delta[c]$, which is bd, and abc. The intersection is not void, and therefore, c is not a predecessor of abc. The reason is that the element bc has already been processed and $\Delta[c]$ contains, at least, the attribute b, meaning that c is the predecessor of an element that is contained by abc. The final checking is between $\Delta[bc]$, which is empty, and abc. The intersection is obviously void, and the algorithm adds (bc, abc) to $\leq_{\mathcal{L}}$, deletes bc from the border and updates $\Delta[bc]$ which now is a.

Table 1. Top: database characteristics. **Bottom:** response times of **iPred**.

database name	# records	# non-empty attributes	# attributes (in average)	largest attribute
T20I6D100K	100,000	893	20	1,000
T25I10D10K	10,000	929	25	1,000
chess	3,196	75	37	75
connect	67,557	129	43	129
pumsb	49,046	2,113	74	7,116
MUSHROOMS	8,416	119	23	128
C20D10K	10,000	192	20	385
C73D10K	10,000	1,592	73	2,177

min_supp	# concepts (including top)	**BorderAlg**	**iPred**
T20I6D100K			
0.75%	4,711	2.29	2.07
0.50%	26,209	74.92	51.88
0.25%	149,218	2,930.29	1,941.07
T25I10D10K			
0.40%	83,063	978.58	707.75
0.30%	122,582	2,207.86	1,763.42
0.20%	184,301	5,155.20	4,740.87
chess			
65%	49,241	974.23	87.60
60%	98,393	4,320.81	374.26
55%	192,864	21,550.23	1,905.60
connect			
65%	49,707	1,331.96	78.15
60%	68,350	2,634.35	140.50
55%	94,917	5,349.14	262.15

min_supp	# concepts (including top)	**BorderAlg**	**iPred**
pumsb			
84%	11,443	614.85	57.80
82%	19,942	2,043.31	173.31
80%	33,296	6,270.42	471.66
MUSHROOMS			
20%	1,169	0.71	0.17
10%	4,850	7.31	1.31
5%	12,789	53.35	7.87
C20D10K			
0.60%	119,734	10,847.59	993.29
0.50%	132,952	13,784.71	1,328.33
0.40%	151,394	19,013.53	1,858.34
C73D10K			
70%	19,501	414.94	37.29
65%	47,491	2,864.02	226.80
60%	108,428	18,323.78	1,296.40

5 Experimental Results

The original **BorderAlg** and the improved **iPred** algorithms were implemented in Java in the CORON data mining platform [15].[1] The experiments were carried out on a bi-processor Intel Quad Core Xeon 2.33 GHz machine with 4 GB RAM running under Ubuntu GNU/Linux. All times reported are real, wall clock times.

For the experiments, we used several real and synthetic dataset benchmarks. Database characteristics are shown in Table 1 (top). The chess and connect datasets are derived from their respective game steps. The MUSHROOMS database describes mushrooms characteristics. These three datasets can be found in the UC Irvine Machine Learning Database Repository. The pumsb, C20D10K, and C73D10K datasets contain census data from the PUMS sample file. The synthetic datasets T20I6D100K and T25I10D10K, using the IBM Almaden generator, are constructed according to the properties of market basket data.

Table 1 (bottom left and right) provides a summary of the experimental results. The first column specifies the various minimum support values for each of the datasets (low for the sparse dataset, higher for dense ones), while the second column comprises the number of FCIs. The third and fourth columns compare the execution times of **BorderAlg** and **iPred** (given in seconds). The CPU time does not include the cost of computing FCIs since it is assumed as given.

[1] `http://coron.loria.fr`

As can be seen, the improved algorithm outperforms the original **BorderAlg** algorithm in all cases. In the case of sparse datasets (T20 and T25), the difference is not that spectacular. However, in the case of dense datasets, there is a significant difference between the two algorithms, especially at lower minimum support thresholds. This is due to the fact that **iPred** reduces the complexity of **BorderAlg** by a factor linear on the size of the attribute set (see Section 4.2). The experimental results prove that the practical performance of **iPred** reflects its better theoretical complexity.

6 Conclusion

We presented a novel method for computing the precedence order among concepts that only manipulates their intents and is therefore suitable for data mining applications. The method explores the basic fact that the faces of all lower covers of a given concept in the Hasse diagram are pair-wise disjoint. Hence, the incremental incorporation of concepts into the current diagram could be organized as a set of gradually unfolding lower-cover list completions. A completion step is executed upon the incorporation of a new intent into the diagram and boils down to testing its disjointness with the already recognized faces.

The new method has been shown to outperform the reference one both on its worst-case complexity (smaller by a multiplicative factor) and practical performances (speedup from 30 to 3 000 %, depending on dataset profile). The advantages of the new method are only starting to unravel as no particular speedup techniques have been employed in the current implementation. Thus, the next step would be to study the benefits of indexing on the border set to avoid unnecessary intersections with the current intent.

Another promising track seems to reside in the batch processing of the levels in the diagram, i.e., the set of intents of identical size. Another intriguing question is the performance of a modern FCI miner, such as **Charm** or **Closet**, completed with our method.

References

1. Ganter, B.: Two basic algorithms in concept analysis (preprint). Technical Report 831, Technische Hochschule, Darmstadt (1984)
2. Bordat, J.-P.: Calcul pratique du treillis de Galois d'une correspondance. Mathématiques et Sciences Humaines 96, 31–47 (1986)
3. Carpineto, C., Romano, G.: Concept Data Analysis: Theory and Applications. John Wiley & Sons, Ltd., Chichester (2004)
4. Ganter, B., Wille, R.: Formal Concept Analysis, Mathematical Foundations. Springer, Heidelberg (1999)
5. Godin, R., Missaoui, R.: An Incremental Concept Formation Approach for Learning from Databases. Theoretical Computer Science 133, 378–419 (1994)
6. Kryszkiewicz, M.: Concise Representation of Frequent Patterns Based on Disjunction-Free Generators. In: Proc. of the 2001 IEEE Intl. Conf. on Data Mining (ICDM 2001), Washington, DC, pp. 305–312. IEEE Computer Society Press, Los Alamitos (2001)

7. Martin, B., Eklund, P.W.: From Concepts to Concept Lattice: A Border Algorithm for Making Covers Explicit. In: Medina, R., Obiedkov, S. (eds.) ICFCA 2008. LNCS (LNAI), vol. 4933, pp. 78–89. Springer, Heidelberg (2008)
8. Nourine, L., Raynaud, O.: A fast algorithm for building lattices. Inf. Process. Lett. 71(5–6), 199–204 (1999)
9. 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, pp. 21–30 (2000)
10. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering Frequent Closed Itemsets for Association Rules. In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 398–416. Springer, Heidelberg (1998)
11. Pfaltz, J.L.: Incremental Transformation of Lattices: A Key to Effective Knowledge Discovery. In: Corradini, A., Ehrig, H., Kreowski, H.-J., Rozenberg, G. (eds.) ICGT 2002. LNCS, vol. 2505, pp. 351–362. Springer, Heidelberg (2002)
12. Zaki, M.J., Hsiao, C.-J.: ChARM: An Efficient Algorithm for Closed Itemset Mining. In: SIAM Intl. Conf. on Data Mining (SDM 2002), pp. 33–43 (April 2002)
13. 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), 462–478 (2005)
14. Szathmary, L., Valtchev, P., Napoli, A., Godin, R.: Constructing Iceberg Lattices from Frequent Closures Using Generators. In: Boulicaut, J.-F., Berthold, M.R., Horváth, T. (eds.) DS 2008. LNCS (LNAI), vol. 5255, pp. 136–147. Springer, Heidelberg (2008)
15. Szathmary, L.: Symbolic Data Mining Methods with the Coron Platform. PhD Thesis in Computer Science, Univ. Henri Poincaré – Nancy 1, France (November 2006)
16. Szathmary, L., Valtchev, P., Napoli, A., Godin, R.: A Modular Approach for Mining Iceberg Lattices with Generators. In: Proc. of the 9th SIAM Intl. Conf. on Data Mining (SDM 2009) (submitted) (2009)
17. Valtchev, P., Missaoui, R., Lebrun, P.: A Fast Algorithm for Building the Hasse Diagram of a Galois Lattice. In: Proc. of Colloque LaCIM 2000, Montreal, Canada, pp. 293–306 (2000)
18. Valtchev, P., Hacene, M.R., Missaoui, R.: A Generic Scheme for the Design of Efficient On-Line Algorithms for Lattices. In: Ganter, B., de Moor, A., Lex, W. (eds.) ICCS 2003. LNCS, vol. 2746, pp. 282–295. Springer, Heidelberg (2003)

A Complexity of the Border Algorithm

We analyze now the complexity of Algorithm 1, which is based on the following costs:

1. The sizewise sorting of the input set (line 1). This can be done in linear time w.r.t. the size of the set: $|\mathcal{C}| \times |M|$, since we can first scan the list and compute the amount of elements for each size, allocate the corresponding slot memory, and in a second scan, we can store each element according to its size.
2. The number of loops in lines 3-8 depends on the amount of elements to be processed, which is the size of the input set: $|\mathcal{C}|$.

3. The cost of computing the *Candidate* set depends on the size of the border, since each element of $\mathcal{L}$ will be intersected with that set (line 4). This size is, in the worst of the cases, the width of the lattice, and each intersection, in the worst case, can be done in time linear on $|M|$. The total cost for computing the *Candidate* set is $\omega(\mathcal{L}) \times |M|$.
4. The computation of the maxima of the set resulting from the intersection, this is, the *Cover* set in line 5,$\mathit{\Delta}[bc] = a$ is performed in $|M| \times \omega(\mathcal{L}) \times d(\mathcal{L})$. Details of this function and its cost can be found in [17].
5. The update of $\leq_{\mathcal{L}}$ with the new connections in line 6 depends on the maximal number of connections that any element may have, which is the maximal degree: $d(\mathcal{L})$.
6. The update of the border in line 7 can be neglected since it only consists in adding a pair in a set.

The total complexity of this algorithm is, therefore:

$$\text{line:}\quad \underbrace{|\,\mathcal{C}\,| \times |M|}_{1} + \underbrace{|\,\mathcal{C}\,|}_{3\text{-}8} \times \Big(\underbrace{\omega(\mathcal{L}) \times |M|}_{4} + \underbrace{|M| \times \omega(\mathcal{L}) \times d(\mathcal{L})}_{5} + \underbrace{d(\mathcal{L})}_{6} \Big)$$

Some factors are subsumed by others: the cost of computing the *Candidate* set $\omega(\mathcal{L}) \times |M|$ in line 4 and the cost $d(\mathcal{L})$ of updating $\leq_{\mathcal{L}}$ in line 6 are both subsumed by the factor $|M| \times \omega(\mathcal{L}) \times d(\mathcal{L})$ in line 5, and the additive factor $|\,\mathcal{C}\,| \times |M|$ in line 1 is also subsumed by the rest of the formula. Therefore, we have that the final complexity is:

$$|\,\mathcal{C}\,| \times \omega(\mathcal{L}) \times |M| \times d(\mathcal{L})$$

Since in the worst of the cases we have that $d(\mathcal{L}) = |M|$, then, we have:

$$|\,\mathcal{C}\,| \times \omega(\mathcal{L}) \times |M|^2.$$

Context Graphs — Representing Formal Concepts by Connected Subgraphs

Jens Kötters[1], Heinz Schmidt[2], and David McG. Squire[1]

[1] Monash University, Melbourne, Australia
[2] RMIT University, Melbourne, Australia

Abstract. The article introduces a representation of a formal context by an undirected graph called a *context graph* with the formal objects being the nodes of the graph. We use as a defining property for this graph that it contains every concept extent as a connected subgraph. The graph is not uniquely defined by this property — we focus on those graphs that are edge-minimal and present a result with respect to the number of their edges. We then study how the structure of an edge-minimal context graph can be updated to adjust to the subsequent addition of an object to the context. This leads to an incremental construction algorithm that does not require the explicit computation of formal concepts.

Keywords: Context Graphs, Formal Concept Analysis, Graph Theory, Information Retrieval, Navigation.

1 Introduction

Overview. A context graph is a structural representation of a formal context that can be seen as an analog of a concept lattice. The basic elements of this model are not formal concepts, but the objects of the context themselves. The correspondence between the two models is quite natural, as the edges of the graph provide a structure that allows the identification of formal concepts in the context graph. The subconcept relation on concept lattices carries over to a subgraph relation on context graphs. The model produced by this approach could be considered a map in some semantic space, where similarity of objects in the concept lattice corresponds to proximity of their nodes in the context graph.

In Sect. 5 we motivate context graphs by their potential use in information retrieval (IR). The focus of this paper is, however, a thorough theoretical underpinning of this model. To this end, the characterization of a context graph by compliant paths (Sect. 2.3) establishes a graph theoretical framework in which context graphs can be described and analyzed without explicit reference to the theory of lattices. On this basis, we can then develop an algorithm which allows the generation of (edge-minimal) context graphs without falling back on the computation of formal concepts and compare its complexity with algorithms for lattice generation.

We will start by giving a short account of the mathematical background and notations that we are going to utilize.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 178–193, 2009.

Formal Concept Analysis (FCA). The theory of Formal Concept Analysis[1] provides a mathematical framework for the analysis, structuring and/or visualization of the kind of data that is shown in Fig. 1. We have objects (here animals and plants) and binary attributes. A further example is shown in Fig. 2.

Living Beings and Water	needs water	lives in water	lives on land	needs chlorophyll	two seed leaves	one seed leaf	can move	has limbs	suckles offspring
Leech	×	×					×		
Bream	×	×					×	×	
Frog	×	×	×				×	×	
Dog	×		×				×	×	×
Spike–weed	×	×		×		×			
Reed	×	×	×	×		×			
Bean	×		×	×	×				
Maize	×		×	×		×			

Fig. 1. Formal Context: Living Beings and Water (source from [1])

Each table can be formally expressed as a *formal context*: a triple (G, M, I) consisting of a set G of objects, a set M of attributes, and a relation $I \subseteq G \times M$. The elements $(g, m) \in I$ correspond to the crosses in the table.

For an object $g \in G$, we define the set of its attributes by

$$\mathrm{att}(g) := \{m \in M \mid (g, m) \in I\} \ . \tag{1}$$

We can extend the definition to sets of objects. For a set $A \subseteq G$, the attributes shared by all $g \in A$ are collected in the set

$$\mathrm{att}(A) := \bigcap_{g \in A} \mathrm{att}(g) \ . \tag{2}$$

For a set $B \subseteq M$, those objects that have all attributes $m \in B$ are collected in the set

$$\mathrm{obj}(B) := \{g \in G \mid B \subseteq \mathrm{att}(g)\} \ . \tag{3}$$

Consider a set $B \subseteq M$. The attributes in B can be seen as a description of the subset $\mathrm{obj}(B)$ of G. The set $\mathrm{att}(\mathrm{obj}(B))$ contains all attributes in B plus those attributes that come along with the attributes in B (i.e., every object of the context that is described by B has all attributes in $\mathrm{att}(\mathrm{obj}(B))$). Therefore, $\mathrm{att}(\mathrm{obj}(B))$ is another (complete) description of $\mathrm{obj}(B)$:

$$\mathrm{obj}(\mathrm{att}(\mathrm{obj}(B))) = \mathrm{obj}(B) \ . \tag{4}$$

Tea Ladies	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Evelyn	×	×	×	×	×	×		×	×					
Laura	×	×	×		×	×	×	×						
Theresa		×	×	×	×	×	×	×	×					
Brenda	×		×	×	×	×	×	×						
Charlotte			×	×	×		×							
Frances			×		×	×		×						
Eleanor					×	×	×	×						
Pearl						×		×	×					
Ruth					×		×	×	×					
Verne							×	×	×			×		
Myra								×	×	×		×		
Katherine								×	×	×		×	×	×
Sylvia							×	×	×	×		×	×	×
Nora						×	×		×	×	×	×	×	×
Helen							×	×		×	×	×		
Dorothy								×	×					
Olivia									×		×			
Flora									×		×			

Fig. 2. Formal Context: Tea Ladies (sourced from [2], original [3]). A cross in the table indicates that a lady attended one of 14 tea parties.

A *formal concept* is a pair (A, B) with $A \subseteq G$, $B \subseteq M$, $A = \text{obj}(B)$ and $B = \text{att}(A)$. Every formal concept can be represented as a pair $(\text{obj}(B), \text{att}(\text{obj}(B)))$:

$$(A, B) = (\text{obj}(B), \text{att}(A)) = (\text{obj}(B), \text{att}(\text{obj}(B))) \ . \tag{5}$$

Conversely, it follows from (4) that every pair $(\text{obj}(B), \text{att}(\text{obj}(B)))$ is a formal concept. The sets A and B are called the *extent* and the *intent* of the formal concept, respectively.

Now consider a set $A \subseteq G$ of objects. The set $\text{obj}(\text{att}(A))$ contains all objects in A plus further objects that are "like those in A" in that they have all attributes shared by the objects of A. We can see analogously that the formal concepts are exactly the pairs $(\text{obj}(\text{att}(A)), \text{att}(A))$ with $A \subseteq G$.

The concepts can be ordered by a subconcept relation: Given two concepts (A, B) and (C, D), we call (A, B) a *subconcept* of (C, D) if $A \subseteq C$ or, equivalently, if $B \supseteq D$. The set of formal concepts, together with the subconcept relation, is called a *concept lattice* and can be depicted by a line diagram. See Fig. 3 for the concept lattice of the 'Living Beings and Water' context.

Graph Theory. In this section we provide some graph theoretical terminology. An *undirected graph* is a pair $G = (N_G, E_G)$ consisting of a set N_G of nodes and a set E_G of edges, which are two-element subsets of N_G. The *induced subgraph* on a set $S \subseteq N_G$ is the graph $G[S] = (S, E_G \cap \mathcal{P}(S))$, where $\mathcal{P}(S)$ denotes the power set of S. A graph G is called edge-minimal with respect to some property of graphs if the property is satisfied by G but not by any of the graphs (N_G, E) with $E \subset E_G$.

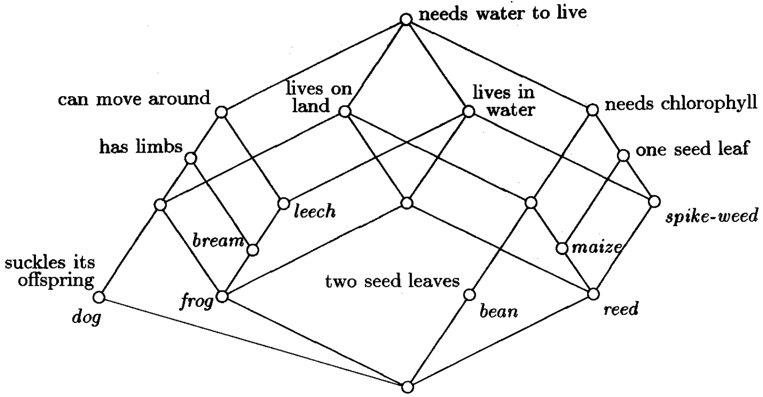


Fig. 3. Concept Lattice: Living Beings and Water (sourced from [1])

A *walk* from $x \in N_G$ to $y \in N_G$ is a sequence of nodes $(x_1, \ldots, x_n)$ with $x = x_1$, $y = x_n$ and $\{x_i, x_{i+1}\} \in E_G$ for $1 \leq i < n$. The walk is called a *path* from x to y if no two nodes are identical, except possibly the first and the last one. A path where the first and the last nodes are identical is called a *circle*. A graph is *connected* if there is a path from x to y for all $x, y \in N_G$. A connected induced subgraph $G[S]$ is called a *component* of G if there is no connected induced subgraph $G[T]$ with $S \subset T$.

2 Context Graphs

2.1 Definition

Definition 1 (Context Graph). *A context graph of a formal context* (G, M, I) *is a triple* (G, E, f) *such that*

(CG1) (G, E) *is an undirected graph,*

(CG2) $f : G \to \mathcal{P}(M)$ *is a labeling function with* $f(g) = \operatorname{att}(g)$ *for all nodes* $g \in G$,

(CG3) *For all* $B \subseteq M$, *the subgraph induced on* (G, E) *by* $\{g \in G \mid B \subseteq f(g)\}$ *is connected.*

A few remarks on the definition:

1. First of all, we will not formally distinguish between a context graph and the undirected graph in (CG1) where the subject of consideration does not require this. In particular, we will use all terminology from Sect. 1 for context graphs alike.
2. The sets $\{g \in G \mid B \subseteq f(g)\}$ are precisely the concept extents of (G, M, I) (cf. (3) and (5)). We can replace (CG3) by the following equivalent condition (using $K := (G, E, f)$ to denote the context graph):

   (CG3′) The subgraph $K[A]$ is connected for every concept extent A of (G, M, I).

The conditions (CG2) and (CG3′) can be used to check if a context graph is correctly implemented (i.e., that a search on the graph returns the correct result set for every query B), provided that the concept extents are available.

3. The context graph of a given context is not unique. Adding any number of edges to a given context graph will result in another context graph of the same context, as (CG3) will still hold. The fully connected graph on a set G is a trivial context graph of any context (G, M, I), provided that the labeling function f is defined according to (CG2).

Figure 4 shows an edge-minimal context graph for the 'Living Beings and Water' context (cf. Figs. 1 and 3).

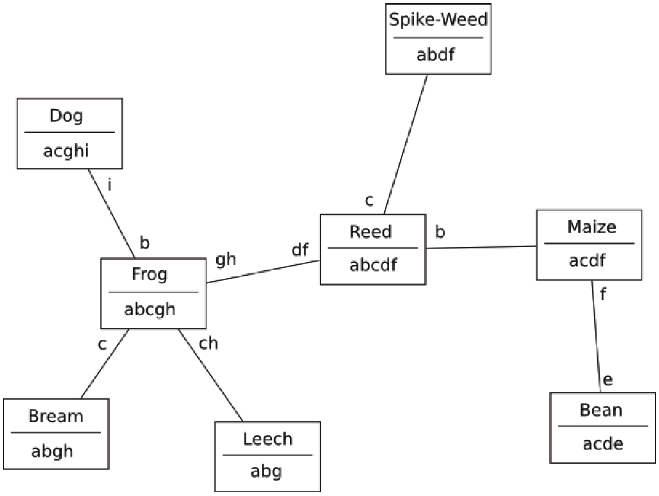


Fig. 4. Context Graph: Living Beings and Water

2.2 A Theorem on Minimality

Theorem 2. *Let $\mathbb{K} := (G, M, I)$ be a formal context. For every formal concept (A, B) there is a number $t_{\mathbb{K}}(A)$ such that every minimal context graph of $\mathbb{K}$ has exactly $t_{\mathbb{K}}(A)$ edges $\{x, y\}$ with $f(x) \cap f(y) = B$.*

Proof. 1. Let K be a context graph of (G, M, I). For a given concept (A, B), consider the graph

$$C(A) := (A, \bigcup_{(X,Y)<(A,B)} E_{K[X]}) \ . \tag{6}$$

If $C(A)$ has $c \geq 1$ components, then there must be at least $t_{\mathbb{K}}(A) := c - 1$ further edges in $E_{K[A]} \setminus E_{C(A)}$ that run between these components, since $K[A]$ is connected.

2. Let $x, y \in A$. There is a path from x to y in $C(A)$ iff there are extents $X_1, \ldots, X_n \subset A$ with $x \in X_1$, $y \in X_n$ and $X_i \cap X_{i+1} \neq \emptyset$ for $i = 1, \ldots, n-1$. Thus the components of $C(A)$ depend on the concept extents, but not on the edges in K. In particular, $t_{\mathbb{K}}(A)$ is the same for all context graphs of $\mathbb{K}$.
3. We show that $E_{K[A]} \setminus E_{C(A)} = \{\{x, y\} \in E_K \mid f(x) \cap f(y) = B\}$. If $\{x, y\} \in E_{K[A]} \setminus E_{C(A)}$, then $\{x, y\} \notin E_{K[X]}$ for all concepts $(X, Y) < (A, B)$. This means (A, B) is the smallest concept that contains $\{x, y\}$, and therefore $f(x) \cap f(y) = \mathrm{att}(\{x, y\}) = B$. The converse implication follows by the same argument.
4. Now suppose the c components are connected to each other by c or more edges. Then the components are connected to each other as a tree by $c-1$ of these edges already. That is, if we remove one of the other edges of $E_{K[A]} \setminus E_{C(A)}$, then $K[A]$ will still be connected. Suppose there is an extent D such that $K[D]$ is not connected after the edge is removed. We can choose D to be minimal with respect to this property, i.e. $K[X]$ is connected for all extents $X \subset D$. Then the edge was removed from $E_{K[D]} \setminus E_{C(D)}$ and we obtain from 3. that $D = A$, contradiction! We conclude that (CG3′) still holds after the edge is removed, so K was not minimal. □

2.3 Paths in a Context Graph

A path $(x_1, \ldots, x_n)$ in a context graph is called

- *compliant*, if $f(x_1) \cap f(x_n) \subseteq f(x_i)$ for $i = 1, \ldots, n$,
- *nonincreasing*, if $f(x_i) \cap f(x_n) \subseteq f(x_{i+1}) \cap f(x_n)$ for $i = 1, \ldots, n-1$.

Every nonincreasing path is also compliant.

Lemma 3. *Let K be a triple (G, E, f) such that (CG1) and (CG2) hold with respect to a given context (G, M, I). The following statements are equivalent:*

1. *K is a context graph.*
2. *$K[A]$ is connected for every concept extent A of (G, M, I).*
3. *For all $x, y \in G$ there is a compliant path from x to y.*
4. *For all $x, y \in G$ there is a nonincreasing path from x to y.*

Proof. We have already seen that "1.⇔2." and "3.⇐4." hold.

1.⇔3.: We observe that a path from x to y is compliant iff it lies in $\mathrm{obj}(f(x) \cap f(y))$. Such a path would connect x and y in every subgraph $K[\mathrm{obj}(B)]$ that contains both x and y, since $B \subseteq f(x) \cap f(y) \subseteq f(z)$ holds for every z on the path. Hence we have that "3.⇒1.". Conversely, if (CG3) holds then $K[\mathrm{obj}(f(x) \cap f(y))]$ is connected and we have a compliant path from x to y.

3.⇒4.: Let $x, y \in G$. By assumption there exists a compliant path $p_1 := (x_1, \ldots, x_n)$ with $x_1 = x$ and $x_n = y$. Let $z_1 := x_1$. Then $f(z_1) \cap f(y) \subseteq f(x_i) \cap f(y)$ holds for all $i = 1, \ldots, n$. Either all sets $f(x_i) \cap f(y)$ are equal and p_1 is nonincreasing, or there is a smallest index i such that $f(z_1) \cap f(y) \subset f(x_i) \cap f(y)$ and we set $x_i =: z_2$.

By hypothesis there exists a compliant path $p_2 := (z_2, \ldots, y)$. As with p_1, we either have that p_2 is nonincreasing or we obtain $p_3 := (z_3, \ldots, y)$, and so on. Because the sequence $f(z_1) \cap f(y) \subset f(z_2) \cap f(y) \subset \ldots$ is bounded by $f(y)$, which is finite, there must eventually be a $k \in \mathbb{N}$ such that p_k is nonincreasing. We obtain a nonincreasing path from z_1 to y by concatenating the initial segments $(z_i, \ldots, z_{i+1})$ of the paths p_i, where $i \leq k$ and $z_{k+1} := y$. □

3 Incremental Context Changes

In this section we will examine how a new object can be integrated into an existing minimal context graph. The question is therefore, given a minimal context graph K of a context (G, M, I) and another context (G^+, M, I^+) with

$$G^+ = G \cup \{g\} \quad \text{for some } g \notin G \ , \tag{7}$$

$$I = I^+ \cap (G \times M) \ , \tag{8}$$

how can a minimal context graph of (G^+, M, I^+) be obtained from K?

Let us first introduce an equivalence relation on the set G of objects:

$$x \sim_g y :\Leftrightarrow f(x) \cap f(g) = f(y) \cap f(g) \ . \tag{9}$$

The class of an object $x \in G$ will be denoted by $[x]_g$, and the following defines a partial order on the set of classes:

$$[x]_g \leq [y]_g :\Leftrightarrow f(x) \cap f(g) \supseteq f(y) \cap f(g) \ . \tag{10}$$

3.1 Connecting a New Object

Theorem 4. *Let $[x_1]_g, \ldots, [x_n]_g$ be the minimal classes w.r.t. the order in* (10)*.*

1. *We obtain a context graph of (G^+, M, I^+) if we add the edges $\{g, x_1\}, \ldots, \{g, x_n\}$ to the edge set of K.*
2. *Let K^+ be a context graph of (G^+, M, I^+). For all $i = 1, \ldots, n$, there is an edge $\{g, y_i\}$ of K^+ with $y_i \in [x_i]_g$.*

Proof. 1. We will show that there is a compliant path between all $x, y \in G^+$. For $x, y \in G$ there is a compliant path in K already. Otherwise, without loss of generality (WLOG) we have $x = g$ and $y \in G$. There exist an edge $\{g, z\}$ such that $[z]_g \leq [y]_g$ and a compliant path $(z, \ldots, y)$. From $[z]_g \leq [y]_g$ we obtain that $f(g) \cap f(y) \subseteq f(z)$, and we can further conclude that $f(g) \cap f(y) \subseteq f(z) \cap f(y) \subseteq f(w)$ for all w on the path $(z, \ldots, y)$. That means $(g, z, \ldots, y)$ is a compliant path from g to y.

2. Let $i \in \{1, \ldots, n\}$. Since K^+ is a context graph, there must be a nonincreasing path $(x_i, \ldots, y_i, g)$ from x_i to g. Then $\{g, y_i\}$ is an edge with $[y_i]_g = [x_i]_g$. □

3.2 Restoring Minimality

Lemma 5. *Let (G, E) be a context graph and $k := \{x, y\} \in E$. If there is a compliant path $(x, \ldots, y) \neq (x, y)$, then $(G, E \setminus \{k\})$ is also a context graph.*

We call a compliant path $(x_1, \ldots, x_n)$ a *g-step* if $[x_1]_g > [x_n]_g$ and $[x_1]_g = [x_i]_g$ for all $i = 1, \ldots, n-1$. Lemma 6 phrases a connection between g-steps and redundant edges in accordance with what we need in Sect. 4.3.

Lemma 6. *Let $\{x, y\}$ be an edge of a context graph (G^+, E^+) of (G^+, M, I^+). If we have $[x]_g, [y]_g > [g]_g$, and if g lies on every compliant path $(x, \ldots, y) \neq (x, y)$, then the following statements are equivalent:*

1. *$(G^+, E^+ \setminus \{\{x, y\}\})$ is a context graph.*
2. *We have $f(x) \cap f(y) \subseteq f(g)$, and there exist two g-steps $(x, x_2, \ldots, x_k)$ and $(y, y_2, \ldots, y_\ell)$ with $x_2 \neq y$ and $y_2 \neq x$.*

Proof. 1.⇒2.: If $(G^+, E^+ \setminus \{\{x, y\}\})$ is a context graph, there must be a compliant path $(x, \ldots, y)$. By the assumption, g lies on the path and therefore $f(x) \cap f(y) \subseteq f(g)$. Moreover, for all z on the path we have $f(x) \cap f(g) = f(x) \cap f(y) \cap f(g) \subseteq f(z) \cap f(g)$, which can be written as $[x]_g \geq [z]_g$. Because of $[x]_g > [g]_g$, the path has an initial segment $(x, x_2, \ldots, x_k)$ which is a g-step with $x_2 \neq y$. By analogy, we see that there is also a g-step $(y, y_2, \ldots, y_\ell)$ with $y_2 \neq x$.

2.⇒1.: The g-steps can be continued into nonincreasing paths $(x_1, \ldots, x_{m-1}, g)$ and $(y_1, \ldots, y_{n-1}, g)$, where $x = x_1$, $y = y_1$, $m \geq k$ and $n \geq \ell$. The two paths can be combined into a walk $p = (x_1, \ldots, x_{m-1}, g, y_{n-1}, \ldots, y_1)$. We can see that if p is a path, then p is also compliant: $f(x) \cap f(y) \subseteq f(x) \cap f(g) \subseteq f(x_i)$ holds for all $i = 1, \ldots, m$, and similarly $f(x) \cap f(y) \subseteq f(y_j)$ for all $j = 1, \ldots, n$.

Now if p was not a path, there would be a smallest index i such that $x_i = y_j$ holds for some j. Then $\hat{p} := (x_1, \ldots, x_{i-1}, y_j, \ldots, y_1)$ would be a compliant path without g. Moreover $\hat{p} \neq (x, y)$, which follows from $x_2 \neq y$ (if $j = 1$) or $y_2 \neq x$ (if $j \neq 1$). This would contradict our precondition. So p is a path and by Lemma 5, $(G^+, E^+ \setminus \{\{x, y\}\})$ is a context graph. □

4 Algorithm

In this section we present an incremental construction algorithm for context graphs that is based on Theorem 4.1 and Lemma 6. Every time a new object g is to be added, the existing context graph is traversed three times. During the first traversal, the $\sim_g$-classes $[x]_g$ and $[y]_g$ are compared for each edge $\{x, y\}$ of the graph, and the result of that comparison (greater, less, equal, incomparable) is stored as a label of that edge (Sect. 4.1). In the second and third traversal, the edge labels are used to connect g to the graph and to remove redundant edges, respectively (Sects. 4.2 and 4.3). In Sect. 4.4, we will address the complexity of the overall algorithm.

4.1 Step 1: Computing Edge Labels

The computation of the edge labels is triggered by calling the function *procNode*() in Fig. 5 on an arbitrary node x of the graph (passing g as a second parameter). Starting in x, the graph will be traversed by recursively calling *procNode*(y, g) for all neighbors y of the current node x (lines 2+5). The variable *state*(x) initially has the value 'new', and it is set to 'entered' when x is entered for the first time (line 1). The value is set to 'finished' when the node is completely processed (line 7). The variable *state*(y) in line 3 holds the value 'entered' iff, during the traversal of the graph, y has been entered before x. This way, the label of each edge is computed exactly once (lines 3+4).

```
procNode(x,g)

1    state(x):='entered';
2    for all y ∈ neighbors(y)
3        if( state(y)='entered' )
4            computeLabel();
5        else if( state(x)='new' )
6            procNode(y,g);
7    state(x):='finished';
```

Fig. 5. Step 1: Computing Edge Labels

The function *computeLabel*() is not given explicitly. The computation could be achieved by testing for $f(x) \cap f(g) \subseteq f(y) \cap f(g)$ and $f(y) \cap f(g) \subseteq f(x) \cap f(g)$, assigning an edge label based on the four possible outcomes. We will assume that the edge labels can then be accessed by a function *label*(x, y), that returns a symbolical representation as follows:

$$label(x, y) := \begin{cases} \downarrow & \text{if } [x]_g > [y]_g \ , \\ = & \text{if } [x]_g = [y]_g \ , \\ \uparrow & \text{if } [x]_g < [y]_g \ , \\ ⪤ & \text{otherwise} \ . \end{cases} \tag{11}$$

Note that the edge labels are only used for the purpose of adapting the context graph to a new object g and are not required afterwards.

We will use the "Tea Ladies" context from Fig. 2 to illustrate the steps of our algorithm. Figure 6 shows a minimal context graph of the formal context that is obtained from the one in Fig. 2 by removing the row for "Theresa". That is, if we add the object "Theresa" following the three steps of our algorithm, we will obtain a minimal context graph for the "Tea Ladies" example. As the result of the first step, we obtain edge labels as shown in Fig. 6.

4.2 Step 2: Connecting the Object

By Theorem 4.1, we have to connect g to exactly one object taken from each minimal $\sim_g$-class. Let x be an object in the graph. If there exists a g-step starting

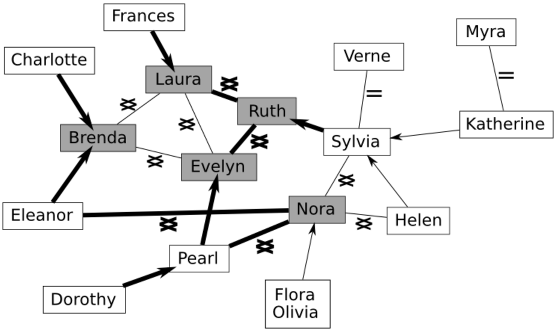


Fig. 6. Context graph for the 'Tea Ladies' context before adding 'Theresa' as the last object. Objects of minimal classes $[x]_g$, where $g =$'Theresa', have been shaded. An edge is directed from x to y if $label(x, y) =$'$\downarrow$'. Edges with property (12) have been highlighted.

in x, then obviously $[x]_g$ is not minimal. To see that the converse is also true, consider that by Lemma 3.4 there will be a nonincreasing path $(x, \ldots, y, g)$ in the graph that we are going to construct (and we know that this graph exists). Furthermore, $[y]_g$ is minimal and $[x]_g$ is assumed to be not, so we have $[x]_g > [y]_g$ and the path starts with some g-step that is also contained in the current graph. To sum up, $[x]_g$ is not minimal iff there is some g-step starting in x.

A g-step can be identified using the *label*() function defined in Sect. 4.1: A g-step is a path $(x_1, \ldots, x_n)$ with $label(x_{n-1}, x_n) =$'$\downarrow$', and $label(x_i, x_{i+1}) =$'$=$' for $i < n - 1$. An algorithm for connecting the new object might work like this: We start in an arbitrary node of the graph and recursively follow all edges labeled by '='. In every node, we check the labels of all outgoing edges and remember if any is labeled by '↓', but do nothing else. If we are back in the starting node and there was no ↓-edge, the starting node is connected with g. We repeat this with a new starting node as long as the graph contains unvisited nodes.

Executing Step 2 on our context graph from Fig. 6 gives us the five minimal elements "Laura", "Brenda", "Ruth", "Evelyn" and "Nora", which are all in their own equivalence class. At this point, we create a new node for the object Theresa" and connect it to these nodes.

4.3 Step 3: Removing Edges

Assumptions. An edge $\{x, y\}$ in a context graph can be removed iff x and y are connected by a compliant path other than (x, y) (Lemma 5). As a result of the previous step, the end nodes of an edge $\{x, y\}$ may have become connected by a new compliant path through g. Assume that the context graph was minimal before g was added. Then there is no compliant path from x to y that does *not* contain g. This amounts to one of the prerequisites in Lemma 6.

The other prerequisite, namely $[x]_g, [y]_g > [g]_g$, does not generally hold. However, if it does not hold then WLOG we have $[x]_g = [g]_g$, which means that $[x]_g$

is the unique minimal class with respect to the order in (10). Then g has only one neighbor and can not lie on a path between $x, y \neq g$. So in this case, Step 3 can be skipped because the context graph is already minimal.

Conditions for Removal. To sum up, we can remove an edge $\{x, y\}$ of the graph iff there are g-steps $(x, x_2, \dots, x_k)$, $x_2 \neq y$, and $(y, y_2, \dots, y_\ell)$, $y_2 \neq x$, and

$$f(x) \cap f(y) \subseteq f(g) \tag{12}$$

also holds. By Lemma 3.4, g-steps starting in x and y do always exist (provided that $[x]_g, [y]_g > [g]_g$). The crucial point is that $x_2 \neq y$ and $y_2 \neq x$ hold, because this means that the g-steps can be continued into *different* paths to g. This is necessarily the case whenever $label(x, y)$ =‘≷’. Similarly, if $label(x, y)$ =‘↓’, we don’t need to check that $y_2 \neq x$. Only for an edge with $label(x, y)$ =‘=’, the test can not be further simplified. The three cases are illustrated in Fig. 7.

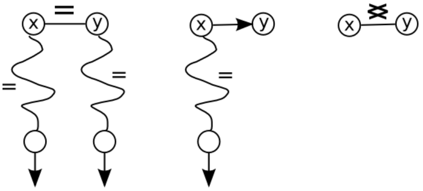


Fig. 7. Patterns indicating that $\{x, y\}$ is a candidate for removal

We could now use the algorithm described in the following section to remove redundant edges from the graph in Fig. 6. However, this case is simple if we have a look at the ten highlighted edges in Fig. 6: The right pattern in Fig. 7 obviously matches the four edges labeled with ’≷’, so these edges can be removed. The other patterns do not match. To sum up, five edges have been added, four removed and we obtain the context graph for the “Tea Ladies” context.

Implementation. The function *minimize*() in Fig. 8 shows how the idea can be implemented. It is instructive to first consider the hypothetical case that no edges with label ‘=’ occur in the graph. As you can guess from Fig. 7, the algorithm could be simplified considerably. Those lines in Fig. 8 that are marked with an asterisk can be left out in this case, and we assume that *minimize*() is a nonrecursive function that is called separately for every node x.

For every node x, we check the labels of all outgoing edges $\{x, y\}$ (line 3). If $label(x, y)$ =‘≷’ and the edge satisfies (12), then the edge can be removed immediately (lines 7-9). We only need to check for (12) if the other node has not been processed already (line 8). The case $label(x, y)$ =‘↓’ requires a bit more effort. We can remove all edges with $label(x, y)$ =‘↓’ and (12) as long as one edge with $label(x, y)$ =‘↓’ remains (cf. Fig. 7 middle).

We use the local variable `exit` to keep track of the edges with $label(x, y)$ =‘↓’. The variable is initialized with 0, which signifies that no ↓-edges have been found

```
minimize(x,g)

 1     state(x) := ‘entered’;
 2     exit := 0;
 3     forall y ∈ neighbors(x)
 4*        if(label(x,y) = ‘↑’)
 5*            if(¬ state(y)=‘entered’)
 6*                minimize(y,g);
 7         else if(label(x,y) = ‘⪤’)
 8             if(¬ state(y)=‘entered’ ∧ f(x) ∩ f(y) ⊆ f(g))
 9                     remove {x,y};
10*        else if(label(x,y) = ‘=’ ∧ f(x) ∩ f(y) ⊆ f(g))
11*            if(minimize(y,g) ≠ 0)
12*                remove {x,y};
13         else
14             if(label(x,y) = ‘↓’)
15                 if(f(x) ∩ f(y) ⊆ f(g))
16                     outedge = {x,y};
17                 else
18                     outedge = □;
19*            else
20*                if(¬ state(y)=‘entered’)
21*                    outedge = minimize(y,g);
22             exit=evaluate(exit,outedge);
23*    return exit;
```

Fig. 8. Step 3: Removing Edges

exit	**outedge**	**exit**	**remove**
0	0	0	-
0	k	k	-
0	□	□	-
k	0	k	-
k	k_2	k	k_2
k	□	□	k
□	0	□	-
□	k	□	k
□	□	□	-

Fig. 9. Specification of the evaluate() function

so far. The variable will be assigned the value □ as soon as we find a ↓-edge which doesn't satisfy (12); this will signify that all subsequent ↓-edges that do satisfy (12) can be removed. If we find a ↓-edge with (12) and we have `exit`≠ □, then the edge itself will be assigned to `exit` so we can remove it later. Figure 9 shows how the value of `exit` is updated when a new ↓-edge is found, where the value of `outedge` is assigned as in lines 14-18. It also shows which edges can be removed in the same process. The function *evaluate*() implements the table in Fig. 9, including the removal of edges (line 22). Note that in the case $label(x,y)$ =‘↑’ nothing needs to be done, because the edge is dealt with when y is processed.

Now let us include lines 19-21 and 23. Line 19 is executed whenever there is an =-edge for which (12) does not hold. Assume that the case in line 10 does not occur. What has changed is that we now recursively follow along paths of =-edges and return any ↓-edges that we find to the calling instance (line 23). If a ↓-edge can be reached from x via a path of =-edges, we will treat it as if it were directly connected to x (line 21). It is now possible that `outedge=0`; Fig. 9 shows that the function *evaluate*() does nothing in this case, as we would expect.

We can also include lines 4-6. This does not interfere with the current computation because subsequently, only ↑-edges and =-edges are followed (and the return value of $minimize(y,g)$ will also be ignored). We will now assume that we call $minimize(x,g)$ only once for $x = g$, assuming that we have correctly set $label(g,z)$ =‘↑’ for all neighbors z of g. From every node of the graph there is a nonincreasing path to g, so conversely this means that every node of the graph is reached by $minimize(g,g)$.

Finally, we include lines 10-12. If a =-edge $\{x,y\}$ satisfies (12), every path between x and y consisting of =-edges is a compliant path. So by our assumptions there is no path of =-edges from x to y other than (x,y). This means that we can treat the nodes on both sides of the edge separately and remove $\{x,y\}$ iff the situation depicted in Fig. 7 (left) occurs. This can be decided after $minimize(y,g)$ returns in line 11: There is at least one ↓-edge on the side of x because we have come from g, and whether or not there is a ↓-edge on the side of y will be returned by the `exit` parameter.

4.4 Complexity

We can obtain the time complexity of our construction algorithm from the pseudocode in Figs. 5 and 8. G, M, E and L denote the sets of objects, attributes, edges and concepts, respectively. We analyze each step separately.

Step 1: The code inside the for-loop is executed twice for every edge $\{x,y\}$ of the graph, once each for the processing of x and y. The function *computeLabel*() computes $f(x) \cap f(g) \subseteq f(y) \cap f(g)$ and $f(y) \cap f(g) \subseteq f(x) \cap f(g)$, which can be done in $\mathcal{O}(|M|)$. So we have a total complexity of $\mathcal{O}(|E| \cdot |M|)$ for Step 1.

Step 2: This step involves only a traversal of the graph. All operations are constant time, resulting in $\mathcal{O}(|E|)$ for this step.

Step 3: As in Step 1, the for-loop is executed $\mathcal{O}(|E|)$ times. All operations are constant time, except $f(x) \cap f(y) \subseteq f(g)$, which takes $\mathcal{O}(|M|)$ time.

In order to construct a context graph from a formal context, the three steps have to be executed once for each object in G. The time complexity of the overall algorithm is therefore $\mathcal{O}(|G| \cdot |E| \cdot |M|)$. An overview of lattice construction algorithms is given in [4]. The paper gives $\mathcal{O}(|G| \cdot |L| \cdot (|G| + |M|))$ as the best known worst case complexity of a lattice construction algorithm. It seems reasonable to assume that $|E| = \mathcal{O}(|L|)$ as we had $|E| < |L|$ for the contexts we examined so far, but we have no proof of this.

5 Applications

FCA has been applied to a wide variety of problem domains, including information retrieval [5,6], ontology construction from RDF descriptions [7], behavioural genetics [8], and software reuse [9]. Here we focus on IR, where the objects G are typically text documents, and the attributes M are the terms that occur in them. The number of terms used is usually restricted via some selection criterion, such as a threshold based on the frequency-based weights used in traditional IR [10]. The query process corresponds to finding the concept (and its neighbors) that contains the query, where the query is treated as a new object.

It has long been known that it is difficult for a user to formulate an effective query for a large corpus of objects when he/she is unfamiliar with its contents [11]. Fixed, pre-computed indices are also often unsatisfactory, providing very limited ways of exploring the corpus [12]. Graph or lattice-based corpus models provide a possible solution to this query formulation problem: they facilitate *navigational search*, which may be thought of as a combination of querying and browsing. Users can navigate the model by following edges. This is a form of browsing, but it is directed (hence *navigational*), as edge labels provide information about attributes that are added or removed from the (implicit) query when the edge is followed. The very large number of possible paths through the structure enables much more flexible browsing than is possible via linear indices.

The context graph provides a structure that can support navigational search. For small contexts, such as that shown in Fig. 4, the context graph may be used directly to visualize all or part of the context. Edge labels aid navigation. For example, a user viewing the node 'Frog' can see that traversing the edge towards node 'Reed' will add attributes d ('needs chlorophyll') and f ('one seed leaf'), and remove attributes g ('can move') and h ('has limbs'). For large contexts, where direct visualization of all the objects and attributes is impractical, the context graph can provide an internal representation that supports a more sophisticated user interface for navigational search, such as that in [12].

6 Related Work

In this article, we have motivated context graphs as potential models for IR. The examples we have given, and other examples we have computed so far, indicate that context graphs use relatively few edges to represent a hierarchy of formal concepts. We believe that this means that navigation in a concept lattice can be efficiently simulated using a context graph as an underlying structure, instead of the lattice itself. Obviously, the size of the graph is at most quadratic in the number of objects, whereas the lattice can grow exponentially. Computing the context graph would thus be a clear advantage compared to computing the concept lattice if the context has a large number of objects. Another option in this case is to compute the lattice as it is traversed. In the context of IR we refer to a paper of Ferré who combines this approach with dynamic taxonomies, which are presented to a user as a local view reflecting their current position in the lattice [12]. In [13], a simple algorithm is presented which can efficiently compute all upper (or lower) neighbors of a concept.

In [2], Wille uses the "Tea Ladies" example to compare FCA with methods of social network analysis. In particular, a graph is shown which results from a method presented in [14]. The objects of the graph are the tea ladies of the formal context, and the basic idea of the graph is that an edge exists between two ladies iff they have been together at one of the meetings (i.e., if they share an attribute). To reduce the number of edges, not all meetings have been taken into account. In the context graph, two ladies who have attended the same meeting are connected by a compliant path, but not necessarily by an edge, which reduces the number of edges significantly.

Finally, we contrast context graphs with Conceptual Graphs [15]. Conceptual Graphs are used to symbolically express meaning. The value of context graphs, at least in the scope of this paper, lies in the organization of data objects which is realized by their structure. The structuring principle is similarity, and this means that some kind of meaning could be reflected in the structure, but context graphs are not devised and probably not suited for *expressing* meaning.

7 Conclusion

The contribution of this paper is the introduction and thorough study of context graphs, a relatively compact representation of formal contexts in which objects are vertices and formal concepts occur as certain subgraphs. We characterized context graphs and their relationship to FCA by several theorems. These led to an algorithm for the incremental construction of context graphs. We analyzed the complexity of this algorithm and compared it with the best known worst-case complexity for the incremental construction of a formal concept lattice. We reasoned that context graphs are a promising basis for navigational search in domains with massive and frequently changing numbers of objects. Such domains are ever more abundant, e.g. social networking sites such as Facebook, the semantic web, large-scale enterprise architecture, and data warehousing.

Formal contexts hold the promise of enabling and facilitating such exploration and navigational searches without the need for a complex lattice-theoretic or algebraic query language other than in advanced searches. A full evaluation of this claim is out of the scope of this paper, which instead uses small and well-known examples to illustrate the formal theory of context graphs and their relationship to FCA. Some of these examples have been computed with an initial implementation of context graphs using Java servlets and a facility to upload and analyze moderately sized formal contexts.

References

1. Ganter, B., Wille, R.: Formal concept analysis: mathematical foundations. Springer, Berlin (1999)
2. Wille, R.: Concept lattices and conceptual knowledge systems. Computers and Mathematics with Applications 23(6–9), 493–515 (1992)
3. Davis, A., Gardner, B.B., Gardner, M.R.: Deep South. Univ. Chicago Press, Chicago (1941)
4. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental and Theoretical Artificial Intelligence 14, 189–216 (2002)
5. Priss, U.: Lattice-based information retrieval. Knowledge Organization 27(3), 132–142 (2000)
6. Carpineto, C.: Conceptual structures in modern information retrieval. In: [16], p. 1
7. Delteil, A., Faron, C., Dieng, R.: Building concept lattices by learning concepts from RDF graphs annotating web documents. In: [16], pp. 191–204
8. Duquenne, V., Chabert, C., Cherfouh, A., Delabar, J.M., Doyen, A.L., Pickering, D.: Structuration of phenotypes/genotypes through Galois lattices and implications. In: The 2001 International Workshop on Concept Lattice-based theory, methods and tools for Knowledge Discovery in Databases, Stanford University, CA, USA, July 30 (2001)
9. Godin, R., Mili, H.: Building and maintaining analysis-level class hierarchies using Galois lattices. In: Paepcke, A. (ed.) The Conference on Object-oriented Programming Systems, Languages and Applications, Washington, DC, pp. 394–410 (1993)
10. Carpineto, C., Romano, G.: Information retrieval through hybrid navigation of lattice representations. International Journal of Human-Computer Studies 45(5), 553–578 (1996)
11. ter Hofstede, A.H.M., Proper, H.A., van der Weide, T.P.: Query formulation as an information retrieval problem. The Computer Journal 39(4), 255–274 (1996)
12. Ferré, S.: CAMELIS: Organizing and browsing a personal photo collection with a logical information system. In: Diatta, J., Eklund, P., Liquière, M. (eds.) The International Conference on Concept Lattices and Their Applications. CEUR Workshop Proceedings, vol. 331, pp. 112–123 (2007)
13. Lindig, C.: Fast concept analysis. In: Working with Conceptual Structures - Contributions to ICCS 2000, pp. 152–161. Shaker Verlag (2000)
14. Breiger, R.L.: The duality of persons and groups. In: Wellman, B., Berkowitz, S.D. (eds.) Social Structures: A Network Approach, pp. 83–98. Cambridge University Press, Cambridge (1988)
15. Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, Reading (1984)
16. Priss, U., Corbett, D.R., Angelova, G. (eds.): ICCS 2002. LNCS, vol. 2393. Springer, Heidelberg (2002)

Handling Large Formal Context Using BDD – Perspectives and Limitations

Andrei Rimsa, Luis E. Zárate, and Mark A. J. Song

Department of Computer Science, Applied Computational Intelligence Laboratory
Pontifical Catholic University of Minas Gerais - Brazil
rimsa@live.com, {zarate,song}@pucminas.br

Abstract. This paper presents Binary Decision Diagrams (BDDs) applied to Formal Concept Analysis (FCA). The aim is to increase the FCA capability to handle large formal contexts. The main idea is to represent formal context using BDDs for later extraction of the set of all formal concepts from this implicit representation. BDDs have been evaluated based on several types of randomly generated synthetic contexts and on density variations in two distinct occasions: (1) computational resources required to build the formal contexts in BDD; and (2) to extract all concepts from it. Although BDDs have had fewer advantages in terms of memory consumption for representing formal contexts, it has true potential when all concepts are extracted. In this work, it is shown that BDDs could be used to deal with large formal contexts especially when those have few attributes and many objects. To overcome the limitations of having contexts with fewer attributes, one could consider vertical partitions of the context to be used with distributed FCA algorithms based on BDDs.

Keywords: Formal Concept Analysis, Formal Context, Formal Concept, Binary Decision Diagrams.

1 Introduction

At the International Conference on Formal Concept Analysis in Dresden (ICFCA 2006) an open problem of "Handling large contexts" was pointed out and as an example was cited the challenge of "how to calculate/generate all concepts of a large context" (e.g. 120,000 x 70,000 objects attributes). In these cases, traditional FCA algorithms have high computational cost and demand high execution times, making the extraction of all concepts infeasible for larger contexts.

One possible solution to deal with the problem of handling large formal contexts is to apply a distributed solution for the processing of contexts. Partial concepts are obtained for later merging through a specific operator to find the final set of concepts. Several authors have presented formal proposals and mathematical formalisms for distributed application of FCA, as can be seen in [1-3]. However, these contributions do not analyze the performance aspects of the distributed version concerning the density impact on the context.

It is clear the potential of FCA to represent and extract knowledge from a set of objects and attributes and it is even more clear the problem of dealing with databases of

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 194–209, 2009.

high dimensionality. Application in real problems often suffers from this common fact. In this work, an approach to meet the challenge mentioned above consists in applying Binary Decision Diagrams [4] to obtain a symbolic representation of a cross table (formal context) that allows a more efficient extraction of the set of all concepts. It will be shown that this approach is promising and that it can handle more efficiently with large contexts when compared with the conventional implementation of algorithms that handles tables. Although BDD suffers from limitations of handling contexts with many attributes, a common problem faced by FCA, it can handle huge amount of objects, making it thus reliable for some set of problems. Also, in these conditions, the BDD representation presents computational improvements. In some cases it can even save days of processing, as it will be later addressed.

BDD has already been used earlier in FCA. In [5], using previously obtained concepts, a concept lattice is built based on ZBDDs (Zero-Suppressed BDDs) [6], a type of BDD. In this paper, BDD have been applied with a different aim, to represent formal contexts in order to improve concepts computation. This article presents an analysis of this new representation, both in its impact in memory consumption as the computational time required to execute an algorithm to extract all concepts.

This article is organized in five sections. In the second section, the main concepts of the FCA and BDD are reviewed. In the third section, examining the representation of formal contexts through BDD is discussed. In the fourth section, the principles and algorithm for extraction of formal concepts from BDD are presented. In the last section, the conclusions and future works are pointed out.

2 Formal Context

2.1 Formal Concept Analysis

Formal Context. Formal contexts have the notation $K:=(G, M, I)$, where G is a set of objects (rows headers), M is a set of attributes (columns headers) and I is an incidence relation ($I \subseteq G \times M$). If an object $g \in G$ and an attribute $m \in M$ are in the relation I, it is represented by $(g, m) \in I$ or gIm and is read as "*the object g has the attribute m*".

Given a set of objects $A \subseteq G$ from a formal context $K:=(G, M, I)$, it could be asked which attributes from M are common to all those objects. Similarly, it could be asked, for a set $B \subseteq M$, which objects have the attributes from B. These questions define the derivation operators, which are formally defined as:

$$A' := \{m \in M \mid gIm \ \forall \ g \in A\} \tag{1}$$

$$B' := \{g \in G \mid gIm \ \forall \ m \in B\} \tag{2}$$

A special case of derivate sets occurs when empty sets of objects or attribute are considered to be derivate:

$$A \subseteq G = \emptyset \Rightarrow A' := M \ ; \ B \subseteq M = \emptyset \Rightarrow B' := G \tag{3}$$

Formal Concept. Formal concepts are pairs (A, B), where $A \subseteq G$ (called extent) and $B \subseteq M$ (called intent). Each element of the extent (object) has all the elements of the intent (attributes) and, consequently, each element of the intent is an attribute of all objects of the extent. The set of all formal concepts in a formal context has the notation *B(G, M, I)*. From a cross table representing a formal context, algorithms can be applied in order to determine its formal concepts and its line diagram [7].

2.2 Binary Decision Diagrams

Binary decision diagrams are a canonical representation of boolean formulas [4]. The BDD is obtained from a binary decision tree by merging identical subtrees and eliminating nodes with identical left and right siblings. The resulting structure is a graph rather than a tree in which nodes and substructures are shared.

Formally, a BDD is a directed acyclic graph with two types of vertex: non-terminal and terminal. Each non-terminal vertex *v* is labeled by *var(v)*, a distinct variable of the corresponding boolean formula. Each *v* has at least one incident arc (except the root vertex). Also, each *v* has two outgoing arcs directed toward two children: *left(v)*, corresponding to the case where *var(v)=0*, and *right(v)* to the case where *var(v)=1*.

A BDD has two terminal vertices labeled by 0 and 1, representing the truth-value of the formula false and true, respectively. For every truth assignment to the boolean variables of the formula, there is a corresponding path in the BDD from root to a terminal vertex. Figure 1 illustrates a BDD for the boolean formula $(a \wedge b) \vee (c \wedge d)$ compared to a Binary Decision Tree for this same formula.

BDDs are an efficient way to represent boolean formulas. Often, they provide a much more concise representation compared to the traditional representations, such as conjunctive and disjunctive normal forms. BDDs are also a canonical representation for boolean formulas. This means that two boolean formulas are logically equivalent if and only if its BDDs are isomorphic. This property simplifies the execution of frequent operations, like checking the equivalence of two formulas.

However, BDD has drawbacks. The most significant is related to the order in which variables appear. Given a boolean formula, the size of the corresponding BDD is highly dependent on the variable ordering. It can grow from linear to exponential according to the number of variables of the formula. In addition, the problem of choosing a variable order that minimize the BDD size is NP-complete [4]. There are heuristics to order the BDD variables; some of them are based on the knowledge over the problem. A review of some heuristics can be found in [8].

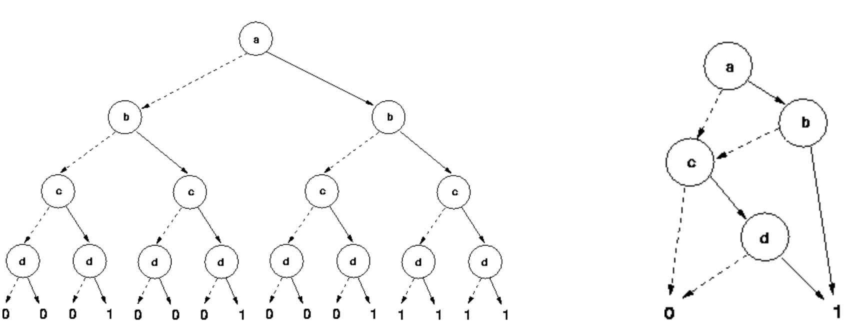


Fig. 1. Binary decision tree and a correspondent BDD for the formula $(a \wedge b) \vee (c \wedge d)$

3 Formal Contexts Represented in BDD

3.1 BDD Construction from Formal Contexts

As mentioned, BDD is able to represent logical expressions through simplified graphs. In this way, a context can be converted into an equivalent logic formula to be used in the creation of the BDD representation. Table 1 shows an example of a formal context and its possible representation through a logic function.

Table 1. Formal Context Example

	a_1	a_2	a_3
o_1	**X**		**X**
o_2	**X**	**X**	
o_3		**X**	

$$f(a_1, a_2, a_3) = a_1\overline{a_2}a_3 + a_1a_2\overline{a_3} + \overline{a_1}a_2\overline{a_3} \quad (4)$$

Note that each object is represented by a logic equation, according to the presence or not of its attributes. The function *f(a_1, a_2, a_3)* results in a positive state (1) when an object is present on the context. This function returns the negative state (0) for objects not present in the context. Thus, any context can be represented by a logic function.

```
Algorithm 1. BDD construction based on the context.

in:  List<Object> list
out: BDD context
 1: context = bddfalse
 2: while !list.empty( ) do
 3:   obj = list.removeFirstObject( );
 4:   BDD tmp = bddtrue
 5:   for i=0; i<obj.attributes; i++ do
 6:     if obj.hasAttribute(i) then
 7:       tmp &= bdd_ithvar(i)
 8:     else
 9:       tmp &= bdd_nithvar(i)
10:     endif
11:   done
12:   context |= tmp
13: done
```

Algorithm 1 allows the construction of BDD based on the objects presented in the formal context. Note that this algorithm maintains the same attribute order of the formal context in order to build the context in BDD. The internal functions *bdd_ithvar* and *bdd_nithvar* are specific to the *BuDDy* [9] library and are used to define the presence or not of an attribute in the BDD, respectively. Once the conjunction of attributes is made, forming the objects (lines 7 and 9), then a disjunction of those objects is realized (line 12).

It is important to emphasize that the main objective of this work is to show the feasibility of BDD to represent formal contexts, and from that representation extract the formal concepts. The feasibility is shown through the manipulation of large formal contexts. In most cases the BDD representation is less memory efficient when compared to a bit table representation of the contexts, as will be seen in the next section.

But the memory consumption is not significant enough to invalidate its representation in BDD. So the BDD can be used to extract concepts more efficiently than the algorithms that work directly in the tabular representation.

3.2 BDD Representativeness Analysis

To achieve a more reliable simulation environment, all context used in this work were built by the SCGaz tool (available at http://www.inf.pucminas.br/projetos/licap) to evaluate the BDD performance. This tool can randomly generate formal contexts with user-defined densities while avoiding the existence of some type of redundant attributes and objects. The use of partially clarified contexts was considered in this work to guarantee that the final size of the BDD representation is not influenced by the existence of repeated objects and attributes, since this representation can internally simplify these redundancies. This is an important step to ensure fairness of the representation that could otherwise be used to mask the true performance of BDD. Note that the BDD representation is not restricted to clarified contexts and can be applied to any type of context, regardless of its information redundancy.

To construct and operate the BDD, the *BuDDy* library was used in which each node of its graph is represented by 20 bytes. Thus, the number of nodes in the graph was the parameter used to quantify the representation memory consumption. In order to compare the BDD representativeness, a relationship between the bit table memory consumption was stipulated. Where S_{table} and S_{bdd} correspond respectively to the table and BDD memory sizes. This metric calculates the gain (*Gain*) of a representation in relation to another. Thus, when the BDD consumes less memory than the bit table, equation (5) is then used. When the bit table is more efficient, the expression of equation (6) is therefore used. The negative sign indicates the loss of the BDD compared to the bit table memory consumption.

$$Gain = \left(\frac{S_{table}}{S_{bdd}} \right) \tag{5}$$

$$Gain = -\left(\frac{S_{bdd}}{S_{table}} \right) \tag{6}$$

Note that the number of nodes in the BDD is not related to the number of filled cells of the context. Contexts with the same density and filled cells can present BDDs with more or less nodes.

To assess the representativeness of contexts through BDD, it is considered their behavior over different types of context: $|G|=|M|$, $|G|<|M|$, $|G|>|M|$ and many-valued. For each type of context considered, a pair (attribute, object) was simulated for 10 cases of density, ranging from its minimum to its maximum value. For the types of contexts with unique densities (many-valued and contexts $|G|>|M|$ with $|G|=2^{|M|\ -\ 2}$), a single simulation was performed. The implementations were made in C++ and the simulations were realized on a Pentium IV 3.06Ghz HT with 2GB of RAM running Slackware Linux 12.0.

Contexts |G| = |M|. Table 2 and Figure 2 correspond to the representation of contexts through BDD, where the number of objects is equal to the number of attributes. Table 2 shows the minimum, maximum, and the median values for the BDD memory consumption (S_{bdd}), as well as the required time to build the BDD (T_{bdd}) and the representativeness gain compared to the bit table. Figure 2 shows the gains obtained for the cases (100, 100), (1000, 1000) and (5000, 5000) in function of the density.

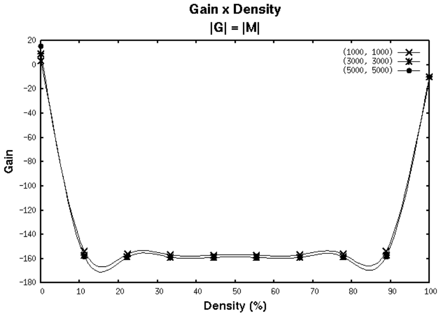


Fig. 2. BDD gain for contexts |*G*| = |*M*|

Table 2. BDD simulation for contexts |*G*| = |*M*|

|G|	|M|	S_{table} (Kb)	S_{bdd} (Kb)			T_{bdd} (s)			Gain		
			Min	Median	Max	Min	Median	Max	Min	Median	Max
50	50	0	2	36	39	0.01	0.02	0.03	-128.21	-117.18	-6.35
100	100	1	4	163	172	0.08	0.22	0.25	-140.64	-133.51	-3.18
300	300	11	12	1636	1669	3.27	9.06	9.18	-151.87	-148.94	-1.06
500	500	31	20	4660	4718	18.19	47.53	49.93	-154.60	-152.69	1.56
1000	1000	122	39	19038	19163	362.66	661.04	786.00	-156.98	-155.96	3.13
1500	1500	275	59	43157	43357	1217.48	2435.05	2585.38	-157.86	-157.13	4.69
2000	2000	488	78	77029	77310	3013.76	5959.37	6059.72	-158.33	-157.76	6.25
2500	2500	763	98	120662	121019	5728.60	11723.10	12051.70	-158.62	-158.15	7.81
3000	3000	1099	117	174050	174493	7252.94	14696.80	14829.50	-158.83	-158.42	9.38
3500	3500	1495	137	237203	237718	11450.90	23694.70	23907.90	-158.97	-158.63	10.94
4000	4000	1953	156	310110	310708	17159.50	35908.95	36225.80	-159.08	-158.78	12.50
4500	4500	2472	176	392771	393462	24577.30	52143.25	52561.80	-159.17	-158.89	14.06
5000	5000	3052	195	485201	485977	34321.40	73011.15	73679.80	-159.25	-158.99	15.63

As expected, BDD obtained a better performance in low and high densities (approximately below 10% and above 90%) when compared to intermediate densities. This occurs because with few or many incidences in the context table, there are several similarities in the BDD graph that allow simplifications of sub-expressions. The BDD is therefore represented with fewer nodes. Meanwhile, in the intermediate densities, the BDD had a constant behaviour. This happens because the BDD is using few objects compared to the total universe of objects ($2^{|M|}$), making no influence in the representation size. The BDD is unable to find enough objects similarities that could allow significant simplification. So, the intermediate density has no effect over the BDD representative size.

It is noteworthy that, although the BDD has had a lower performance than the bits table, the representation of formal contexts in BDD is computationally feasible. Through data collected by the simulations, it is pointed out that, for a context with

5,000 attributes and 5,000 objects, the BDD required expressively more memory than the bit table representation and was able to build it in a viable computational time of about 20 hours.

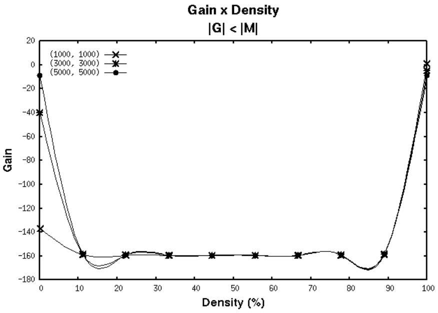


Fig. 3. BDD gain for contexts |*G*| = |*M*|

Table 3. BDD simulation for contexts |*G*| < |*M*|

|G|	|M|	S_{table} (Kb)	S_{bdd} (Kb)			T_{bdd} (s)			Gain		
			Min	Median	Max	Min	Median	Max	Min	Median	Max
100	1000	12	42	1921	1929	25.19	27.68	28.03	-158.05	-157.40	-3.43
600	1000	73	390	11444	11515	151.76	253.24	255.87	-157.22	-156.24	-5.33
900	1000	110	860	17140	17251	237.00	419.00	423.23	-157.02	-156.02	-7.83
200	2000	49	82	7739	7757	211.45	286.26	289.19	-158.87	-158.49	-1.67
1200	2000	293	1612	46263	46419	1318.07	2475.16	2498.55	-158.44	-157.91	-5.50
1800	2000	439	3664	69342	69589	2099.88	3783.11	3819.29	-158.35	-157.79	-8.34
300	3000	110	117	17457	17489	725.02	1267.96	1276.93	-159.19	-158.90	-1.06
1800	3000	659	3179	104497	104745	4678.94	8663.08	8734.99	-158.90	-158.53	-4.82
2700	3000	989	8648	156663	157054	9454.73	13168.50	13297.90	-158.84	-158.44	-8.75
400	4000	195	213	31079	31124	1725.56	3227.89	3258.88	-159.36	-159.12	-1.09
2400	4000	1172	4969	186152	186497	11886.00	20967.85	21207.00	-159.15	-158.85	-4.24
3600	4000	1758	15518	279119	279660	26804.10	32140.80	32479.40	-159.10	-158.79	-8.83
500	5000	305	273	48605	48664	3403.50	6507.70	6564.56	-159.46	-159.27	1.12
3000	5000	1831	9071	291238	291674	30512.60	42217.25	42495.30	-159.29	-159.06	-4.95
4500	5000	2747	23234	436726	437409	55495.90	64979.45	65884.40	-159.26	-159.01	-8.46

Contexts |G| < |M|. Table 3 presents the data collected in simulations to settings where the number of objects is less than the number of attributes. This is considered the worst situation for the BDD, since there is a small amount of objects, resulting in a low probability of finding objects with similar characteristics. Thus, the BDD is unable to simplify enough its representation to overcome the performance of the bits table. The representation of the bits table is still very compact in order to verify a representativeness gain of the BDD version. A similar behavior of Figure 2 can be observed in Figure 3. In extreme values of densities, the BDD performance had a slightly improvement over intermediate densities. However, the gains were not enough for the memory space consumed by the BDD representation suppress the bits table representation. Again, the bit table representation was extremely compact. The expressive number of attributes increases the BDD graph depth and requires more nodes to be represented.

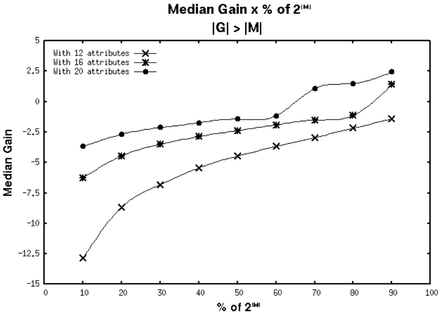


Fig. 4. BDD gain for contexts |*G*| = |*M*|

Table 4. BDD simulation for contexts |*G*| > |*M*|

|G|	|M|	S_{table}	E_{bdd} (Kb)			T_{bdd} (s)			Gain		
		(Kb)	Min	Median	Max	Min	Median	Max	Min	Media	Max
102	10	0	1	2	3	0.00	0.00	0.00	-22.52	-20.08	-10.87
613	10	1	2	4	5	0.01	0.01	0.01	-6.16	-5.56	-2.27
920	10	1	1	3	3	0.01	0.01	0.01	-2.42	-2.23	-1.15
409	12	1	3	8	8	0.00	0.01	0.01	-14.03	-12.85	-5.55
2457	12	4	3	13	14	0.03	0.03	0.03	-3.88	-3.66	1.08
3685	12	5	4	8	8	0.04	0.04	0.04	-1.53	-1.41	1.53
1638	14	3	7	24	27	0.02	0.02	0.03	-9.57	-8.73	-2.46
9829	14	17	6	42	45	0.13	0.13	0.14	-2.66	-2.52	2.82
14744	14	25	7	24	27	0.19	0.20	0.21	-1.05	1.04	3.49
6553	16	13	13	80	88	0.10	0.12	0.13	-6.90	-6.25	-1.01
39321	16	77	10	145	160	0.77	0.78	0.92	-2.08	-1.89	7.90
58981	16	115	13	80	88	1.19	1.21	1.22	1.31	1.44	8.96
26214	18	58	66	271	298	0.61	0.71	0.77	-5.18	-4.70	-1.14
157285	18	346	55	515	582	4.70	4.97	5.01	-1.68	-1.49	6.25
235928	18	518	65	270	298	7.32	7.37	7.39	1.74	1.92	7.93
104857	20	256	248	938	1033	3.71	4.54	4.79	-4.03	-3.66	1.03
629145	20	1536	225	1759	1962	28.09	30.13	43.25	-1.28	-1.15	6.84
943717	20	2304	267	936	1034	39.58	40.26	40.33	2.23	2.46	8.62

Contexts |G| > |M|. The data shown in Table 4 reflects the simulations realized with a wider number of objects than attributes. For each type of context, the selected amount of objects was 10%, 60% and 90% of the maximum number of objects possible ($2^{|M|}$).

It is noticed by Figure 4 that the gains and losses were not significant when a large amount of objects are used. The more the number of objects cover the total objects universe of possibilities ($2^{|M|}$), better will be the BDD representativeness. This is the case when 90% of the maximum number of objects is used. Also, BDD presented a stable behavior with fewer variations in the minimum, median and maximum gain for this type of context. With fewer attributes and many objects, BDD may become an attractive alternative to express data contained in cross tables. Also, the computational time required to assemble the BDD graph is not a limiting factor, allowing the construction of a BDD with 943,717 objects in times around 40 seconds. However, this situation is only reflected in context with few attributes. Increase the amount of attributes in the BDD has serious consequences in its size.

Many-Valued Contexts. Table 5 shows the data collected for many-valued contexts concerning attributes ranging from 1 to 5, where each attribute was simulated with 5, 10 and 15 intervals of discretization. All contexts used in these simulations have only one incidence per attribute, i.e. only one attribute-value by attribute. The amount of objects considered in the simulations is shown in Figure 5 along with the considered density. As it can be seen through Table 5, is possible to assemble a BDD context with more than 700,000 objects in approximately 13 minutes. The gains obtained in the data presented in Table 5 show that BDD has a satisfactory memory performance for this type of context. The density of this type of context is naturally lower, allowing the BDD representation to find more simplifications and be represented in a more compact form.

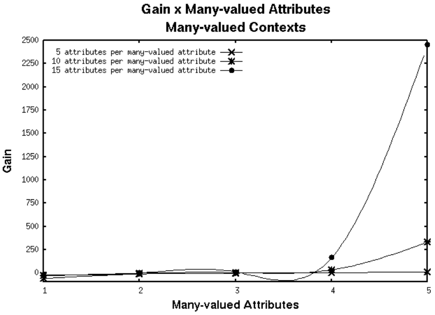


Fig. 5. BDD gain for many-valued contexts

Table 5. BDD simulation for many-valued contexts

$\|G\| = \|V\|^{\|M\|}$	$\|M\|$	$\|V\|$	$\|M\|\|V\|$	Den	S_{tabela} (Kb)	S_{bdd} (Kb)	T_{bdd} (s)	Gain
5	1	5	5	20%	0	0	0.00	-60.00
25	2	5	10	20%	0	0	0.00	-11.61
125	3	5	15	20%	0	0	0.00	-2.31
625	4	5	20	20%	1	0	0.01	2.17
3125	5	5	25	20%	10	0	0.08	10.85
10	1	10	10	10%	0	0	0.00	-31.67
100	2	10	20	10%	0	0	0.00	-3.04
1000	3	10	30	10%	4	1	0.04	3.29
10000	4	10	40	10%	49	1	0.72	32.89
100000	5	10	50	10%	610	2	24.86	328.95
15	1	15	15	6%	0	0	0.00	-20.71
225	2	15	30	6%	0	1	0.01	-1.38
3375	3	15	45	6%	19	2	0.26	10.91
50625	4	15	60	6%	371	2	17.54	163.66
759375	5	15	75	6%	6952	3	759.04	2454.88

This section presented the assessment of BDD as a representation of formal contexts. It was observed that the BDD has a satisfactory performance only on context with fewer attributes and a large amount of objects, i.e., when the number of objects covers much of the maximum number of objects possible ($2^{|M|}$). Unfortunately, to achieve this exorbitant amount of objects, the number of attributes must be very small. Moreover, as mentioned, the performance of BDD deteriorates as the number of attributes increases. As the level of depth in the BDD graph increases, less simplification are found to reduce its size. The construction of the BDD is also affected when the time for its assembly grows exponentially when more attributes are expressed in the context. In addition, better results can be obtained in contexts with densities closer to the minimum and maximum values than in intermediate values.

It is important to emphasize that the FCA derivation operator, necessary to obtain the formal concepts, is applied on the context expressed in BDD. Therefore, the more satisfactory is the performance of BDD, smaller will be the computational time required to operate it. For this reason, the concepts extraction should take advantage of this situation.

4 Formal Concepts Extraction Using BDD

4.1 Concept Extraction Algorithm Implementation Using BDD

In order to use a BDD representation of formal context, algorithms to extract concepts and/or to construct the concept lattice available in the literature must be adapted to handle this new form of representation. To demonstrate the feasibility of BDD, the adapted algorithm was the Attribute Intersections [10]. Note that the purpose of this paper is to evaluate the feasibility of BDD and not its most efficient implementation over several others algorithms.

The implementation of the Attribute Intersection algorithm in BDD was divided in three primary stages (Fig. 6). In the first stage, the construction of the formal context in BDD is made (Algorithm 1). The second stage is responsible to extract the set of all concepts from the BDD context. In the final stage, it is necessary to identify the attributes and objects from the concepts represented in BDD.

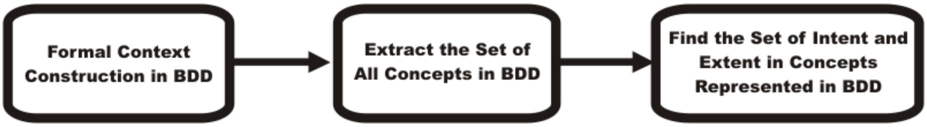

Formal Context Construction in BDD
Extract the Set of All Concepts in BDD
Find the Set of Intent and Extent in Concepts Represented in BDD

Fig. 6. Steps to implement the Attribute Intersection algorithm in BDD

Extracting the Set of All Concepts in BDD. Algorithm 2 is the kernel of the Attribute Intersection algorithm, but slightly modified to work with BDD. This implementation in BDD takes advantage of two distinct moments when the derivation operator is used (Line 4) and the intersection between two concepts is made (Line 8). The derivation operator is easily implemented through the implicit *bdd_ithvar* operator, which obtains a BDD representation of all objects that has an attribute. The intersection between two concepts is also implemented through an implicit BDD operation. In this case, the conjunction operator is represented in the algorithm as "&" but implemented as *bdd_and*. Moreover, the concepts list in this algorithm was implemented as a Hash to achieve a faster verification of concepts duplicity.

Algorithm 2. BDD construction based on the context.

```
in:  BDD context
out: List<BDD> concepts
concepts = new List<BDD>
concepts.addConcept(context)
for i=0; i<attributes; i++ do
  BDD tmp1 = context & bdd_ithvar(i)
  size = concepts.size()
  for j=0; j<size; j++ do
    BDD tmp2 = concepts.getConcept(j)
    BDD intersection = tmp1 & tmp2
    if !concepts.exist(intersection) then
      concepts.add(intersection)
    endif
  done
done
```

Unfortunately, storing all the concepts as BDD in the list reflects a very expressive memory consumption. The algorithm was slightly modified to save the concept intent (B_i) rather than the concept (A_i, B_i) in BDD. From the intent set (B_i), one can rebuild the concept in BDD through the formal context, thereby maintaining the essence of the proposed Algorithm 2.

Finding the Set of Intent and Extent in Concepts Represented in BDD. This section shows how to obtain the extent and intent of these concepts represented in BDD. Algorithm 3 is used to check if all objects represented by the BDD share a common attribute. Algorithm 4 is used to verify whether or not an object is present in the BDD.

Algorithm 3. Verify the presence of an attribute in a concept represented in BDD.

```
in:  BDD concept, attr
out: presence
BDD tmp = concept & bdd_ithvar(attr)
if tmp == concept then
  present = true
else
  present = false
endif
```

For the extraction of all objects (extent) of the concept, Algorithm 4 can be used to verify if each object that exists in the formal context is present in the concept. The same can be applied to the set of attributes (intent), through Algorithm 3, covering all formal context attributes checking whether or not they are present in the concept represented in BDD.

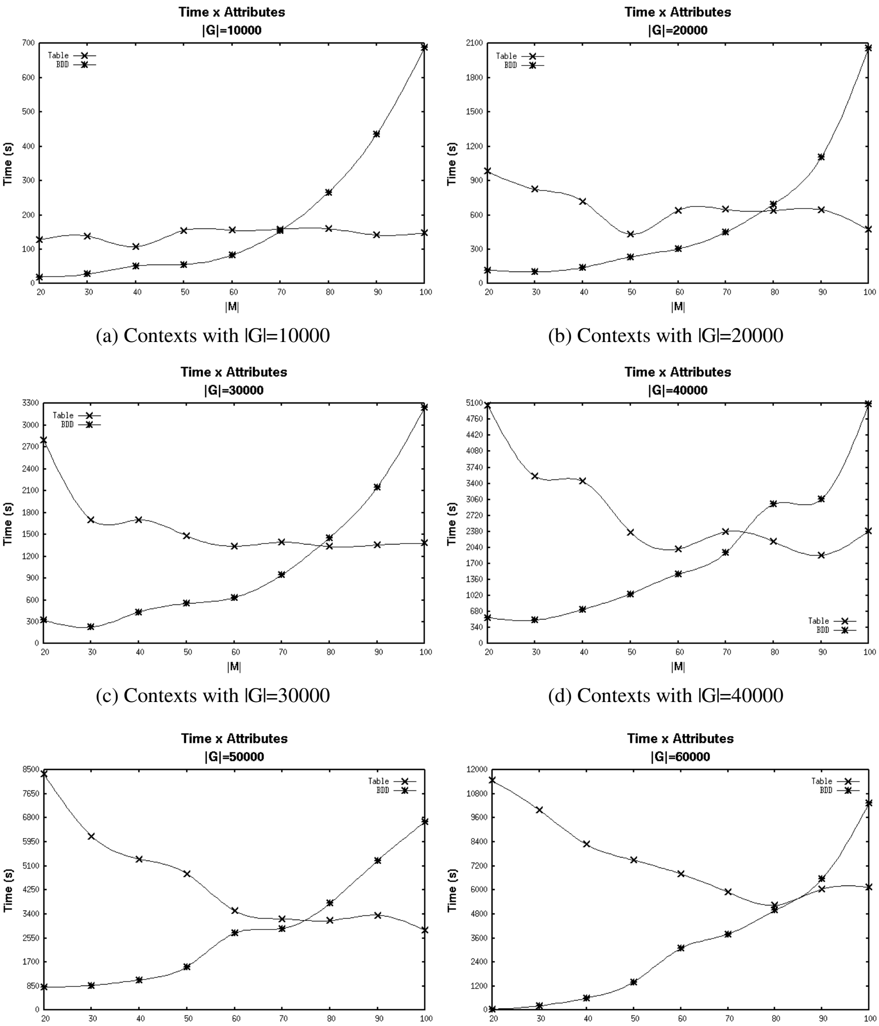
Algorithm 4. Verify the presence of an object in a concept represented in BDD.

```
in:  BDD concept, objc
out: presence
BDD tmp = concept
for i=0; i<objc.attributes; i++ do
  if tmp == bddtrue then
    presence = true
    return
  else if tmp == bddfalse then
    presence = false
    return
  endif
  if bdd_varlevel(tmp) == i then
    if obj.hasAttribute(i) then
      tmp = bdd_high(tmp)
    else
      tmp = bdd_low(tmp)
    endif
  endif
done
presence =(tmp == bddtrue)
```

4.2 Feasibility Analysis of BDD to Extract Concepts

One of the requirements to assess the representativeness of BDD to extract concepts was to compare its performance under the same conditions as its tabular version. For this reason, it was decided to implement a unique algorithm for both situations: contexts represented by BDD and by a table. As previously mentioned, the algorithm selected was the Attribute Intersections. This algorithm choice was driven by its inherent characteristics that allow a more effectively concepts extraction from contexts where the number of objects is superior to the attributes.

To create a more reliable simulation environment, both versions of the algorithm were constructed sharing the same types of strategies. The BDD version was constructed according to the diagram in Figure 6, while the tabular version was constructed with several optimizations. Both of them uses a list that holds concepts intent as a hash-table and shares the same hash function. The BDD version has an intrinsically feature that, when there is an intersection between two other concepts, the result is already a concept; while in the tabular version it is necessary to further use derivation operators to acquire the concept. To overcome this problem, the concept is obtained only after the verification of if its intent is not present in the list yet. Thus, the tabular version of the algorithm avoids unnecessary derivation operations and maintains similarity to the BDD version. Another feature was the implementation in the conventional version of the algorithm: the derivation operator uses a data structure similar to a *BitSet*. The efficiency of the operators becomes superior by decreasing the amount of comparisons between two sub-sets of concept extents. In this sense, various enhancements aimed at a more rapid extraction of concepts in order to achieve a more effective comparison of the BDD viability.



(a) Contexts with |G|=10000

(b) Contexts with |G|=20000

(c) Contexts with |G|=30000

(d) Contexts with |G|=40000

(e) Contexts with |G|=50000

(f) Contexts with |G|=60000

Fig. 7. Evaluation of Attribute Intersections implemented as a table and BDD

Another difference between the two versions evaluated is related to how each of them carries out their intersections. The BDD version performs the intersection between concepts represented in BDD through the implicit *bdd_and* operator, while the tabular version performs the intersection between the previously computed concepts extents. After that step, both algorithms must identify the concepts intent. The BDD version takes advantage of this situation because of its extremely efficient *bdd_ithvar* operator, but looses in performance in stage 3 of the diagram in Figure 6. The table version is not affected by this problem since it obtains the concept intent and extent through the derivation operators. Thus, several simulation scenarios are necessary to evaluate the algorithm behavior over different conditions.

Figure 7 shows the behavior of the Attribute Intersections algorithm for the BDD and tabular version for contexts with fewer attributes (20 to 100) and many objects (10,000 to 60,000). This algorithm has better performance for contexts $|G|>|M|$. To ensure that the BDD graph would not be extremely compact, the used density for all contexts was the minimum plus 10% of it. Moreover, lower density values result into smaller amounts of concepts, thus making the simulations consume less time to execute. All simulations were realized on a Pentium Dual Core 2.66Ghz with 2Gb of RAM running Linux Slackware Linux 12.0. The implementations for both versions of the Attribute Intersections algorithm were implemented in C++.

As it can be seen in Figure 7, the implementation of the algorithm in BDD had an exponential performance in all the simulations, while the tabular version has presented an irregular decreasing behavior. For the table version, the density can explain the decreasing behavior, since lower attributes value had higher density for these considered simulations. In addition to that, how the incidences are spread into the context can explain its irregular behavior. Different contexts with the same density may have different execution performance. On the other hand, the BDD version presented a stable exponential behavior. Increasing the quantity of attributes in the context, more nodes will be required to construct the BDD. Therefore, as more nodes are used by the BDD, less efficient will be the operations in this representation. Thus, explaining this uniform behavior. Also, as can be seen by simulations of 20 and 30 attributes, while the tabular version had worst time performance, this BDD maintained a very low execution time, despite of the higher density. So, the BDD size is extremely relevant in the computation of all concepts.

Through simulation, it is demonstrated that BDD has a better overall performance than the table version for a number of attributes lower than or equal to 70. Above this threshold, the BDD graph becomes complex and begins to turn into an unattractive solution. Also, as the amount of objects increases, greatest has become the difference between the execution times of both implementations, considering attributes up to 70. Thus, the implicit representation of concepts in BDD becomes an alternative to a more efficient extraction of concepts in these conditions.

Considering now a threshold of 70 attributes, another simulation scenario was created. This time, the number of objects chosen was based on the ICFCA'06 challenge. A many-valued context was simulated with 7 attributes, a fixed number of 10 attribute-values per attribute and 120,000 objects. The context had a density of 10% and generated 1,172,960 concepts. Table 6 presents the spent time consumed by both algorithm implementations, in BDD and in table.

Table 6. Execution time for many-valued context with 70 attributes and 120000 objects

	Construction of the Context (s)	Concepts Extraction (s)	Intent and Extent Identification (s)	Total (s)	Total
Table	-	-	-	251283	2d 21:48:03
BDD	128	18345	33289	51762	0d 14:22:42

As it can be seen in Table 6, the BDD version obtained the set of all concepts in less than 15 hours, while the tabular version demanded almost three days for its complete execution. Applicability to process larger contexts could be achieved with the use of a distributed version of an algorithm implemented in BDD. If we consider that a context with 70,000 attributes and 120,000 objects can be divided into sub-contexts of 70 attributes and 120,000 objects, still maintaining a low density, then it would be necessary, in general, 15 thousands of execution hours. Considering that all sub-contexts were executed in execution times around 15 hours. If a cluster of 50 computers were used, then it would be required around 300 execution hours, about 15 days. It will be still necessary to join the sub-concepts to form the concepts final set, but the BDD opens a possibility to process this large contexts.

Note that the required time to identify the set of extents from the concepts represented in BDD was very significant, as seen by Table 6. This happens because of the used algorithm quadratic complexity relative to the number of objects. If more efficient algorithms were used, lower computational times may be achieved to process contexts. Instead of a brute force strategy to check objects presence in a concept, another strategy could be visiting BDD nodes identifying the objects, inverted form.

5 Conclusions

The present work is related to a challenge raised at the ICFCA'06 conference, which refers to the manipulation of large formal contexts. Through the use of an implicit representation of formal context in BDD, it has been demonstrated that this new representation became computationally feasible for handling large contexts, when compared to the conventional manipulation of a table.

In this work, the representation of the formal contexts in BDD were evaluated in two distinct aspects, as the memory consumption in relation to a bit table and as the computational time spent in its construction. It was later assessed the performance of the algorithm Attribute Intersection adapted to be used with BDD compared to the conventional implementation as a table. It has been verified that the BDD can be applied to the FCA algorithms to improve the execution time required to complete the extraction of all concepts. Although this representation allows the manipulation of contexts with a large number of objects, it is restricted to contexts with few attributes (up to 70 attributes, as experimentally verified). This is due to the fact that BDD tends to improve their representation with a larger number of objects, allowing further simplifications on its graph and thus making the operations on it more efficient. It has been also realized that the lower the number of attributes in the context the higher will be the BDD performance when compared to the conventional implementation of the algorithm, as verified in Figure 7. Thus, if the context meets this feature, a significant efficiency can be achieved with the application of this new alternative. This can also be verified for many-valued contexts in Table 6, in which the difference between both execution times was approximately of 2 days of uninterrupted processing.

The context density is an aspect that is intimately related to the number of concepts obtained. The concepts extraction using a BDD representation is still conditioned to this characteristic. Therefore, all simulations were limited to low densities.

Several future works may be pointed out: Evaluating new libraries for BDDs construction and manipulation, like CUDD [11]; measuring the behavior over different BDD technologies, like ZBDDs; evaluate different orders for attributes (statically or dynamically chosen) to construct the BDD; and adapting others FCA algorithms that could be used with BDD. ZBDD have already proven to be satisfactory for spare contexts [5], but in some type of contexts, in our preliminary results, the standard BDD was able to beat the ZBDD performance. Further analysis is therefore required. Also, in order to adapt others FCA algorithms a study must be conducted to verify whenever the BDD can be applied. In other words, which algorithm operations can be similarly replaced by a BDD operation in order to increase the algorithms capabilities. For example, in this work, the intersection of concepts in the Attribute Intersection was implemented by replacing this function with a correlated BDD conjunction operator that enabled performance improvements.

Although the results presented in this paper have been shown to be satisfactory for many objects (120,000) and a few attributes (in the order of 70), it is possible to use the BDD approach in conjunction with distributed FCA algorithms. Thus increasing processing power of contexts with larger number of attributes while still maintaining its inherent capability of processing huge amounts of objects.

References

1. Li, Y., Liu, Z.T., Shen, X.J., Wu, Q., Qiang, Y.: Theoretical research on the distributed construction of concept lattices. In: International Conference on Machine Learning and Cybernetics, 2003, vol. 1, pp. 474–479 (2003)
2. Liu, Z., Li, L., Zhang, Q.: Research on a union algorithm of multiple concept lattices. In: Wang, G., Liu, Q., Yao, Y., Skowron, A. (eds.) RSFDGrC 2003. LNCS, vol. 2639, pp. 533–540. Springer, Heidelberg (2003)
3. Lévy, G., Baklouti, F.: A distributed version of the ganter algorithm for general galois lattices. In: CLA 2005, pp. 207–221 (2005)
4. Bryant, R.: Graph-based algorithms for boolean function manipulation. IEEE Transactions on Computers C-35(8), 677–691 (1986)
5. Yevtushenko, S.: Computing and Visualizing Concept Lattices. PhD thesis, TU Darmstadt, Fachbereich Informatik (2004)
6. Minato, S.: Zero-suppressed BDDs for set manipulation in combinatorial problems. In: DAC 1993: Proceedings of the 30th International Conference on Design Automation, pp. 272–277. ACM, New York (1993)
7. Grätzer, G.: General Lattice Theory. Birkhäuser, Basel (1978)
8. Butler, K.M., Ross, D.E., Kapur, R., Mercer, M.R.: Heuristics to compute variable orderings for efficient manipulation of ordered binary decision diagrams. In: DAC 1991: Proceedings of the 28th Conference on ACM/IEEE Design Automation, pp. 417–420. ACM, New York (1991)
9. Lind-Nielsen, J.: Buddy: A binary decision diagram. Technical report, Department of Information Technology, Technical University of Denmark, Lyngby, Denmark (1999), http://www.itu.dk/research/buddy
10. Carpineto, C., Romano, G.: Concept Data Analysis: Theory and Applications. John Wiley & Sons, Indianapolis (2004)
11. Somenzi, F.: CUDD: CU decision diagram package release (1998)

A Novel Approach to Cell Formation*

Radim Belohlavek[1,2], Niranjan Kulkarni[1], and Vilem Vychodil[1,2]

[1] Dept. Systems Science and Industrial Engineering
T. J. Watson School of Engineering and Applied Science
Binghamton University–SUNY, PO Box 6000, Binghamton, NY 13902–6000, USA
rbelohla@binghamton.edu,nkulkar1@binghamton.edu,vychodil@binghamton.edu
[2] Dept. Computer Science, Palacky University, Olomouc
Tomkova 40, CZ-779 00 Olomouc, Czech Republic

Abstract. We present an approach to the cell formation problem, known from group technology, which is inspired by formal concept analysis. The cell formation problem consists in allocating parts (objects) to machines (attributes), based on the machine-part matrix. This can be viewed as forming groups consisting of a set of parts and a set of machines. Such groups resemble formal concepts in the input data. Due to the specific nature of the performance assessment in the cell formation problem, good groups can be thought of as rectangles which, unlike those corresponding to formal concepts, contain a few blanks, i.e. which are not full of crosses in terms of formal concept analysis. Moreover, such groups need to be disjoint both in terms of objects and attributes. In this paper, we present an algorithm for the cell formation problem, experimental results, and a comparison to some methods proposed in the literature.

1 Introduction

Group technology (GT) is an approach to manufacturing management which capitalizes on grouping of products with similar manufacturing characteristics. Conceived originally in the 1940s in the Soviet Union, it has since been developed and used in numerous countries [10,15]. There are several benefits from applying GT and they are discussed in e.g. [3,15,18].

One particular application of GT is cellular manufacturing (CM) [2,12,18]. CM involves grouping of machines or processes into manufacturing cells and operation of manufacturing cells. Such grouping is based on parts or part families processed by the machines. This makes CM different from a traditional jobshop environment in which machines are grouped according to their functional similarities [8]. The companies surveyed in [17] reported several benefits from implementing CM, including setup time reduction, material handling cost reduction, equipment cost and labor cost reduction, improvement in quality, improvement in material flow, machine and space utilization, and improvement in employee morale.

* Supported by grant No. 1ET101370417 of GA AV ČR and by institutional support, research plan MSM 6198959214.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 210–223, 2009.

One of the first problems encountered in implementing CM is the cell formation (CF) problem which consists in grouping of machines into cells, grouping of parts into families, and assignment of the part families to machine cells, so that machine utilization is high and inter-cellular movement is low. Several other constraints need often be considered, such as safety and technological requirements regarding the location of machines, maximum size of cells and maximum number of cells specified by a user, requirements regarding the capacity of machines, or the need for designing flexible cells, but the principal concern is machine utilization and inter-cellular movement [8,9,18].

Several approaches to the CF problem were proposed in the literature. [2,18,12,9,16] provide overviews of these approaches. [13] identifies numerous approaches and classifies them according to their methodology into the following classes:

- descriptive procedures (they include informal methods based on rules of thumb or visual inspection, as well as formal methods based on part coding and classification),
- methods based on cluster analysis (both hierarchical and non-hierarchical clustering algorithms are utilized in these methods),
- methods based on graph partitioning,
- methods based on artificial intelligence techniques (a variety of techniques underlies these approaches, such as rule-based knowledge systems, pattern recognition, and artificial neural networks),
- methods based on mathematical programming (particularly, linear and quadratic programming, and dynamic programming).

The evaluations available in the literature, see e.g. [10,13] suggest that there is no clear winner among the proposed approaches to the CF problem, as the approaches perform differently on different types of datasets.

In this paper, we present a novel approach to the CF problem. The approach is inspired by formal concept analysis (FCA) [5,7]. FCA identifies particular clusters, called formal concepts, in the input data which consists of objects, attributes, and an incidence relation between them. The main idea of our approach consists in linking the conceptual framework of CF to the notions of FCA in a way in which parts correspond to objects, machines correspond to attributes, and the part-machine relationship, indicating which parts need to be processed on which machines, is represented by the incidence relation. Doing so, the formal concepts in the input data obtained using such link can naturally be interpreted as cells whose machines correspond to the attributes of the concept intent and whose parts correspond to the object of the concept extent. In terms of CF, the original methods of FCA can recognize only cells with full machine utilization and allow for overlapping cells. In order to fit the requirements of CF, we thus modify the notions of FCA and develop an algorithm that extracts a set of formal concepts from the part-machine data which can be interpreted as a set of cells provided as a solution to the CF problem. We demonstrate by comparison to other methods described in the literature on several benchmark datasets that our algorithm performs well both in terms of machine utilization and inter-cellular movement.

The paper is organized as follows. In Section 2.1, we define the cell formation problem. Section 2.2 reviews basic notions from formal concept analysis and links them to the conceptual framework of cell formation. Our method for cell formation based on formal concept analysis is presented in Section 3. Section 4 presents an experimental evaluation of the proposed method including a comparison to other methods proposed in the literature. Section 5 contains conclusions and directions for future research.

2 Cell Formation Problem and Formal Concept Analysis

2.1 Cell Formation Problem

Let

$$X = \{M_1, \ldots, M_n\}$$

be a set of machines,

$$Y = \{P_1, \ldots, P_m\}$$

be a set of parts,

$$I \subseteq X \times Y$$

be a machine-part incidence relation with the following interpretation:

$$\langle M, P\rangle \in I \quad \text{iff} \quad \text{part } P \text{ needs to be processed on machine } M.$$

The triplet $\langle X, Y, I\rangle$ can be depicted by a table in which rows correspond to machines, columns correspond to parts, and a table entry is black or white depending on whether $\langle M, P\rangle \in I$ or $\langle M, P\rangle \notin I$. Such table is called a *part-machine matrix* in the cell formation problem.

A *cell* in $\langle X, Y, I\rangle$ is a pair $\langle A, B\rangle$ of a set $A \subseteq X$ of machines and a set $B \subseteq Y$ of parts. The *cell formation problem* (CF problem) can be described as follows.

Definition 1. *A* ***solution*** *to the CF problem is a set*

$$\mathcal{S} = \{\langle A_1, B_1\rangle, \ldots, \langle A_k, B_k\rangle\} \tag{1}$$

of cells for which

1. *$\{A_1, \ldots, A_k\}$ forms a partition of the set X of machines,*
2. *$\{B_1, \ldots, B_k\}$ forms a partition of the set Y of parts.*

That is, (1) is a solution if

1. for each $l = 1, \ldots, k$: $A_l \neq \emptyset$ and $B_l \neq \emptyset$,
2. for $i, j = 1, \ldots, k$, $i \neq j$: $A_i \cap A_j = \emptyset$ and $B_i \cap B_j = \emptyset$,
3. $A_1 \cup \cdots \cup A_k = X$ and $B_1 \cup \cdots \cup B_k = Y$.

Given $\langle X, Y, I\rangle$, the following two objectives need to be achieved by any solution $\mathcal{S}$ which is considered to be a “good solution”:

1. *high machine utilization* within cells, which means that for each cell $\langle A_l, B_l\rangle \in \mathcal{S}$, the number of machine-part pairs $\langle M_i, P_j\rangle$ in this cell (i.e. pairs $\langle M_i, P_j\rangle \in A_l \times B_l$) for which P_j needs to be processed on M_i (i.e. $\langle M_i, P_j\rangle \in I$) is (relatively) high;
2. *low percentage of exceptional elements*, which means that the number of pairs $\langle M_i, P_j\rangle \in I$ for which M_i belongs to a different cell than P_j (i.e. for each $l = 1, \ldots, k$: $\langle M_i, P_j\rangle \notin A_l \times B_l$) is (relatively) low.

Evaluations of goodness of a solution are usually based on some variants of the following functions.

Definition 2. *Given a part-machine matrix represented by* $\langle X, Y, I\rangle$ *and a solution* (1)*, we define*

1. *the* machine utilization $MU(\mathcal{S})$ *of* $\mathcal{S}$ *by*

$$MU(\mathcal{S}) = \frac{1}{k}\sum_{l=1}^{k}\frac{|(A_l \times B_l) \cap I|}{|A_l| \cdot |B_l|}, \tag{2}$$

2. *the* percentage of exceptional elements $PE(\mathcal{S})$ *of* $\mathcal{S}$ *by*

$$PE(\mathcal{S}) = \frac{|I - \bigcup_{l=1}^{k} A_l \times B_l|}{m \cdot n}. \tag{3}$$

Given a weight $w \in [0,1]$*, the* grouping efficiency $GE(\mathcal{S}, w)$ *of* $\mathcal{S}$ *is defined by*

$$GE(\mathcal{S}, w) = w \cdot MU(\mathcal{S}) - (1 - w) \cdot PE(\mathcal{S}). \tag{4}$$

Instead of (3), one sometimes uses

$$PE(\mathcal{S}) = \frac{|I - \bigcup_{l=1}^{k} A_l \times B_l|}{m \cdot n - \sum_{l=1}^{k} |A_l| \cdot |B_l|}. \tag{5}$$

Note that $MU(\mathcal{S})$ is the average machine utilization per cell, given that a machine utilization of a cell is the percentage of entries in a cell which are black. $PE(\mathcal{S})$ given by (3) is the percentage of black entries in the collection of entries which do not belong to any cell.

2.2 Formal Concept Analysis

We refer to [7] and [5] for information on formal concept analysis (FCA). We denote a formal context by $\langle X, Y, I\rangle$, i.e. $I \subseteq X \times Y$ (object-attribute data table, objects $x \in X$, attributes $y \in Y$); the concept-forming operators by $^{\uparrow}$ and $^{\downarrow}$, i.e. for $A \subseteq X$, $A^{\uparrow} = \{y \in Y \mid \text{for each } x \in A : \langle x, y\rangle \in I\}$ and dually for $^{\downarrow}$; a concept lattice of $\langle X, Y, I\rangle$ by $\mathcal{B}(X, Y, I)$, i.e. $\mathcal{B}(X, Y, I) = \{\langle A, B\rangle \in 2^X \times 2^Y \mid A^{\uparrow} = B, B^{\downarrow} = A\}$.

3 Proposed Method

This section describes our approach to find a solution of a given cell-formation problem which meets certain quality criteria. The criteria are formulated using suitable measures. The basic idea of our approach can be summarized as follows: (1) Define a function f which measures quality of a cell in a given context; (2) take a formal context $I \subseteq X \times Y$ representing the part-machine relationship; (3) take sets $A \subseteq X$ and $B \subseteq Y$ which maximize f; (4) output $\langle A, B\rangle$ and repeat step (3) until all objects and attributes are covered. Thus, we follow a greedy approach utilizing the measure f.

The greedy approach sketched above may end up in a situation where all machines (parts) are covered by the discovered cells and some of the parts (machines) are not. In such a case, we are not able to form a solution from the discovered cells because one of the conditions 1. and 2. of Definition 1 is not satisfied. We therefore relax the notion of a solution as follows.

Definition 3. *A cell $\langle A, B\rangle$ where exactly one of the sets A and B is empty is called a* ***degenerate cell****. An* ***admissible solution*** *to the CF problem is any set $\mathcal{S} = \{\langle A_1, B_1\rangle, \ldots, \langle A_k, B_k\rangle\}$ of cells such that*

1. *$\mathcal{S}$ contains at most one degenerate cell,*
2. *for $i, j = 1, \ldots, k$, $i \neq j$: $A_i \cap A_j = \emptyset$ and $B_i \cap B_j = \emptyset$, and*
3. *$A_1 \cup \cdots \cup A_k = X$ and $B_1 \cup \cdots \cup B_k = Y$.*

The algorithm can be formalized as follows:

```
FindCells($I, f$)
1  $U \leftarrow X; V \leftarrow Y; \mathcal{C} \leftarrow \emptyset$
2  while $U \neq \emptyset$ and $V \neq \emptyset$
3     do $\langle A, B\rangle \leftarrow$ FindBestCell($I, U, V, f$)
4        $\mathcal{C} \leftarrow \mathcal{C} \cup \{\langle A, B\rangle\}; U \leftarrow U - A; V \leftarrow V - B$
5  if $U \neq \emptyset$ or $V \neq \emptyset$
6     then
7           $\mathcal{C} \leftarrow \mathcal{C} \cup \{\langle U, V\rangle\}$;
8  return $\mathcal{C}$
```

The algorithm FindCells(I, f) first initializes sets U and V denoting the remaining objects and attributes which can be used to form cells. FindBestCell(I, U, V, f) at line 3 returns a new cell $\langle A, B\rangle$, i.e. $A \subseteq U$, $B \subseteq V$, which has a high value of f (preferably the highest one) among all possible cells formed from U and V in I. Obviously, FindBestCell(I, U, V, f) can be defined in many ways; some of them will be discussed later. Once a suitable cell $\langle A, B\rangle$ is found by calling FindBestCell(I, U, V, f), objects from A and attributes from B are removed from U and V (see line 4) which ensures that the next cell will not have an overlap with the cells computed in the previous steps. If if-then clause between lines 5–7 adds to $\mathcal{C}$ a degenerate cell $\langle U, V\rangle$ consisting of remanining machines U and parts V provided that $U \neq \emptyset$ or $V \neq \emptyset$. At the end of the computation, $\mathcal{C}$ contains an admissible solution.

Brute-Force Algorithm. We now focus on FindBestCell(I, U, V, f) which is the core of our algorithm. Since FindBestCell is supposed to find a cell which, in the ideal case, maximizes f, the best cell can be obtained by going through all possible subsets of U and V:

```
FindBestCell$_1$($I, U, V, f$)
$s \leftarrow -\infty$
for $C \in 2^U$
   do for $D \in 2^V$
          do if $s < f(I, U, V, C, D)$
                then
                     $s \leftarrow f(I, U, V, C, D)$; $\langle A, B\rangle \leftarrow \langle C, D\rangle$
return $\langle A, B\rangle$
```

Needless to say, such an algorithm has an exponential time complexity. However, it can be applied to some of the small real-world problems presented in the literature.

In what follows we focus on variants of FindBestCell$_1$ which do not go through the space of all possible subsets of U and V but only through a smaller portion which contains promising cells (i.e., cells with high values of f).

Algorithm Using Formal Concepts. The number of cells which are calculated during a single call of FindBestCell$_1$(I, U, V, f) can be reduced if we use formal concepts as cells. This is based on the idea that any cell $\langle A, B\rangle$ which is a formal concept has a full machine utilization and it is a maximal cell containing $\langle A, B\rangle$ with this property. This is due to the fact that formal concepts in I correspond to maximal rectangles in I containing black entries only. The corresponding modification of FindBestCell$_1$ can be formalized as follows:

```
FindBestCell$_2$($I, U, V, f$)
1  $s \leftarrow -\infty$
2  for $\langle C, D\rangle \in \mathcal{B}(X, Y, I)$
3     do $C \leftarrow C \cap U$; $D \leftarrow D \cap V$
4        if $s < f(I, U, V, C, D)$
5          then
6                $s \leftarrow f(I, U, V, C, D)$; $\langle A, B\rangle \leftarrow \langle C, D\rangle$
7  return $\langle A, B\rangle$
```

Note that after obtaining a formal concept $\langle C, D\rangle \in \mathcal{B}(X, Y, I)$ at line 2, the sets C and D are restricted to the objects and attributes from the sets of remaining objects U and attributes V only. (See line 3.) As we demonstrate in the next section, in several cases this method can deliver results which are nearly as good as the results obtained by the brute-force algorithm. In addition to that, if the best possible solution to the cell-formation problem exists, solution $\mathcal{S}$ for which $MU(\mathcal{S}) = 1$ and $PE(\mathcal{S}) = 0$, it will be always found:

Theorem 1. *If the cell-formation problem for $I \subseteq X \times Y$ has a solution such that $MU(\mathcal{S}) = 1$ and $PE(\mathcal{S}) = 0$, this solution can be found by* FindCells *combined with* FindBestCell$_2$*.*

Proof. Let f be a function such that $f(I, U, V, C, D) = 1$ if (i) $(C \times D) \cap I = C \times D$ (i.e., if $\langle C, D\rangle$ is a rectangle full of black entries) and (ii) $((U - C) \times D) \cap I = \emptyset$ and

$(C\times(V-D))\cap I=\emptyset$ (i.e., both $(U-C)\times D$ and $C\times(V-D)$ are empty rectangles), and $f(I,U,V,C,D)<1$ otherwise. (Such assumptions can be considered natural requirements for f.) It is easily seen that if $MU(\mathcal{S})=1$ and $PE(\mathcal{S})=0$, then for each cell $\langle C,D\rangle$ in the solution of the cell-formation problem, we have $f(I,X,Y,C,D)=1$ and $\langle C,D\rangle$ needs to be a maximal rectangle. Therefore, the first cell generated by the algorithm is one the cells contained in the solution. The rest follows directly by induction. □

Algorithm Using Dense Rectangles. If the best possible solution does not exist, i.e., if there is no admissible solution for a cell-formation problem such that $MU(\mathcal{S})=1$ and $PE(\mathcal{S})=0$, the method based on formal concepts may not yield optimal results. Intuitively, there may be interesting cells which have almost full machine utilization (i.e., $MU(\mathcal{S})$ is close to 1) and low percentage of exceptional elements. Such cell may be more useful than a cell with full machine utilization but a higher percentage of exceptional elements.

Therefore, we modify the "cells as concepts" approach to include cells formed of rectangles almost full of 1's. We find such rectangles by finding formal concepts first and then adding promising objects and attributes as long as the quality measure of the particular rectangle increases. This leads to a modified version of the algorithm described in the previous paragraph:

```
FindBestCell3($I,U,V,f$)
$s \leftarrow -\infty$
for $\langle E,F\rangle \in \mathcal{B}(X,Y,I)$
   do $E \leftarrow E\cap U; F \leftarrow F\cap V$
      repeat
         $C \leftarrow E; D \leftarrow F; r \leftarrow f(I,U,V,C,D)$
         select $M \in U-C$ that maximizes $f(I,U,V,C\cup\{M\},D)$
         select $P \in V-D$ that maximizes $f(I,U,V,C,D\cup\{P\})$
         if $f(I,U,V,C\cup\{M\},D) \leq f(I,U,V,C,D\cup\{P\})$
            then $q \leftarrow f(I,U,V,C,D\cup\{P\}); F \leftarrow D\cup\{P\}$;
            else $q \leftarrow f(I,U,V,C\cup\{M\},D); E \leftarrow C\cup\{M\}$;
      until $q < r$
      if $s < f(I,U,V,C,D)$
         then
               $s \leftarrow f(I,U,V,C,D); \langle A,B\rangle \leftarrow \langle C,D\rangle$
return $\langle A,B\rangle$
```

Compared to the previous algorithm, a new repeat∼until loop is added. The loop searches for remaining objects and attributes which can be added to the current rectangle (originally, a formal concept) and which increase (or do not decrease) the quality measure.

Quality Measures. The quality of solution found by FindCells combined with any of the variants of FindBestCell depends on our choice of the quality measure f. For given I, U, V, and $A\subseteq U$ and $B\subseteq V$, $f(I,U,V,A,B)$ is the measure of goodness of $\langle A,B\rangle$ in I. We consider quality measures which assign higher values to better cells. Although the notion of a "better cell" is subjective, we can agree that in certain situations there are "best cells" with full machine utilization (in the cell) and no exceptional elements (in the cell).

All the quality measures proposed below use percentages of machine utilization and exceptional elements in a single cell to compute the resulting value of f. We introduce functions $g(I, U, V, A, B) \in [0, 1]$ and $h(I, U, V, A, B) \in [0, 1]$ as follows:

$$g(I, U, V, A, B) = \frac{|(A \times B) \cap I|}{|A| \cdot |B|}, \tag{6}$$

$$h(I, U, V, A, B) = 1 - \frac{|((A \times (V - B)) \cup ((U - A) \times B)) \cap I|}{|A| \cdot |V - B| + |U - A| \cdot |B|}. \tag{7}$$

Clearly, $g(I, U, V, A, B)$ is the machine utilization of $\langle A, B\rangle$, i.e. the fraction of black entries in the rectangle $\langle A, B\rangle$. Analogously, $h(I, U, V, A, B)$ is the fraction of non-exceptional elements, i.e. the fraction of white entries in the rectangles $\langle A, V - B\rangle$ and $\langle U - A, B\rangle$. Obviously, if $\langle A, B\rangle$ is a cell with full machine utilization and no exceptional elements then $g(I, U, V, A, B) = h(I, U, V, A, B) = 1$.

Remark 1. In the rest of this section, we fix I, U, V and simplify the notation: For brevity, we write just $f(A, B)$, $g(A, B)$, $h(A, B), \ldots$ instead of $f(I, U, V, A, B)$, $g(I, U, V, A, B)$, $h(I, U, V, A, B), \ldots$

We can introduce two families of quality measures which are based on weighted arithmetic and geometric averages of $g(A, B)$ and $h(A, B)$. Namely, we define f_1^w and f_2^w as follows:

$$f_1^w(A, B) = w \cdot g(A, B) + (1 - w) \cdot h(A, B), \tag{8}$$

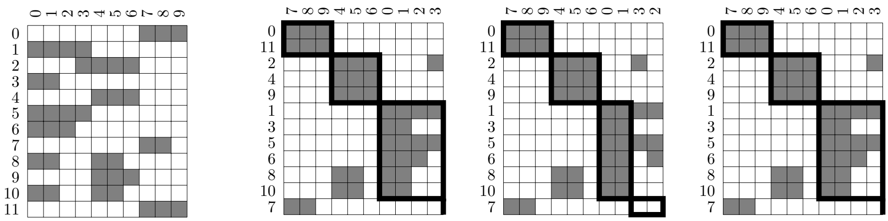
$$f_2^w(A, B) = g(A, B)^w \cdot h(A, B)^{\frac{1}{w}}. \tag{9}$$

These measures will be used in the next section.

4 Experimental Evaluation

In this section we present examples of solutions to sample cell-formation problems which we identified in the literature and provide a comparison with other approaches. First, let us note that qualifying a solution as "good" among admissible solutions is highly subjective. Usually, an expert judgment or additional knowledge is needed to select the best solution among several ones. Second, despite the computational complexity of the method we propose, the results are obtained with acceptable response times because the data sets that appear in the cell-formation problem domain are usually small (with $|Y|$ around 30 or less). In this section we focus on the effect of selecting various quality measures and variants of the FindBestCell algorithm.

Results Obtained by Variants of FindBestCell *Algorithm.* As discussed in the previous section, the algorithm can deliver the best solution possible if it exists. If not, the algorithm varies based on the choice of f and FindBestCell$_i$. If one



input machine-part data brute-force concepts dense rectangles

Fig. 1. Various quality measures used to solve the same cell-formation problem

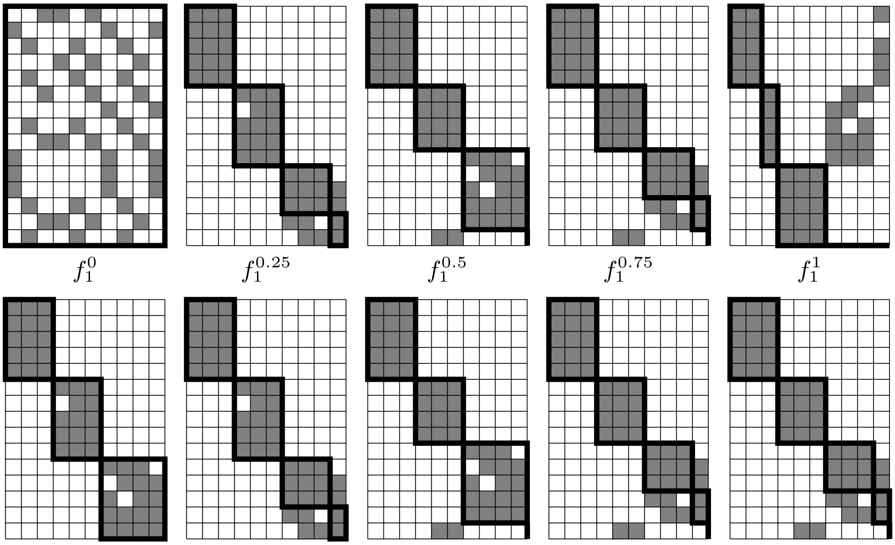
uses FindBestCell$_2$ using formal concepts as cells, we usually get an acceptable solution which however might not be natural for a user. For instance, in Fig. 1 we have the input data matrix (left), a brute-force admissible solution (second from left), and a solution found by taking "formal concepts as cells" (third from left) with f_2^1 as the quality measure. Note that the cells in Fig. 1 are depicted by thick rectangles drawn in data tables with permuted rows and columns. For example, the first cell in the second diagram consists of machines 0 and 11 and parts 7, 8, and 9.

For an industrial engineer, the second solution may seem not natural because it has more exceptional elements and it contains an "empty cell" with no machine utilization. Interestingly, if we use FindBestCell$_3$ which uses "dense rectangles", we obtain an admissible solution which is the same is in case of the brute-force algorithms (first from right). We have tested the algorithms on various artificial as well as real-world machine-part datasets and we have observed that the admissible solution using "dense rectangles" produces the same or almost as good results as the brute-force algorithm with considerable smaller demands.

Choice of Quality Measures. The choice of a quality measure, i.e. function f, seems to be crucial for finding a satisfactory solution. There seems to be no single measure which works well for all datasets because the problem of finding a "good solution" is subjective. In general, good choices seem to be $f_1^{0.5}$ (equal emphasis is put on utilization and exceptions) and f_2^w with lower values of w (tends to suppress exceptions). Sample results corresponding to various measures are depicted in Fig. 2.

In Fig. 2, we have used the "dense rectangles" to form the cells. Notice that f_1^0 is trivial because all emphasis is put on no exceptional elements with no emphasis on the utilization, i.e. a trivial solution is to have one cell covering the entire part-machine dataset. In a similar sense, f_1^1 produces degenerate solution as well because the high demand of utilization leaves large amount of exceptional elements and 4 machines are not contained in any cell. In case of this dataset, $f_2^{0.01}$ seems to produce the best solution.

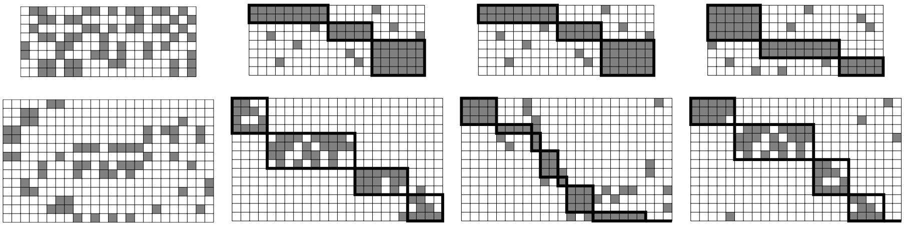
Illustrative Examples. We now show examples of (admissible) solutions to the cell-formation problem with datasets which we identified in the literature [1,6,14]. The solutions have been found using "dense rectangles" as cells and f_2^w taken



f_1^0 $f_1^{0.25}$ $f_1^{0.5}$ $f_1^{0.75}$ f_1^1

$f_2^{0.01}$ $f_2^{0.5}$ f_2^1 f_2^2 f_2^{100}

Fig. 2. Various quality measures used to solve the same cell-formation problem



input data $f_2^{0.2}$ f_2^1 f^*

Fig. 3. Various quality measures used to solve the same cell-formation problem

as the quality measure. In addition to that, we have used a quality measure which takes into account the size of cells. The motivation is the following: an engineer often follows not only the utilization but also the numbers of cells. As an extreme example, it does not make much sense to have as much cells as machines in the system, making each machine a separate cell. Therefore, we introduce the following measure:

$$f^*(A,B) = g(A,B) \cdot h(A,B)^2 \cdot \ln\Big(1 + \sqrt{|A| \cdot |B|}\Big), \tag{10}$$

which is similar to f_2^w except for it multiplies the result by the size of a possible cell, putting more emphasis on large cells. In order to avoid another extreme (having all machines in one or just a few cells), we adjusted the quality measure by the logarithm of the size of the edge of a possible cell. Fig. 3 contains results for two datasets using various quality measures.

Table 1. Comparing results from [14] with the proposed method for the 8 × 20 matrix from Fig. 3 (top)

Method	Cells	Cell 1 Utilization	Cell 2 Utilization	Cell 3 Utilization	Exceptional Elements	Voids
[14]	3	1	1	1	9	0
$f_2^{0.2}$	3	1	1	1	9	0
f_2^1	3	1	1	1	9	0
f^*	3	1	1	1	9	0

Table 2. Comparing results from [1] with the proposed method for 12 × 10 matrix from Fig. 1

Method	Cells	Cell 1 Utilization	Cell 2 Utilization	Cell 3 Utilization	Cell 4 Utilization	Exceptional Elements	Voids
[1]	3	0.8125	0.8667	0.8889	-	5	6
Brute Force	3	1	1	0.7083	-	7	7
Formal Concepts	4	1	1	1	0	12	0
Dense Rectangles	3	1	1	0.7083	-	7	7

In the first example, the measures produce practically the same solution which can be seen as satisfactory solutions. In the second case, $f_2^{0.2}$ and f^* produce similar solutions which have lower machine utilization than the solution obtained using f_2^1 (which has full machine utilization). On the contrary, f_2^1 has a large number of small cells and larger amount of exceptional elements. Which of the solutions is actually the best one depends on particular application and preferences of users. This example demonstrates that by tuning parameters of quality functions, one can influence the solutions based on user-specified requests (i.e., larger utilization, smaller number of cells, etc.).

Comparison With Other Approaches. The aim of this section is to compare the quality results obtained by other authors to the results obtained by the method proposed in this paper. For this purpose, we use some of the datasets which we identified in the literature. We present the comparisons by means of uniform tables which contain the characteristics of solutions available in the literature.

The data sets used for comparisons are benchmark data problems and are obtained from [1,6,14]. We use the comparison criteria for which the results are available in those papers. The papers use different approaches to solve the cell formation problem. [1] proposes a two-phase approach. The first phase makes use of principal component analysis to identify machine cells; the second phase uses an integer programming model to assign parts to these identified machine cells. [6] uses a particular iterative clustering algorithm to find cells. [14] utilizes a similarity matrix assessing similarity between machines and uses this matrix in an assignment procedure which solves a particular maximization problem to form cells.

Table 3. Comparing results from [14] with the proposed method for 14 × 24 matrix from Fig. 2 (bottom)

Method	Cells	Cell 1	Cell 2	Cell 3	Cell 4	Cell 5	Cell 6	Cell 7	Exceptional	Voids
		Utilization	Utilization	Utilization	Utilization	Utilization	Utilization	Utilization	Elements	
[14]	4	1	0.5	0.625	0.639	-	-	-	4	29
$f_2^{0.2}$	4	0.6875	0.6	0.8333	0.6667	-	-	-	2	28
f_2^1	7	1	1	1	1	1	1	1	20	0
f^*	4	0.9333	0.6389	0.6875	0.6667	-	-	-	4	23

Table 4. Comparing results from [6] with the proposed method for 15 × 10 matrix from Fig. 2

Method	Cells	Cell 1	Cell 2	Cell 3	Cell 4	Exceptional	Voids
		Utilization	Utilization	Utilization	Utilization	Elements	
[6]	3	0.8125	1	0.9333	-	0	4
$f_2^{0.01}$	3	1	0.9333	0.85	-	0	4
f_2^1	4	1	0.9333	1	1	6	1
f_2^1	3	1	1	0.85	-	2	3

Table 1, Table 2, Table 3, and Table 4 contain the comparisons. For every table, we provide a reference to the paper from which we got the dataset and the characteristics of the solutions obtained by the authors in the respective paper. For every solution listed, we provide the number of cells in the solution, machine utilization for every cell, the number of exceptional elements (black entries in the data matrix which are not covered by any cell), and the number of voids (white entries in the cells).

5 Conclusions

We presented a new method for the cell-formation problem known from the group technology. The method is inspired by formal concept analysis. We provided results of experiments and a basic comparison with other methods presented in the literature. One advantage of our method is that it is transparent in that it does not use any preprocessing method (such as the principal component analysis) which some of the methods in the literature use. Another advantage is the fact that our method is parameterizable and yields solutions based on user's preference regarding the importance of machine utilization and exceptional elements, as well as the overall number of cells.

Future research will include more comprehensive comparison with existing approaches; exploring the possibility to add interactivity to the method via a visual inspection of the concept lattice associated to the input data (the user might give some initial information to the algorithm based on such inspection); and extending our method to be able to take into account in a natural way the user's expert knowledge and preferences. In addition, we plan to explore other approaches to dense rectangles which appeared in the literature (suggested by an anonymous reviewer).

Acknowledgment

We thank the anonymous reviewers for their helpful comments.

References

1. Albadawi, Z., Bashir, H.A., Chen, M.: A mathematical approach for the formation of manufacturing cells. Computers & Industrial Engineering 48, 3–21 (2005)
2. Askin, R.G., Selim, H.M., Vakharia, A.J.: A methodology for designing flexible cellular manufacturing systems. IIE Transactions 29(7), 599–610 (1997)
3. Belarmino, A.-D., Lozano, S., Racero, J., Guerrero, F.: Machine cell formation in generalized group technology. Computers & Industrial Engineering 41(2), 227–240 (2001)
4. Belohlavek, R., Vychodil, V.: Dense rectangles in object-attribute data. In: Proc. IEEE GrC 2006, IEEE International Conference on Granular Computing, Atlanta, GA, May 10–12, 2006, pp. 586–591 (2006) IEEE Catalog Number 06EX1286, ISBN 1-4244-0133-X
5. Carpineto, C., Romano, G.: Concept Data Analysis. Theory and Applications. Wiley, Chichester (2004)
6. Chan, H., Milner, D.: Direct clustering algorithm for group formation in cellular manufacturing. J. Manufacturing Systems 1(1), 65–67 (1982)
7. Ganter, B., Wille, R.: Formal Concept Analysis. Mathematical Foundations. Springer, Heidelberg (1999)
8. Heragu, S.S.: Group technology and cellular manufacturing. IEEE Transactions on Systems, Man, and Cybernetics 24(2), 203–215 (1994)
9. Khan, M.S.I., Bhaba, S.R.: A similarity coefficient measure and machine-parts grouping in cellular manufacturing systems. International Journal of Production Research 38(3), 699–720 (2000)
10. Miltenburg, J., Zhang, W.: A comparative evaluation of nine well-known algorithms for solving the cell formation problem in group technology. Journal of Operations Management 10(1), 44–72 (1991)
11. Pearson, K.: On Lines and Planes of Closest Fit to Systems of Points in Space. Philosophical Magazine 2(6), 559–572 (1901)
12. Pillai, M.V., Subbarao, K.: A robust cellular manufacturing system design for dynamic part population using a genetic algorithm. International Journal of Production Research 46(18), 5191–5210 (2008)
13. Selim, H., Askin, R.G., Vakharia, A.: Cell Formation in Group Technology: Review, Evaluation, and Direction for Future Research. Computers & Industrial Engineering 34(1), 3–20 (1998)

14. Srinivasan, G., Narendran, T., Mahadevan, B.: An assignment model for the part families problem in group technology. Int. J. Production Research 28(1), 145–152 (1990)
15. Suresh, N.C., Meredith, J.R.: Achieving factory automation through group technology principles. Journal of Operations Management 5(2), 151–167 (1985)
16. Vakharia, A.J.: Method of cell formation in group technology: a framework for evaluation. Journal of Operations Management 6(3-4), 257–271 (1986)
17. Wemmerlöv, U., Hyer, N.L.: Cellular manufacturing in the U.S. industry: A survey of users. Int. Journal of Production Research 27(9), 1511–1530 (1989)
18. Yasuda, K., Yin, Y.: A dissimilarity measure for solving the cell formation problem in cellular manufacturing. Computers & Industrial Engineering 39, 1–17 (2001)

Identifying Ecological Traits: A Concrete FCA-Based Approach

Aurélie Bertaux[1,2], Florence Le Ber[1,3], Agnès Braud[2],
and Michèle Trémolières[1]

[1] LHyGeS UMR 7517,
ENGEES, 1 quai Koch BP 61039 F 67070 Strasbourg cedex
{aurelie.bertaux,florence.leber}@engees.u-strasbg.fr,
michele.tremolieres@bota-ulp.u-strasbg.fr
http://engees-web.u-strasbg.fr/site/
[2] LSIIT UMR 7005,
Bd Sébastien Brant BP 10413 F 67412 Illkirch cedex
agnes.braud@unistra.fr
https://lsiit.u-strasbg.fr/fdbt-fr/index.php/Accueil
[3] LORIA UMR 7503,
BP 35 F 54506 Vandœuvre-ls-Nancy cedex

Abstract. This paper describes a method to identify so-called *ecological traits* of species based on the analysis of their biological characteristics. This biological dataset has a complex structure that can be formalized as a *fuzzy many-valued context* and transformed into a binary context through *histogram scaling*. The core of the method relied on the construction and interpretation of formal concepts and was used on a 50 species × 124 histogram attributes table. The concepts were analyzed with the help of an hydrobiologist, leading to a set of ecological traits which were inserted in the original context for validation.

Keywords: Galois lattice, fuzzy many-valued context, histogram scaling, hydrobiological data.

1 Introduction

Water quality is an important problem in Europe that has been highlighted by the recent European Water Framework Directive [1]. An important issue is the evaluation of the quality of the whole ecosystem *wrt* pressures it endures (chemical pollution, buildings, lack of water...). In France, for example, running waters are qualified with physico-chemical and biological tools. Contrarily to physico-chemical tools, biological tools give global informations on the ecosystem and keep signs of fugitive pressures such as punctual pollutions. However their results are difficult to compare because they are based on compartmented and regionalized expertises. A particular problem is that biological tools are based on species (plants, fishes ...) living in different areas, which prevents from a large comparison of the existing tools. Thus, biological as well as physico-chemical

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 224–236, 2009.

tools seem not to be sufficient and new tools are needed to evaluate the whole ecological system [1].

The work presented in this article is part of a wider project that aims at comparing biological answers concerning various pressures endured by water bodies [2]. We are exploring a promising approach that is to build sets of ecological traits which characterize the functioning of the species within their environment and can be used instead of the species themselves [3]. Several ecological traits have been described [4] and the main problem is to select the suitable traits for characterizing water quality. The goal of this paper is to present a full approach - based on FCA - to identify those traits from the analysis of biological data (namely *biological traits*) about species [5].

We explored these data with Formal Concept Analysis [6], [7], [8]. We have chosen to use Galois lattices because they are useful tools for extracting knowledge [9] and allow to interact with an expert. The goal is to find concepts, i.e. sets of biological traits shared by a group of species, which can be interpreted according to the ecosystem, and so lead to ecological traits. The whole method to identify *ecological traits* from species and their biological traits is illustrated on Fig. 1.

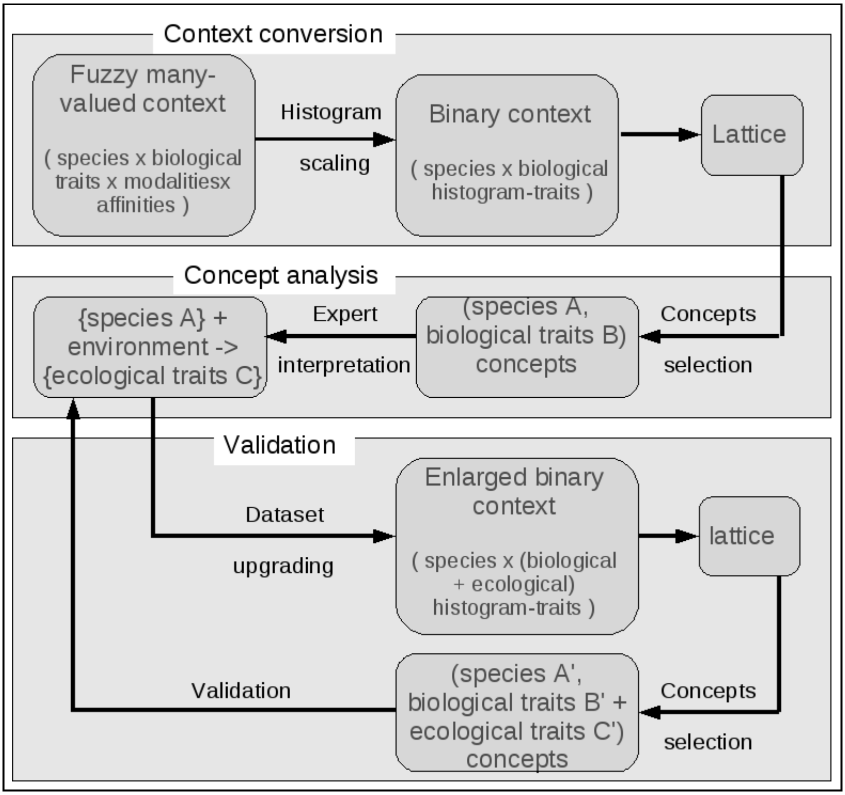

Context conversion
Fuzzy many-valued context
(species x biological traits x modalitiesx affinities)
Histogram
scaling
Binary context
(species x biological histogram-traits)
Lattice
Concept analysis
{species A} + environment -> {ecological traits C}
Expert
interpretation
(species A, biological traits B) concepts
Concepts
selection
Validation
Dataset
upgrading
Enlarged binary context
(species x (biological + ecological) histogram-traits)
lattice
Validation
(species A', biological traits B' + ecological traits C') concepts
Concepts
selection

Fig. 1. Identification method for ecological traits

The method is divided in 3 steps: context conversion, concept analysis and validation. For the purpose of context conversion, we will introduce the definitions of *fuzzy many-valued context*, which represents a formal setting for hydrobiological data, and that of *histogram scaling* which is used to transform these data to get a binary context. From this binary context a lattice is built. In the second step, concepts potentially interesting for an expert interpretation are selected. The expert interprets them by associating their species with ecological traits according to the knowledge of the species environments. Then starts the third step that aims at validating the ecological traits highlighted. They are thus added to the initial context and a new lattice is built. Concepts are selected from the enlarged lattice to find those with an extent matching the extent of the concepts selected at the previous step. The selected ecological traits are validated by comparing the intents of the concepts of the lattice based on biological traits with the intents of the lattice based on biological and ecological traits.

This article describes the whole process for the identification of ecological traits and is organized according to the successive steps of the method. The first part presents the dataset of biological traits and its conversion into a binary context. The second part is about concept analysis with a selection of the appropriate concepts and their interpretation. The third part presents the validation step of the method and the last part gives some conclusions.

2 Framework and Context Conversion

This part presents the first step of the method. First, we introduce the hydrobiological dataset. Then we recall some definitions to introduce two new definitions: *fuzzy many-valued context*, and *histogram scaling*. Finally we present the lattice obtained.

2.1 Dataset

The dataset concerns macrophytes (or hydrophytes) i.e. macroscopic plant species living in water bodies. These data have been collected from the literature [5], [4] and correspond to species living in the Alsace plain. There are 50 species and each of them has 10 characteristics called *traits*. For example, Tab. 1 shows the *vegetative reproduction* trait for a 25 species subset of the dataset.

Each trait is divided into several *modalities*: the modality *bulb or tubercle* is one of the four modalities of the vegetative reproduction trait. There are 35 modalities for the 10 traits. For each m modality of a trait and each s specie, there is an *affinity* corresponding to the percentage of plants of s potentially having the m value. For example, Tab. 1 shows that individuals of GROD (*Groenlandia densa*) have a vegetative reproduction mainly through a bulb, a tubercle, a rhizome or a stolon, but none have a bulbil, a turion or a dormant apex and some have non specialized fragments.

Table 1. Vegetative reproduction trait of half the dataset

Traits	Vegetative reproduction			
Modalities	bulb or tubercle	Rhizome or stolon	bulbil, turion or dormant apex	non specialized fragments
ALIP	0	100	0	0
BERE	0	66	0	33
CALO	0	0	0	100
CALP	0	0	0	100
CARA	0	100	0	0
CERD	0	0	50	50
ELEA	0	100	0	0
ELOC	0	0	50	50
ELON	0	0	33	66
GROD	40	40	0	20
HIPV	0	66	0	33
HOTP	0	50	0	50
HYDM	0	50	50	0
IRIP	25	25	0	50
JUNA	0	100	0	0
LEMM	0	0	100	0
LEMT	0	0	100	0
MENA	0	100	0	0
PHAA	0	100	0	0
PTCR	0	40	40	20
PTLU	0	66	0	33
PTNA	0	50	25	25
SEFC	0	100	0	0
TYPL	0	100	0	0
UTRV	0	0	50	50

2.2 Fuzzy Many-Valued Contexts

The structure of the previous dataset can be considered as a fuzzy context or as a many-valued context. We introduce thus the notion of *fuzzy many-valued context*, which is defined in the following. Let us first recall the definition of a formal context [8].

Definition 1. *A formal context* $K := (O, T, I)$ [1] *is composed of two sets* O *and* T *and a relation* I *between* O *and* T*. The elements of* O *are called the objects and the elements of* T *are called the traits (or attributes). In order to express that an object* o *is in relation* I *with a trait* t*, we write* oIt *or* $(o, t) \in I$ *and read it as "the* $o \in O$ *object has the* $t \in T$ *trait".*

This notion has been extended in [10] to deal with fuzzy data. Fuzzy data allow to represent a fuzzy relation between an object and a trait, or said in a

[1] We use OTI instead of GMI, $OTMI$ instead of $GMWI$ to recall the Traits and Modalities sets used.

different way, the affinity of an object towards a trait. The underlying context is then called a *fuzzy context.*

Definition 2. *Let A be a set of truth degrees. Then a fuzzy context is a triplet $K := (O, T, I)$ where O and T are sets of objects and traits respectively, and $I : O \times T \rightarrow A$ is a fuzzy relation between O and T. A degree $I(o, t) \in A$ is interpreted as the degree to which the o object has the t trait.*

The definition 1 of a formal context is a specific case of the definition 2 where $I : O \times T \rightarrow \{0, 1\}$. The notion of formal context has also been extended in another way in [8] to deal with multi-valued data. Such contexts are called *many-valued contexts.*

Definition 3. *A many-valued context K is defined as a quadruple (O, T, M, I), where O is a set of objects, T is a set of many-valued attributes called traits, M is a set of trait values called modalities, and I is a ternary relation, $I \subseteq O \times T \times M$ such that: $(o, t, m) \in I$ and $(o, t, n) \in I$ always implies $m = n$. The notation $(o, t, m) \in I$ (or $t(o) = m$) means that the t attribute has the m value for the o object.*

Nevertheless some data exhibit both aspects. This is the case for our dataset so that we have defined *fuzzy many-valued contexts*, which extend many-valued contexts to the case where $(o, t, m) \in I$ and $(o, t, n) \in I$ do not imply $m = n$: the m and n modalities of the t trait belong to the o object with different degrees. In our dataset, for example, the 'non specialized fragments' and 'rhizome or stolon' modalities of the 'vegetative reproduction' traits both belong to the GROD object with (respectively) a 20% and a 40% affinity.

Definition 4. *A fuzzy many-valued context K is a quintuplet (O, T, M, A, I) where O, T, M and I are defined as for the many-valued context, A is a set of affinities and corresponds to a set of truth degrees as for a fuzzy context. The notation $I(o, t, m) \in A$ or $I(o, t, m) = a$ means the o object has the a affinity for the m modality of the t trait.*

2.3 Histogram Scaling

Conceptual scales [11], [12] have been defined in order to deal with increasing amounts of data and many-valued contexts. They consist in grouping related attributes. [13] defines a conceptual scale for a t many-valued attribute as a one-valued context which has the attribute values of t among its objects. A scale may be associated to each t many-valued attribute, and t is replaced by the set of its scale attributes. Each value of t is substituted by the corresponding row of the scale.

The aim of our work is to build a Galois lattice from the hydrobiological dataset in order to extract valuable concepts describing groups of species with common properties. For that purpose we need to put the data in a suitable format corresponding to the definition of a binary context. In a previous work [14], the

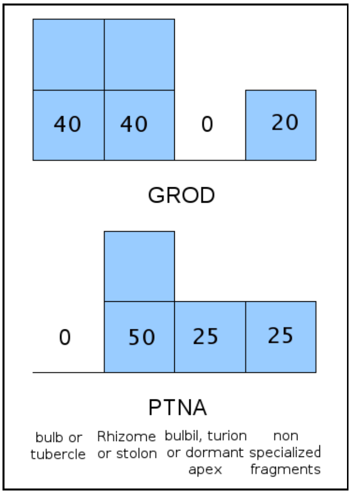


Fig. 2. Examples of histograms for the Vegetative reproduction trait

nominal scaling was tested, but the results were not satisfying for the biologists, because the modalities of a same trait were scattered into several attributes. We also tested another approach which took into account the biological meaning of the data, and which turned out to be relevant. The various modalities of a trait were therefore merged into a single multi-valued attribute and a (non-fuzzy) many-valued context was built, to which a nominal scaling was applied. The whole conversion we named *histogram scaling.*

Definition 5. *Let $K := (O, T, M, A, I)$ be a fuzzy many-valued context. For each $t \in T$ trait having m_t modalities, we associate a H_t* [2] *set of histogram-traits (called histograms). An $h_t \in H_t$ histogram is such that $h_t = \{a_1, \ldots, a_i, \ldots, a_{m_t}\}$ where $a_i \in A$. Let $H = \cup_{t \in T} H_t$ be the set of all histograms. The K fuzzy many-valued context can be represented as a (O, H, I_H) binary context. For an $o \in O$ object and an $h_t \in H$ histogram, $(o, h_t) \in I_H \Leftrightarrow I(o, t, m_i) = a_i$ for all $i \in [1, m_t]$.*

An $h_t \in H$ histogram is composed of a letter to qualify the considered t trait (by example V for Vegetative reproduction), and of the m_t affinities of the considered object for the m_t modalities of t. Fig. 2 illustrates Vegetative reproduction modalities for GROD and PTNA (*Potamogeton natans*) species. For the bulb or tubercle modality, GROD has 40% affinity and PTNA 0%, for the Rhizome or stolon modality, GROD has 40% affinity and PTNA 50%, for the bulbil, turion or dormant apex modality, GROD has 0% affinity and PTNA 25% and for the non specialized fragments modality, GROD has 20% affinity and PTNA 25%. So their *Vegetative reproduction histogram* are V40-40-0-20 for GROD and V0-50-25-25 for PTNA.

[2] $|H_t| \leq |A|^{m_t}$.

2.4 The Galois Lattice of Histograms

The context resulting from the histogram scaling relies on an I_H binary relation, which associates a unique $h_t \in H_t$ to each specie since the histograms mutually exclude each other. So each specie has 10 histograms, i.e. one per trait. Tab. 2 shows the *Vegetative reproduction* trait of the new context obtained by histogram scaling for a 25 species subset. The whole converted dataset is composed of 124 histograms. On average, this means that an histogram is shared by 4 species. A × indicates that a specie fits the characteristics represented by the corresponding histogram.

Fig.3 shows the lattice obtained with the ordinary Galois connexion [8] from the (O, H, I_H) context of histograms (25 macrophytes x 10 traits).

Table 2. Vegetative reproduction histogram of the converted dataset for 25 species

	V0-100-0-0	*V0-66-0-33*	*V0-0-0-100*	*V0-0-50-50*	*V0-0-33-66*	*V40-40-0-20*	*V0-50-0-50*	*V0-50-50-0*	*V25-25-0-50*	*V0-0-100-0*	*V0-40-40-20*	*V0-50-25-25*
ALIP	×											
BERE		×										
CALO			×									
CALP			×									
CARA	×											
CERD				×								
ELEA	×											
ELOC				×								
ELON					×							
GROD						×						
HIPV		×										
HOTP							×					
HYDM								×				
IRIP									×			
JUNA	×											
LEMM										×		
LEMT										×		
MENA	×											
PHAA	×											
PTCR											×	
PTLU		×										
PTNA												×
SEFC	×											
TYPL	×											
UTRV				×								

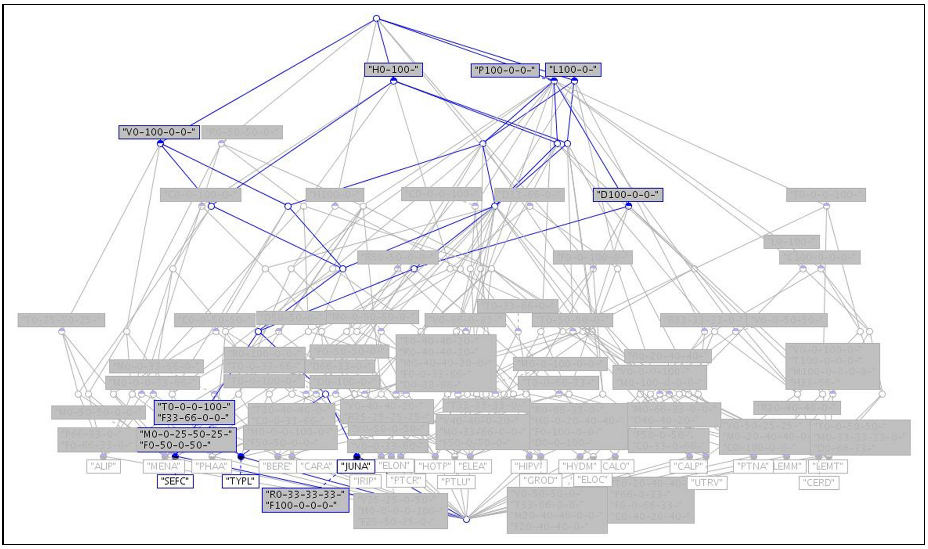


Fig. 3. Galois lattice for the 25 macrophytes subset

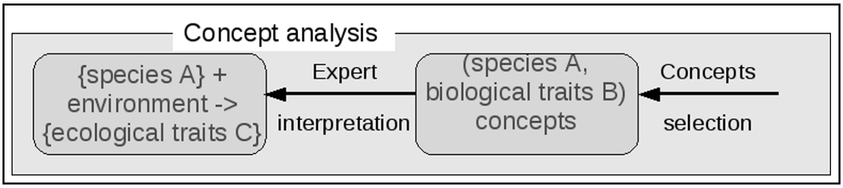

Concept analysis
{species A} + environment -> {ecological traits C}
Expert
interpretation
(species A, biological traits B) concepts
Concepts
selection

Fig. 4. Second part of the method : analysis of the formal concepts

3 Analysis of the Formal Concepts

Our aim is to interpret concepts with respect to expert knowledge on species environment in order to highlight ecological traits which link the species and the characteristics of the environment where they live. Such information exist in the literature but hydrobiologists look for the most appropriated traits to evaluate water quality. Thus they need to know the relations between species and their biological traits to associate them to ecological traits. In this section we present the method to determine these traits as shown on Fig. 4: the choice of the interesting concepts, their analysis and the conclusions obtained in terms of ecological traits.

3.1 Concept Selection

The Galois lattice gives all possible concepts for sets of species and their common biological descriptions. The concepts in the middle of the lattice are more meaningful for biologists, i.e. those containing between 3 and 5 histograms, which usually corresponds to a number of species between 3 and 7. Concepts with more species are too general and cannot be linked to specific environmental conditions whereas concepts with less species are too specific and give no information. Let us describe two of the 'middle' concepts:

- □ Concept 1: (JUNA SEFC TYPL, L100-0 H0-100 P100-0-0 V0-100-0-0 D100-0-0), illustrated in Fig. 3. The species considered are *Juncus articulatus, Sparganium emersum, Typha latifolia.* The traits they share are an annual flowering (L100-0), a phenology during the vegetative period only (H0-100), perennial organs (aerial or underground) (P100-0-0), a vegetative reproduction by rhizomes or by stolons (V0-100-0-0) and a high dispersion (with small flying seeds) (D100-0-0).
- □ Concept 2: (CERD PHAA PTCR PTLU PTNA UTRV, H0-100 F0-0-0-100 P100-0-0). This concept concerns *Ceratophyllum demersum, Phalaris arundinacea, Potamogeton crispus, Potamogeton lucens, Potamogeton natans, Utricularia vulgaris* macrophytes. Their common properties are a phenology during the vegetative period only (H0-100), an high flexibility (> 300°) (F0-0-0-100) and perennial organs (aerial or underground) (P100-0-0).

3.2 Expert Interpretation: From Biological to Ecological Traits

On the one hand, the Galois lattice gives us concepts, i.e. relations between biological traits and species. On the other hand, the hydrobiologist gives us knowledge on the relation between species and ecological traits which are information on the environment where they live. Based on the species sets extracted from the lattice, the expert analysis determines the common characteristics of the ecosystems where the species live. For example :

- □ Concept 1: these species live in mesotrophic to eutrophic water.
- □ Concept 2: these species live in calm water surfaces.

These conclusions reveal the need for ecological traits describing the tolerance of species to trophic status and flow.

Actually, from the common characteristics found in the concepts, five ecological traits and their modalities are determined that are potentially useful for water quality evaluation:

- □ water level stability: stable, fluctuations, occasionally
- □ resistance to flow: no tolerance, weak, medium, strong
- □ tolerance to organic matter: <10%, 10-40%, >40%
- □ tolerance to sedimentation (deposition or accumulation of mineral or organic matter): suffocated plant, medium root, strong root
- □ trophic status (pertaining to nutrition): oligotrophic, mesotrophic, eutrophic, hypertrophic.

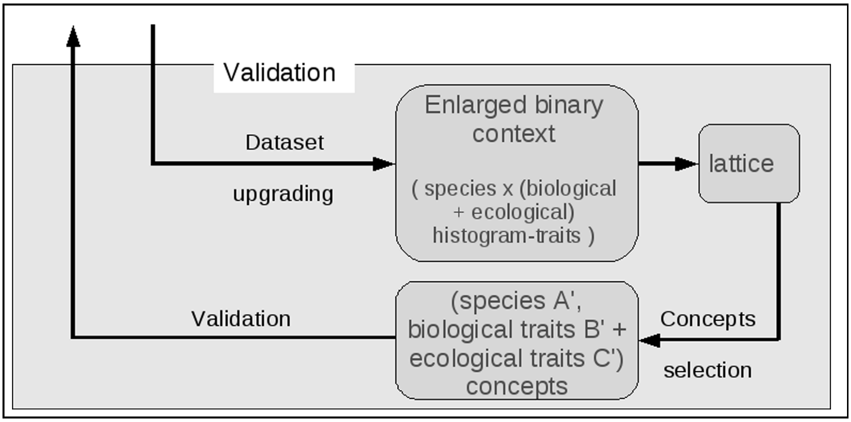

Validation
Dataset
upgrading
Enlarged binary context
(species x (biological + ecological) histogram-traits)
lattice
Validation
(species A', biological traits B' + ecological traits C') concepts
Concepts
selection

Fig. 5. Last part of the method : validation

4 Validation

In the biological literature, many ecological traits exist. The approach provides a selection of the most suitable ecological traits (and modalities) to describe alsacian species. The steps of the method already described lead us to five ecological traits. We present a validation approach of these choices with the three last steps of the method shown in Fig. 5.

4.1 Dataset Upgrading

Actually every specie is concerned by the ecological traits (with its own affinity), not only the ones belonging to the concept(s) which pointed out the ecological trait. So the ecological traits identified and their modalities are added to the dataset, based on [4]. Until now, 46 species are fully completed.

The lattice built from the enlarged context (biological and ecological traits) is shown on Fig. 6 and corresponds to the 25 species subset. The highlighted concept corresponds to concept 2 presented in section 3.1.

4.2 Validation of the Method

We have two pieces of information. The first comes from the lattice and concerns the relations between species and biological traits. The second comes from the expert and concerns the relations between the species and ecological traits. According to these informations, we should obtain relations between the biological and ecological traits of species, thus eliminating the species. For the two examples:

□ Concept 1: we should put into relations annual flowering, phenology during the vegetative period only, perennial organs, vegetative reproduction by rhizomes or by stolons and high dispersion (biological traits) with mesotrophic to eutrophic water (ecological traits)

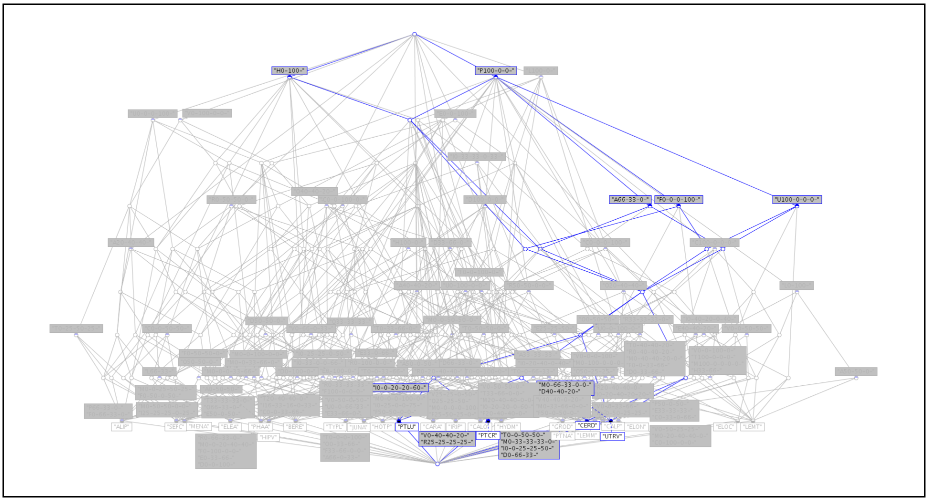


Fig. 6. Galois lattice for the 25 macrophytes subset, 10 biological and 6 ecological traits

- □ Concept 2: we should put into relations phenology during the vegetative period only, high flexibility and perennial organs (biological traits) with calm water surfaces (ecological traits)

In order to validate our process, we need to verify this reasoning. So we analyze the concepts from the lattice of biological and ecological traits to check the relations between them. Besides, we can verify the expert knowledge about species and their ecological traits.

4.3 Application

We will illustrate this step with the two concepts:

- □ Concept 1: The intent of the corresponding concept in the enlarged lattice includes new histogram attributes that confirm expertise exactly: I0-50-50-0 and E0-0-100.
  - I0-50-50-0 attribute means that the individuals of each of these species are fairly spread between mesotrophic and eutrophic waters.
  - The lattice indicates that these species share the E0-0-100 histogram too, which means they have a constant implanting. This information is revealed by the lattice.
- □ Concept 2: The 6 species of the concept do not share the same ecological traits. Only 4 species share the attributes A66-33-0 and U100-0-0-0, as illustrated in Fig. 6.
  - The A66-33-0 attribute means that each of these species have 66% of their plants living in stable water, 33% live in water having fluctuations.

This trait fits the expertise, which states that species live in calm water, for the 4 following species : CERD, PTCR, PTLU and UTRV. PTNA has the A40-40-20 attribute, indicating it lives mainly in calm water too for 80% of its plants, which is close to the expertise. On the contrary, PHAA has the A0-50-50 attribute which disagrees with the expert.

- The U100-0-0-0 attribute indicates the 4 species (CERD, PTCR, PTLU and UTRV) dislike variations of water level. PTNA still quite agrees (U0-100-0-0) : it bears small variations contrarily to PHAA which endures easily important variations (U0-0-0-100).

These examples illustrate that the lattice not only confirms expert indications (biological and ecological traits are associated such as indicated by the expert) but brings some accuracy such as for the PHAA specie. It also reveals more relations between species and ecological traits. Actually the expert gave general ecological trait for groups of species. When the data were collected from [4], specific values were given for all traits and all species, which were pointed out by the lattice. Finally, the results proved to be valuable and we obtained sets of ecological and biological traits that can be further analyzed.

5 Conclusion and Future Work

This article presents an approach to bypass the problem of biological tools used to qualify water quality, which depend on species which do not live in every place where water quality is measured. The purpose is to identify sets of ecological and biological traits to be used instead of the species.

The method consisted in analyzing first biological traits of species. The dataset was converted by histogram scaling for building a Galois lattice. This tool was chosen because it provided us with concepts of species and their common traits. These concepts were analyzed by an expert to reveal relations between species of the selected concepts and ecological traits (from their environment). Then these ecological traits were added to the database and filled according to biological literature and expert knowledge. The concepts from the enlarged lattice were examined to validate the process and to reveal sets of biological and ecological traits. Actually we found again the information given by the expert. Furthermore, the lattice gave more accurate and larger information because it was based on several expertises.

Finally this method appeared to be reliable for the hydrobiologists. The future works are to extract association rules between biological and ecological traits, and to apply this method to other biological data used to qualify water quality, such as the invertebrate compartment. This will be a more complex operation because the invertebrate species are considered with several taxonomical degrees, contrarily to the macrophytes. Furthermore, we intend to develop specific Galois connections, such as proposed in [15], to deal with the histogram format and thus better handle fuzzy many-valued contexts within the FCA framework.

Acknowledgments

A. Bertaux, F. Le Ber, A. Braud and M. Trémolières thank the Agence de l'Eau Rhin Meuse and the Région Alsace for supporting this work.

References

1. Bazerques, M.-F.: Directive-cadre sur l'eau: le bon état écologique des eaux douces de surface, sa définition, son évaluation. Communication au Ministère de l'écologie et du Développement Durable, Paris (2004)
2. Grac, C., Le Ber, F., Braud, A., Handja, A., Hermann, A., Lachiche, N., Trémolières, M.: Mining a database on Alsatian rivers. In: Proceedings of the seventh International Conference on Hydroinformatics HIC (2006)
3. Lafont, M.: A conceptual approach to the biomonitoring of freshwater: the Ecological Ambience System. Journal of Limnology 60(suppl. 1), 17–24 (2001)
4. Willby, N.J., Abernethy, V.J., Demars, B.O.L.: Attribute-based classification of European hydrophytes and its relationship to habitat utilisation. Freshwater Biology 43(1), 43–74 (2000)
5. Staerck, J.-F.: Analyse des traits biologiques de macrophytes aquatiques en relation avec des perturbations types. Mémoire de licence professionnelle ULP - ENGEES - CEVH (2005)
6. Barbut, M., Monjardet, B.: Ordre et classification - Algèbre et combinatoire. Hachette, Paris, France (1970)
7. Davey, B., Priestley, H.: Introduction to Lattices and Order. Cambridge University Press, Cambridge (1990)
8. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical foundations. Springer, Heidelberg (1999)
9. Napoli, A.: A smooth introduction to symbolic methods in knowledge discovery. In: Cohen, H., Lefebvre, C. (eds.) Categorization in Cognitive Science. Elsevier, Amsterdam (2006)
10. Belohlávek, R., Vychodil, V.: What is a fuzzy concept lattice? In: 3rd Int. Conference on Concept Lattices and Their Applications, pp. 34–45 (2005)
11. Ganter, B., Kuznetsov, S.: Pattern Structures and Their Projections. In: Proceedings of the 9th International Conference on Conceptual Structures, pp. 129–142 (2001)
12. Stumme, G.: Hierarchies of Conceptual Scales. In: Proceedings of Workshop on Knowledge Acquisition, Modeling and Management (KAW 1999), Banff, pp. 78–95 (1999)
13. Ganter, B., Wille, R.: Applied Lattice Theory: Formal Concept Analysis. In: Grätzer, G. (ed.) General Lattice Theory. Birkhäuser, Basel (1997)
14. Bertaux, A., Le Ber, F., Braud, A., Trémolières, M.: Mining Complex Hydrobiological Data with Galois Lattices. International Journal of Computing and Information Sciences (to appear)
15. Polaillon, G.: Organisation et interprétation par les treillis de galois de données de type multivalué, intervalle ou histogramme. Thèse de doctorat, Université Paris IX Dauphine (1998)

A Concept Lattice-Based Kernel for SVM Text Classification

Claudio Carpineto, Carla Michini, and Raffaele Nicolussi

Fondazione Ugo Bordoni, Rome, Italy
{carpinet,cmichini,rnicolussi}@fub.it

Abstract. Standard Support Vector Machines (SVM) text classification relies on bag-of-words kernel to express the similarity between documents. We show that a document lattice can be used to define a valid kernel function that takes into account the relations between different terms. Such a kernel is based on the notion of conceptual proximity between pairs of terms, as encoded in the document lattice. We describe a method to perform SVM text classification with concept lattice-based kernel, which consists of text pre-processing, feature selection, lattice construction, computation of pairwise term similarity and kernel matrix, and SVM classification in the transformed feature space. We tested the accuracy of the proposed method on the 20NewsGroup database: the results show an improvement over the standard SVM when very little training data are available.

1 Introduction

Kernel-based learning methods are being actively investigated because they permit to decouple the problem of choosing a suitable feature space from the design of an effective learning algorithm. The idea is to use a linear algorithm to solve a non-linear problem by mapping the original features into a higher-dimensional space where the linear algorithm is subsequently used. A key enabling factor is that kernel methods exploit inner products between all pairs of data items and that such products can be often computed without explicitly representing the transformed, high-dimensional feature space.

Support Vector Machines (SVM) is probably the best known learning algorithm based on kernel, with text classification being one of its most natural applications because retrieval techniques are based just on the inner-products between vectors. While SVMs with bag-of-words kernel have shown to perform well ([10], [12]), they are limited by their inability to consider relations between different terms. If, due to the vocabulary problem, two documents refer to the same issue using different terms, such documents would be mapped to distant regions of the transformed feature space.

The question is whether it is possible to define a semantically-enriched document similarity measure and to embed it in a kernel-defined feature space. This issue has been addressed in a few earlier works, mainly using latent semantic indexing [6] and WordNet [16] as knowledge sources. Our work is in the same

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 237–250, 2009.

research line, with the difference that we take an approach based on formal concept analysis.

We use a document lattice (i.e., the concept lattice associated with the training documents) to discover relations between the terms in the documents. The relation between two terms is determined using the shortest path (topological distance) between the corresponding attribute concepts in the document lattice; the closer the two attributes are to each other, the greater their semantic relation. Such relations between terms can then be easily incorporated into a *valid* document similarity kernel function, i.e., such that it can be rewritten as an inner product $K(x, y) = \langle \phi(x), \phi(y) \rangle$ for some $\phi : X \rightarrow F$ defining a feature space.

Note that although document lattices have been used in several information retrieval tasks, this is the first attempt, to the best of our knowledge, to investigate the potentials of a concept lattice-based kernel for text. Likewise, the exploitation of structural interdocument similarities to expand the document representation is a novel approach to defining semantic kernels for text.

Following this idea, we have implemented a full SVM text classification system with concept lattice-based kernel. It consists of several steps, namely text preprocessing, feature selection, lattice construction, computation of pairwise term proximity and kernel matrix, and finally SVM classification in the transformed feature space. Each step is described in detail in the paper.

The proposed method has been evaluated by comparing its accuracy to that of SVM with standard kernel. We used the 20 NewsGroup dataset and simulated critical learning conditions with little training data. We found that in this situation the concept lattice-based kernel was more effective.

The remaining of the paper has the following organization. We first introduce SVM and kernels for text, showing the general formulation of a document similarity function used as a kernel. Then we present our specific kernel function based on pairwise term proximities extracted from a document lattice, and we describe the full SVM text classification system in which it has been been embodied. The next sections are about experimental results and related work. We finally offer some conclusions and future research directions.

2 SVM and Kernels for Text

Let us define $\{(d_i, c_i), i = 1...l\}$ the training set for a binary classification problem, where each d_i is a document described by a set of terms and c_i is either 1 or -1, indicating the class to which the document d_i belongs. Support Vector Machines construct the separating hyperplane that maximizes the *margin* between the two sets of document vectors, i.e., such that it hat has the largest distance to the neighboring documents of both classes. From the definition of Support Vector Machines and the kernel theory ([17], [15]), the decision function is defined as:

$$f(d) = sgn(\sum_{i=1}^{l} \alpha_i \; c_i \; K(d, d_i) + b) \tag{1}$$

where d is the document to be classified, the vector α and the scalar b are the parameters of the maximum margin hyperplane, and K is a document similarity function that satisfies the Mercer's condition. The Mercer's condition states that the Gram matrix $G = K(d_i, d_j), \forall i, j = 1...l$, must be symmetric and positive definite. Under this condition, the function K is a valid kernel.

One of the most popular (Mercer) kernels is the linear kernel, which corresponds to the mapping $\phi(d) = d$. Using the bag-of-words model, the feature space is defined by the terms used to index the documents, and the linear kernel is given by the inner product between the document feature vectors, i.e.

$$K(d_1, d_2) = \langle d_1, d_2 \rangle = d_1^T d_2 \tag{2}$$

where $TD = [d_1..d_l]$ is the classic term by document matrix, whose columns are the documents and whose rows are the terms. Each element $TD(i, j)$ is equal to the number of occurrences of term i in document j, or to a term weight which best reflects the importance of each term in each document. In general, the matrix TD is sparse, as the fraction of elements for which $TD(i, j) = 0$, meaning that term i is not contained in document j, is very high. With this model, the Gram matrix is just the document by document matrix $G = TD^T TD$.

Another classical kernel is the Gaussian kernel, which is given by:

$$K(d_1, d_2) = exp\ (-\ \gamma\ ||d_1 - d_2||^2) \tag{3}$$

It can be shown that a Gaussian kernel performs a mapping into an infinite dimensional space, which can better handle the case when both classes are not linearly separable in the input space and often yields better performance than the linear kernel.

Instead of using the bag-of-words model with the original input features, it is possible to consider a linear mapping of the document vectors $\phi(d) = Pd$, where P is any appropriately shaped matrix. In this case, a valid (linear) kernel is given by:

$$K(d_1, d_2) = d_1^T\ P^T\ P\ d_2 \tag{4}$$

because the corresponding Gram matrix is symmetric and positive definite [6]. The matrix P typically encodes pairwise term similarities, thus implying that the mapping $\phi(d) = P\ d$ allows to represent each document not only by its original terms but also by the terms that are related to each of them. For example, if a document is indexed with only the first element of the term index, and the second term of the index is highly related to the first one, then the second component in the *mapped* document vector will be increased from zero to a positive value. In a sense, this amounts to performing a semantic smoothing of the original features via document expansion.

Note also [16] that the the linear transformation expressed by the mapping $\phi(d) = Pd$ can be used to redefine a gaussian kernel according to the semantic smoothing, i.e.:

$$K(d_1, d_2) = exp\ (-\ \gamma\ ||(d_1 - d_2)^T\ P^T\ P\ (d_1 - d_2)||^2) \tag{5}$$

By varying the matrix P one can obtain different transformations of the document feature space, which can be traced back to various document representation models. In the next section we describe a term similarity matrix based on the conceptual proximity between terms in a document lattice.

3 A Kernel Based on Document Lattice

Let D be the set of (training) documents, T the set of terms describing the documents, and TD the term by document matrix. Consider the ordered set $(\mathcal{C}(D,T,TD); \succ\prec)$ formed by the set of concepts of the context (D,T,TD) along with the nearest neighbour relation $(\succ\prec)$, i.e., for $x,y \in \mathcal{C}(D,T,TD)$, $x \succ\prec y$ if $x \succ y$ or $y \succ x$. Define the *concept distance* between concepts x and y as the least $n \in \mathcal{N}$ for which the following condition holds:

$$\exists z_0,\ z_1,\ \ldots,\ z_n\ \in\ \mathcal{C}(D,T,I);\ \succ\prec)\text{ such that }x = z_0 \succ\prec z_1 \ldots \succ\prec z_n\ =\ y.$$

Consider now two terms $(t_1,t_2), t_i \in T$. The *term distance* between t_1 and t_2 is given by the concept distance (as defined above) between the two corresponding term concepts (t_1', t_1'') and (t_2', t_2''), expressed by the standard prime and double prime operators.

Note that we remove the top and the bottom elements of the document lattice before computing the pairwise term proximities, because such concepts do not have a real meaning (even when the intent of the top concept is not empty it does not bear any information) and they may short-circuit conceptually distant concepts. The top element of the lattice is especially critical because term concepts are typically co-atoms.

The found relations have an intuitive meaning in terms of the properties of near concepts on the document lattice. A zero term distance means that the two terms always occur together in the documents (i.e., their mutual information is maximum). Distance equal to 1 means that there is a term a in the pair that always co-occurs with the other term b (i.e., the conditional probability of a given b is equal to 1). Distance 2 means that *either* term a always co-occurs with term b and there is some other term c that co-occurs with b more frequently than a (i.e., when a is a nephew of b), *or* (when they have a child in common) that a and b co-occur in one or more documents and there is no other term that co-occurs with either a or b in a superset of such documents. And so on.

The proximity between two terms is inversely related to their distance. In order to find the proximity matrix P, we normalize the distance values by dividing by their maximum and then we subtract the normalized values from 1.

As an illustration, consider the simple context for vertebrate animals in Table 1; each row can be seen as a document and each column as an indexing term (possibly formed by multiple words). We first show in Table 2 the bag-of-word kernel matrix, computed with equation 2. The kernel values shown in Table 2 have been normalized to take into account the different length of documents,

Table 1. A context for vertebrate animals

		breathes	can	has	has	has	lives in	vivipar-	produces
		in water	fly	beak	hands	wings	water	ous	light
		(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
1	Bat		x			x		x	
2	Eagle		x	x		x			
3	Monkey				x			x	
4	Parrot fish	x		x			x		
5	Penguin			x		x	x		
6	Shark	x					x		
7	Lantern fish	x					x		x

Table 2. Bag-of-words kernel matrix for the context in Table 1 (the matrix is symmetric)

	1	2	3	4	5	6	7
1	1	0,666667	0,408248	0	0,333333	0	0
2		1	0	0,333333	0,666667	0	0
3			1	0	0	0	0
4				1	0,666667	0,816497	0,666667
5					1	0,408248	0,333333
6						1	0,816497
7							1

by using: $\overline{K}(d_1, d_2) = \frac{d_1^T d_2}{||d_1||\,||d_2||}$. Note that when two animals do not have any property in common their kernel value is always equal to 0.

We now turn to the illustration of the concept lattice-based kernel. We show in Figure 1 the concept lattice built from the context in Table 1, and we report in Table 3 the pairwise term distances derived from the concept lattice in Figure 1 after the removal of its top and bottom element. For instance, the distance between terms (g) and (e) is equal to 3, because the shortest connecting path is: (1 3, g) $\succ\!\prec$ (1, b e g) $\succ\!\prec$ (1 2, b e) $\succ\!\prec$ (1 2 5, e).

The proximity matrix is shown in Table 4. The found conceptual proximities are meaningful. For instance, the most related terms are: (e) and (b) (i.e., 'has wings' and 'can fly'), (f) and (a) (i.e., 'lives in water' and 'breathes in water'), (g) and (d) (i.e., 'viviparous' and 'has hands'), (h) and (a) (i.e., 'produces light' and 'breathes in water'); note that the last relation is due to the fact that the only animal in the given context who produces light is a fish. The most unrelated terms are: (d) and (a) (i.e., 'has hands' and 'breathes in water'), and (h) and (g) (i.e., 'produces light' and 'viviparous').

Turning to pairwise document similarity, the concept lattice-based kernel matrix for the animal context can be computed with equation 4 using the proximity matrix in Table 4. The normalized kernel values are shown in Table 5.

Due to the implicit relationships between terms, two documents may have a varying degree of similarity even when they share the same number of terms

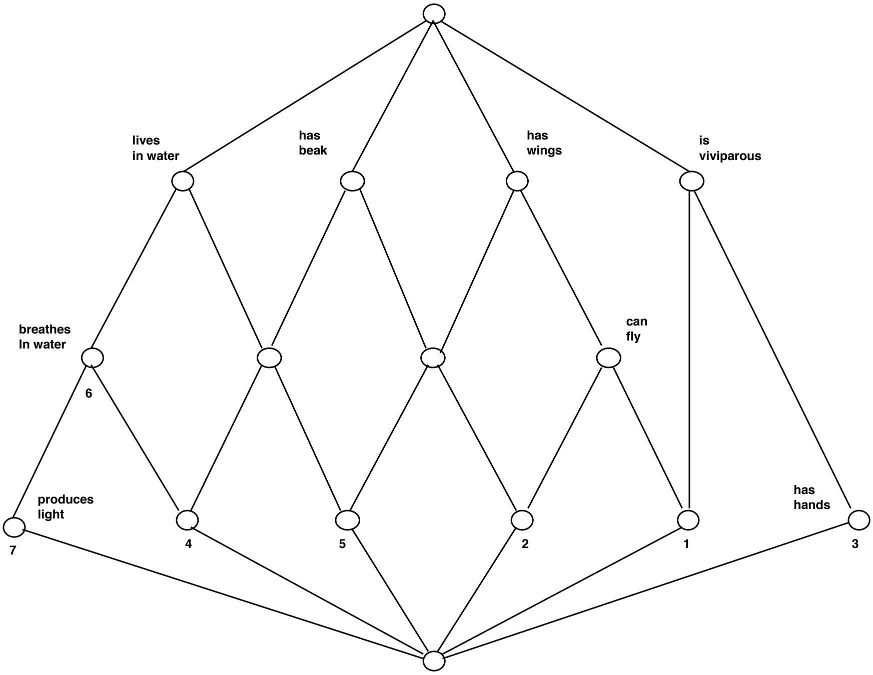


Fig. 1. Concept lattice for the context of Table 1, with minimal labelling

Table 3. Distances between pairs of the attributes in Table 1, derived from the concept lattice in Figure 1 (the matrix is symmetric)

	a	b	c	d	e	f	g	h
a	0	6	3	9	5	1	8	1
b		0	3	3	1	5	2	7
c			0	6	2	2	5	4
d				0	4	8	1	10
e					0	4	3	6
f						0	7	2
g							0	9
h								0

(e.g., according to Table 5, ‘bat’ is more similar to ‘eagle’ than to ‘penguin’, although ‘bat’ has two terms in common with both ‘eagle’ and ‘penguin’), or if they do not share any term (e.g., ‘eagle’ is more similar to ‘monkey’ than to ‘shark’).

A less obvious result is that the pairwise similarities in Table 5 may be relatively high even when the two documents are apparently very different (such as with ‘eagle’ and ‘lantern fish’) and that most values are distributed in the upper part of the range. This is probably due to the characteristics of this particular context, because each animal does not neatly fit in one class and shares

Table 4. Pairwise term proximities derived from corresponding distances in Table 3 (the matrix is symmetric)

	a	b	c	d	e	f	g	h
a	1	0.4	0.7	0.1	0.5	0.9	0.2	0.9
b		1	0.7	0.7	0.9	0.5	0.8	0.3
c			1	0.4	0.8	0.8	0.5	0.6
d				1	0.6	0.2	0.9	0
e					1	0.6	0.7	0.4
f						1	0.3	0.8
g							1	0.1
h								1

Table 5. Concept lattice-based kernel matrix for the context in Table 1 (the matrix is symmetric)

	1	2	3	4	5	6	7
1	1	0,978114	0,95642	0,809072	0,914123	0,740581	0,704969
2		1	0,878241	0,908368	0,976945	0,855411	0,827088
3			1	0,627978	0,769209	0,545051	0,502219
4				1	0,976284	0,993071	0,98505
5					1	0,944986	0,92595
6						1	0,998065
7							1

resemblances to animals in other classes. Indeed, this example was originally given to illustrate the fact that several reasonable clusters can be formed out of a set of vertebrates in addition to the standard vertebrates groups, and thus it is probably not the best choice as an example for a classification task. On the other hand, this observation suggests that more analysis is needed to check whether the high similarity values may also be attributed to the specific notion of proximity used in our approach, which may be overly unconstrained. This issue deserves more attention and is left for future work, as alternative criteria are available to measure the distance between concepts in a concept lattice (see Section 6.2).

Before concluding this section we would like to point out that a different approach to finding a concept lattice-based kernel is conceivable. The key idea is that a document lattice can be used to find a conceptual similarity exactly at the document level, thus ignoring the similarities between the single terms describing the documents. One could merge the document to be classified on the concept lattice of training documents, and then compute the distance between the found concept and each training document concept. This approach is appealing since, unlike the concept lattice-based kernel illustrated above, it allows to compute the similarity between documents directly in the transformed space, without performing the linear transformation $\phi(d) = Pd$. In addition, it would have a nice interpretation from the point of view of document similarity, because the distance in the document lattice can be thought of as the minimal number of admissible intent changes – with respect to the collection at hand – that lead from the description of one document to the description of another document (see [3]).

The main disadvantage of such a similarity measure is that it is not guaranteed that the kernel is valid. On the other hand, its properties (e.g., it is symmetric, has zero diagonal, is nonnegative, obeys the triangle inequality) do not allow us to rule out such a possibility. Note also that good classification is possible despite indefinite kernel matrices, because it has been proved [9] that SVMs still solve a data separation problem (although the solution may be only a local optimum). Experimenting with this approach is an avenue for future research.

4 Full Method Description

In this section we describe the main steps involved in our implementation of a full SVM text classification system with concept lattice-based kernel.

4.1 Text Pre-processing

The aim of text pre-processing is to transform each document into a sequence of *features* that will be used in subsequent steps. The input documents go through text segmentation, punctuation removal, conversion of upper to lower case, and stop-word removal. We use strict single-word indexing and do not not perform any stemming.

4.2 Selection of Input Features

The features extracted from each document are reduced through an explicit feature selection phase. Working with a restricted set of features improves the overall system efficiency and it may also be useful for increasing classification accuracy, because index terms with low predictive power usually add noise. The selection criterion used in our system is mutual information. Both the text pre-processing and feature selection steps were performed using the Bow (or libbow) toolkit, available at: `www.cs.cmu.edu/~mccallum/bow/`.

4.3 Construction of Document Lattice

Once each document has been represented with a set of predictive features, we partition the entire dataset in training and test sets, and construct the concept lattice associated with the training set. To build the document lattice, we use the NextNeighbors algorithm, described in [4] on page 35.

4.4 Pairwise Term Proximity

To compute the proximity matrix P we need to find the pairwise term distances. This problem is solved by using a free package based on Dijkstra's algorithm to find the shortest path between two nodes in a graph. We apply Dijkstra's algorithm to any pair of term concepts in the document lattice, after removing the top and the bottom elements. If the term concept corresponding to a given term becomes disconnected (which rarely happened in our experiments), then this is signaled by the program and we assign a zero proximity value to all the pairs in which the disconnected term is involved. The computation of the pairwise term distances is the most critical step from a computational point of view, because the running time of Dijkstra algorithm for any pair of term concepts is $O(n\,logn)$, where n is the number of concepts in the lattice, and this operation must be repeated for $\frac{k(k-1)}{2}$ times, where k is the number of attributes.

4.5 Kernel Matrix

Using the proximity matrix found in the earlier step, the kernel matrix is computed with equation 4 (or equation 5 for the gaussian kernel). To take into account the relative importance of terms in the documents, we represent the two document vectors d_1 and d_2 as vectors of weighted terms, using the classical *tf-idf* scheme (i.e., term frequency times inverse document frequency) to assign a weight to each term in each document. The same weights were also used, in the experiments described below, to find the bag-of-words kernels using equations 2 and 3.

4.6 SVM Classification

To perform the SVM classification with the various kernels, we used the LIBSVM package, available at `www.csie.ntu.edu.tw/~cjlin/libsvm/`.

5 Experiments

The aim was to compare the concept lattice-based kernel (CL) to the traditional bag-of-words (BOW) kernel, when used within a SVM learning algorithm. The complexity of our method does not allow us to use high-dimensional data, on which the standard SVM has proved to perform well. So we decided to focus on small datasets, containing little training data.

We hypothesized that the reliance of the standard SVM on the input feature vectors can be questioned when there are not sufficient training data, because there is a higher chance that the terms in the test documents will not be matched, and thus it may be more difficult to identify the regions of the feature space that belong to each class. We ran two experiments. In the first, we considered datasets of varying content and size, but with the same proportion of test and training documents; in the second we kept the test set fixed and let the training set decrease until its size became a small fraction of the test set size. We now describe each experiment in detail.

For the first experiment we used the 20 NewsGroup (20NG) collection, available at `http://people.csail.mit.edu/jrennie/20Newsgroups/`, which contains 20,000 Usenet articles partitioned in 20 thematic categories. After selecting the

Table 6. Accuracy on small subsets of the 20NG collection with two random split of the data

	40 docs	80 docs	120 docs	160 docs	200 docs
linear BOW-SVM	69.23	63.75	67.27	68.57	69.56
linear CL-SVM	71.79	65.00	69.09	76.42	72.28
gaussian BOW-SVM	69.23	66.25	67.27	70.00	70.65
gaussian CL-SVM	71.79	66.25	70.00	77.14	72.28

eight most different categories (i.e., *Atheism, Computer Graphics, Misc Forsale, Autos, Sport Baseball, Medicine, Talk Religions, and Talk Politics*), we randomly chose four samples of different size (i.e., 5, 10, 15, 20, 25) from each category and merged the documents of each category contained in the corresponding samples. We thus produced five datasets of varying content and size, each containing the same number of documents for each category.

We evaluated the performance of the two document similarity metrics on each dataset. We randomly split each dataset in two halves , each with a proportional number of documents per category. We used one half for training and the other half for testing, measuring the accuracy of each method (i.e., the percentage of test documents that were correctly classified). All runs were performed with a restricted term index containing the 200 words with the highest mutual information value. We experimented with both linear kernels (equations 2 and 4) and gaussian kernels (equations 3 and 5, with $\gamma = 0.001$), so we tested four methods in all, denoted by *linear CL-SVM*, *linear BOW-SVM*, *gaussian CL-SVM*, and *gaussian BOW-SVM*. The results are shown in Table 6.

The main finding is that CL-SVM outperformed BOW-SVM for any comparable pairs of data points, i.e., the superiority of CL-SVM over BOW-SVM held for both the linear and the gaussian versions and across all five datasets. The results in Table 6 also show that the accuracy of the gaussian kernel was only slightly greater or more often just equal to that of the corresponding linear kernel.

To take into account the effect of the feature selection step on performance, we varied the value of the threshold used to select the terms with the highest mutual information scores. We repeated the tests over the same datasets with indexes ranging from few tens to several hundreds of words. We did not notice a significant change in performance, provided that the index size did not become very small. Even the relative performance of the methods was substantially stable across indexes of various size.

For the second experiment, we used the mini 20 NewsGroup collection, which is a reduced version of the full 20NG dataset. It consists of the same set of 20NG categories, but each category contains only 100 articles. To simulate critical learning conditions, we were interested in evaluating what happens when the set of documents used to learn the classifier becomes increasingly smaller. After selecting the same eight categories as in the first experiment, we randomly chose 80 documents from each category and merged the documents of each category, thus forming a dataset with 640 documents. This was the test set, kept constant throughout the

Table 7. Accuracy on a test set of 640 documents of the mini 20NG collection with increasingly smaller training sets

	160 training docs	80 training docs	40 training docs
linear BOW-SVM	58,516	59,8651	58,516
linear CL-SVM	68,1282	65,43	64,5868
gaussian BOW-SVM	61,8887	59,1906	58,516
gaussian CL-SVM	62,3946	62,3946	60,7083

experiment. The 20 remaining documents of each category were used to form several training sets. The largest had just all the 160 documents; then we created a smaller training set by randomly removing from it half of the documents in each category, thus producing a set with 80 documents (i.e., 10 per category). We iterated the same procedure two times, thus producing in all three training sets such that they could be fordered according to the set inclusion relation.

We then evaluated the performance of the two document similarity metrics on the three datasets formed by merging each training set with the same test set containing 640 documents. The results are shown in Table 7.

The main result is that the conceptual method performed better than the standard method across all kernels and datasets. This is consistent with the findings of the first experiment. The benefits of the conceptual method over the standard method are more evident with a linear kernel, although even with a gaussian kernel the conceptual method always achieved higher accuracy. As the size of the training set decreased, the performance of each method decreased very gently, while the gain in performance due to the use of the conceptual method did not grow. One possible explanation for the latter phenomenon is that with very few training documents there is less chance to recover from bad document expansion, which may more easily result in topic drift. Also, the use of the gaussian kernel together with the conceptual document similarity metric seemed to hurt performance. Here one should consider that a combination of these techniques may not work well, because both ultimately produce the same effect, namely an expansion of the terms describing each document.

6 Related Work

There are three main areas related to this work, which are described in turn below.

6.1 Semantic Kernels

One of the first semantic kernels for SVM text classification was proposed in [16]. A term similarity function based on WordNet was used, with the similarity between two terms being inversely proportional to the length of the path linking the two nodes. Such similarities were used to perform a semantic smoothing of the input feature vectors, similar to our approach. More recently [1], WordNet has been used to compute a similarity function between terms based on the sub-hierarchy rooted in their lowest common hypernym. The similarity function between two documents was then defined as the sum of the similarities of all pairs of terms in common to the documents. WordNet is a valuable source of semantic information that can improve text representation, but its use for document expansion is made difficult by several reasons, including its limited coverage, the need for sense disambiguation, and the lack of proper nouns. Another issue is the computational complexity of these algorithms. They require the evaluation of all the term pairs, for each of which it is necessary to navigate the WordNet hierarchy. Our method suffers from similar computational limitations.

Another approach to semantic kernels for text is described in [6]. Inspired by latent semantic indexing [7], which consists of projecting the data into a subspace through a singular value decomposition of the term by document matrix, Cristianini et al. showed that it is possible to apply the same technique to any kernel defined feature space. This approach is, by nature, more similar to ours because it exploits interdocument relationships, but the similarity between documents is based on extracting a subset of invariant features from the input data, rather than expanding the original feature vectors. The main limitations of latent semantic kernels are its high time complexity (i.e., to compute the eigenvalue decomposition of the kernel matrix) and the high number of dimensions required to draw relevant features from the training data. Also, the resulting features might be difficult to interpret.

6.2 Concept Similarity

In this paper we have adopted a simple term similarity measure based on edge counting, i.e, the shorter the path between two term concepts, the more related the terms associated with them. Belohlavek [2] used a similar, more restricted notion in which two attributes are similar if the distance between their corresponding attribute concepts is equal to zero. The edge counting approach takes into account the structural relationships between the subsets of attributes in the collection, but it has the disadvantage that it ignores the specific description of the nodes.

The similarity between formal concepts has been studied in a few recent papers from other perspectives. Formica [8] presents a more comprehensive method where the similarity between the intents of the concepts is explicitly considered. The overall concept similarity is obtained by combining the cardinality of the intersection of the concept extents with the similarity of the *information content* [11] of the two concept intents. The information content of a concept intent int is given by $-\ log\ p(int)$, and the information content similarity of two concept intents is computed as the ratio of the information content of the least upper bound of the two concepts to the sum of the information content of each concept intent. Another recent method that combines structural and featural information is given in [18]. It is based on the rough set theory and considers only the join-irreducible and meet-irreducible elements of the lattice to express the similarity between concepts.

6.3 Applications of Document Lattices

In the last years, a number of information retrieval applications based on document lattices have been proposed. A detailed account of the issues involved in developing concept lattice-based information retrieval applications along with a survey of approaches and systems are given in [4], spanning integration of thesauri, query refinement, text ranking, and search results clustering. Other notable applications include faceted browsing [5], and query elicitation [13]. Uta Priss [14] offers a somewhat complementary view, showing the applications of concept lattices to several subfields of information science including information retrieval.

The most similar earlier approach is [3], in which a user query is mapped on the concept lattice built from the collection being searched, and then a document ranking is automatically computed based on the topological distance between the query concept and the document concepts. This paper takes a step forward by further expanding the technical and practical scope of this research line to text classification.

7 Conclusions and Future Work

There are several ongoing attempts to extend kernel learning methods with semantic information. In this paper we have defined a concept lattice-based kernel for SVM text classification that makes use of term relations encoded in the document lattice associated with the training documents. The proposed method is formally sound and it has an intuitive meaning. Furthermore, in an experimental evaluation performed on small datasets, it achieved higher classification accuracy than the standard SVM. Our main conclusion is that the exploitation of hidden, structural document relationships beyond the input description of each document may help build a better classifier, at least when little training data are available.

We plan to extend this research in several direction: (a) using the concept lattice-based document similarity metric within other learning algorithms (e.g., K-nearest neighbors), (b) experimenting with other collections, larger datasets, and more difficult learning conditions (e.g., imbalanced data), (c) using different criteria to measure the distance between concepts in the document lattice, (d) investigating the potentials of the method based on measuring the distance between documents (rather than terms) directly in the document lattice.

References

1. Basili, R., Cammisa, M., Moschitti, A.: A semantic kernel to classify texts with very few training examples. Informatica 30, 163–172 (2006)
2. Belohlavek, R.: Similarity relations in concept lattices. Journal of Logic and Computation 10(6), 823–845 (2000)
3. Carpineto, C., Romano, G.: Order-Theoretical Ranking. Journal of the American Society for Information Science 51(7), 587–601 (2000)

4. Carpineto, C., Romano, G.: Concept Data Analysis — Theory and Applications. Wiley, Chichester (2004)
5. Cole, R., Eklund, P., Stumme, G.: Document retrieval for e-mail search and discovery using formal concept analysis. Applied Artificial Intelligence 17(3), 257–280 (2003)
6. Cristianini, N., Shawe-Taylor, J., Lodhi, H.: Latent semantic kernels. Journal of Intelligent Information Systems 18(2–3), 127–152 (2002)
7. Deerwester, S., Dumais, S.T., Furnas, W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science 41(6), 391–407 (1990)
8. Formica, A.: Concept similarity in formal concept analysis: An information content approach. Knowledge-Based Systems 21(1), 80–87 (2008)
9. Haasdonk, B.: Feature space interpretation of svms with indefinite kernels. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(4), 482–492 (2005)
10. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)
11. Lin, D.: An information-theoretic definition of similarity. In: Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin, USA, pp. 296–304. Morgan Kaufmann, San Francisco (1998)
12. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
13. Meghini, C., Spyratos, N.: Computing intensions of digital library collections. In: Kuznetsov, S.O., Schmidt, S. (eds.) ICFCA 2007. LNCS, vol. 4390, pp. 66–81. Springer, Heidelberg (2007)
14. Priss, U.: Formal concept analysis in information science. Annual Review of Information Science and Technology (ARIST) 40 (2006)
15. Schölkopf, H., Smola, A.J.: Learning with Kernels. MIT Press, Cambridge (2002)
16. Siolas, G., d'Alche Buc, F.: Support vector machines based on a semantic kernel for text categorization. In: Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000), vol. 5, pp. 205–209 (2000)
17. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, Heidelberg (1995)
18. Wang, L., Liu, X.: A new model of evaluating concept similarity. Knowledge-Based Systems 21(4), 842–846 (2008)

Two FCA-Based Methods for Mining Gene Expression Data

Mehdi Kaytoue[1], Sébastien Duplessis[2], Sergei O. Kuznetsov[3],
and Amedeo Napoli[1]

[1] Laboratoire Lorrain de Recherche en Informatique et ses Applications (LORIA)
Campus Scientifique, B.P. 235 – Vandœuvre-lès-Nancy – France
mehdi.kaytoueuberall@loria.fr, amedeo.napoli@loria.fr
[2] UMR 1136 Institut National de la Recherche Agronomique (INRA)
Nancy Université – Interactions Arbres/Micro-organismes
54280 Champenoux – France
duplessi@nancy.inra.fr
[3] State University Higher School of Economics
Kirpichnaya 33/5 – 125219 Moscow – Russia
skuznetsov@hse.ru

Abstract. Gene expression data are numerical and describe the level of expression of genes in different situations, thus featuring behaviour of the genes. Two methods based on FCA (Formal Concept Analysis) are considered for clustering gene expression data. The first one is based on interordinal scaling and can be realized using standard FCA algorithms. The second method is based on pattern structures and needs adaptations of standard algorithms to computing with interval algebra. The two methods are described in details and discussed. The second method is shown to be more computationally efficient and providing more readable results. Experiments with gene expression data are discussed.

1 Introduction

Gene expression data (GED) consist of numerical tables with thousands of genes in rows and dozens of biological environments or situations (different cells, times,...) in columns (See Table 1). Each table entry is called an *expression value* and reflects the behaviour of the gene in a row in the situation in column. A whole line is a numerical vector called the *expression profile* of the gene. Genes having similar expression profiles are said to be *co-expressed*. GED analysis is of high interest mainly for the identification of groups of co-expressed genes that are known to possibly interact together within a same biological process [1]. GED analysis is an active area of research involving mainly data-mining methods: many clustering [2], biclustering [3,4] and FCA-based [5,6,7] methods have been recently designed and applied in this domain.

Clustering methods group genes into *clusters* w.r.t. a global similarity, e.g. based on Euclidean distance, of their expression profiles. Clustering may fail to detect biological processes common only to some columns of a dataset [1,3].

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 251–266, 2009.

To overcome this difficulty, biclustering algorithms have been suggested [3,8]. *Biclusters* in GED are defined as groups of genes that have similar expression values in a same group of situations, but not necessarily all. However, we know that most of the genes are involved in several processes [1]: extracted biclusters should overlap. Then extracting "homogeneous" biclusters is difficult as the number of possible groups may grow exponentially. Biclustering algorithms generally extract k best biclusters w.r.t. an evaluation function that relies on heuristics: Their complete enumeration is generally not possible, interesting patterns may also be missed [3,5].

This problem is limited when considering binary GED, i.e. *binary relations* between the set of genes and the set of situations [4,9]. A numerical GED is scaled before binary biclusters are extracted. Intuitively, a bicluster is a rectangle in a binary table (modulo line and column permutations) "completely or mostly" filled with crosses, e.g. like in Table 4. Then a complete enumeration respecting some constraints like closure and minimal frequency is possible [4,5,10]. In [4] the authors have proposed the Bi-Max bi-clustering algorithm, which extracts *inclusion-maximal biclusters* defined as follows: Given m genes, n situations and a binary table e such that $e_{ij} = 1$ or $e_{ij} = 0$ for all $i \in [1, m]$ and $j \in [1, n]$, the pair $(G, C) \in 2^{\{1,\ldots,n\}} \times 2^{\{1,\ldots,m\}}$ is called an inclusion-maximal bicluster if and only if (1) $\forall i \in G, j \in C : e_{ij} = 1$ and (2) $\nexists (G', C') \in 2^{\{1,\ldots,n\}} \times 2^{\{1,\ldots,m\}}$ with (a) $\forall i' \in G, \forall j' \in C$: $e_{i'j'} = 1$ and (b) $G \subseteq G' \wedge C \subseteq C' \wedge (G', C') \neq (G, C)$. Note that an inclusion-maximal bicluster is nothing else than a formal concept as defined in [11]. Formal Concept Analysis (FCA) can be viewed as a binary biclustering method: It provides means for extracting local patterns from a binary relation, namely *formal concepts*. In application to GED analysis concept extents are maximal sets of genes related to a common maximal set of situations (not necessarily all) [5,6,7].

However to apply either binary biclustering or FCA-based methods, one needs to scale numerical data. Scaling introduces biases and may result in loss of information [4,5,6,7,12]. Our goal here is to try to avoid these problems by designing an FCA-based method that does not need a scaling, but would benefit from formal and computational framework of FCA.

We propose two mathematically equivalent FCA-based methods for extracting groups of co-expressed genes in numerical GED. Co-expression, or similarity is considered by using interval values in initial data. The first approach uses *interordinal scaling*. This scaling is able to reflect all possible value intervals arising from a numerical dataset by a binary relation without loss of information. However it produces large and dense binary data, which are hard to analyse with existing FCA algorithms. This is probably the reason why it has never been used for GED analysis. The second method relies on pattern structures [13] by extending interval algebra in real numbers [13,14] and does not need any scaling.

We have experimented with both methods, trying to compare the quality of their results and their computational efficiency. We show that both methods extract equivalent sets of patterns, but the method based on pattern structures is more efficient than that based on interordinal scaling, and provides with more readable and interpretable results. Processing pattern structures needs slight adaptations of the FCA framework and well-known efficient algorithms (see [15] for survey).

In Sections 2-3, we present gene expression data and related work. In Sections 4-5, we introduce and discuss both methods and an interestingness measure that allows one to prune uninteresting groups of genes. In Sections 6-7 we discuss computation and experimental results, and conclusion draws further researches.

2 Gene Expression Data

Gene expression is the mechanism that produces a protein from a gene in two steps. First the transcription builds a copy of a gene called an mRNA. Then the mRNA is translated into a protein. This mechanism differs in different biological situations: for each gene the concentration, i.e. the relative quantity, of mRNA and proteins depends on the current cell, time, etc. and reflects the behaviour of the gene. Indeed, biological processes of a living cell are based on chemical reactions and interactions mainly between proteins and mRNA. Thus, it is a major interest to measure and analyse mRNA and protein concentration to understand biological processes activated in a cell.

Microarray biotechnology is able to measure the concentration of mRNA of a gene into a numerical value called *gene expression value.* This value characterizes the behaviour of a gene in a particular cell. Microarray can monitor simultaneously the expression of a large number of genes, possibly the complete coding space of a genome. When several microarrays are considered, the expression value of a gene is measured in multiple situations or environments, e.g. different cells, time points, cells responding to particular environmental stresses, etc. This characterizes the behaviour of the gene w.r.t. all these situations and is represented by a vector of expression values called a *gene expression profile.*

A *gene expression dataset* (GED) consists of a table with n rows corresponding to genes and m columns corresponding to situations. A table entry is called an *expression value.* A table line is called an *expression profile.* For example, in Table 1, the expression value of g_1 in the situation s_1 is 5 and the expression profile for the gene g_1 is $\langle 5, 7, 6 \rangle$. In this paper, we consider the NimbleGen Systems Oligonucleotide Arrays technology[1]: expression values are ranged from 0 (not expressed) to 65535 (highly expressed).

3 Related Work

In this paper we discuss methods of extracting co-expressed groups of genes, i.e. sharing similar numerical values in some or all situations. Methods of these kind allow discovering and describing biological processes in living cells [1].

For most of the binary biclustering methods, an *l-cut* scaling is operated by using a single threshold l on expression values determined for each object (see

[1] Details on this biotechnology can be found at http://www.nimblegen.com/

[4,7,16] for threshold definitions). Expression values greater than this threshold are said to be over-expressed and encoded by 1, otherwise by 0. Then strong relations are extracted from the binary table representing genes simultaneously over-expressed. In [6], we proposed a kind of generalization[2] with an interval-based scaling of numerical data, where interval number and size were chosen by experts. Given a set of genes G, a set of situations S and a set of ordered intervals T, $(g,(s,t)) \in I$, where $g \in G$, $s \in S$, $t \in T$ and I and binary relation, means that the expression value of the gene g is the interval of index t for the situation s. Formal concepts of the context $(G, S \times T, I)$ represent groups of genes whose expression values are in same intervals for a subset of situations (may be for all situations), however these intervals are hard to determine *a priori*.

In [17], the authors present an FCA-based method to mine numerical data that does not need any scaling procedure. This is a similar approach of the two equivalent methods presented in this paper as extracted patterns are composed of intervals arising from the data whose size is less than a given parameter (see Section 6). However, in [17], no algorithm for dealing with large data was proposed and no link to interordinal scaling was made. A similar approach for the case of logical formulas was realized in [18] and [19].

4 Mining GED by Means of Interordinal Scaling

4.1 FCA: Main Definitions

Here we use standard definitions from [11]. Let G and M be arbitrary sets and $I \subseteq G \times M$ be an arbitrary binary relation between G and M. The triple (G, M, I) is called a formal context. Each $g \in G$ is interpreted as an object, each $m \in M$ is interpreted as an attribute. The fact $(g, m) \in I$ is interpreted as "g has attribute m". The two following derivation operators $(\cdot)'$ are considered:

$$\begin{aligned} A' &= \{m \in M \mid \forall g \in A : gIm\} \qquad && for\ A \subseteq G, \\ B' &= \{g \in G \mid \forall m \in B : gIm\} \qquad && for\ B \subseteq M \end{aligned}$$

which define a Galois connection between the powersets of G and M. For $A \subseteq G$, $B \subseteq M$, a pair (A, B) such that $A' = B$ and $B' = A$, is called a *(formal) concept*. Concepts are partially ordered by $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2$ ($\Leftrightarrow B_2 \subseteq B_1$). With respect to this partial order, the set of all formal concepts forms a complete lattice called the *concept lattice* of the formal context (G, M, I). For a concept (A, B) the set A is called the *extent* and the set B the *intent* of the concept. Certain data are not given directly by binary relations, they require transformation to contexts, called conceptual scaling. The choice of a scale is done w.r.t. data and goals and directly affects the size and interpretation of resulting concept lattice.

[2] Using a threshold θ is equivalent to considering the interval [θ, maximum_attribute_value].

Table 1. A gene expression data (GED)

	s_1	s_2	s_3
g_1	5	7	6
g_2	6	8	4
g_3	4	8	5
g_4	4	9	8
g_5	5	8	5

Table 2. Scale $\mathbb{I}_N := (N, N, \leq)|(N, N, \geq)$ for s_1, $N = \{4, 5, 6\}$

	$s_1 \leq 4$	$s_1 \leq 5$	$s_1 \leq 6$	$s_1 \geq 4$	$s_1 \geq 5$	$s_1 \geq 6$
4	×	×	×	×		
5		×	×	×	×	
6			×	×	×	×

4.2 Interordinal Scaling for GED

Let G be a set of genes, S a set of situations, $W \subset \mathbb{R}$ a set of expression values and I_1 a ternary relation defined on the Cartesian product $G \times S \times W$, then $\mathbb{K}_1 = (G, S, W, I_1)$ is the many-valued context representing a GED. $(g, s, w) \in I_1$ or simply $g(s) = w$ means that the gene g has an expression value w for the situation s. In the example of Table 1, $G = \{g_1, g_2, g_3, g_4, g_5\}$, $S = \{s_1, s_2, s_3\}$, and I_1 is illustrated, for example, by $g_1(s_1) = 5$, i.e. $(g_1, s_1, 5) \in I_1$. Here the objective is to extract formal concepts (A, B) from $\mathbb{K}_1$, where $A \subseteq G$ is a subset of genes that share "similar values" of W, i.e. lying in a same interval with borders arising from the data in the situations of $B \subseteq S$. To this end, we use an appropriate scale to build the derived formal context $\mathbb{K}_2 = (G, S_2, I_2)$.

A scale is a formal context (cross-table) taking original attributes of $\mathbb{K}_1$ with the derived ones of $\mathbb{K}_2$, i.e. a "plan" to construct $\mathbb{K}_2$. As attributes do not take necessarily same values, each of them is scaled *separately*. $W_s \subseteq W$ is the set of values for the attribute s and is defined for each $s \in S$ as follows: $W_s \subseteq W$ and $(g, s, w) \in I_1 \implies w \in W_s, \forall g \in G$.The following interordinal scale (see pp. 42 in [11]) can be used to represent all possible intervals of attribute values:

$$\mathbb{I}_{W_s} = (W_s, W_s, \leq)|(W_s, W_s, \geq)$$

Indeed, the extents of this scale are value intervals. $\mathbb{I}_{W_s}$ is given for the many-valued attributes s_1 in Table 2, where $W_{s_1} = \{4, 5, 6\}$.

Once a scale is chosen, conceptual scaling consists in replacing each many-valued attribute of $\mathbb{K}_1$ by a certain number of attributes to construct $\mathbb{K}_2$ w.r.t. the chosen scale. Here each many-valued attribute s is replaced by $2 \cdot |W_s|$ one-valued attributes with names "$s \leq w$" and "$s \geq w$", for all $w \in W_s$. For example, many-valued attribute s_1 is replaced by the following attributes $\{s_1 \leq 4, s_1 \leq 5, s_1 \leq 6, s_1 \geq 4, s_1 \geq 5, s_1 \geq 6\}$. Derived context $\mathbb{K}_2 = (G, S_2, I_2)$ is given in Table 3 for the attribute s_1 only. Note that this transformation is without loss of information: the many-valued context can easily be reconstructed from the formal context. For example, derived attributes for $(g_1, s_1, 5)$ are $s_1 \leq 5$, $s_1 \leq 6$, $s_1 \geq 4$, $s_1 \geq 5$. The only value in W_{s_1} respecting these predicates is 5 which is the original value.

Density of a formal context (G, M, I) is defined as the proportion of elements of I w.r.t. the size of the Cartesian product $G \times M$, i.e. density $d = |I|/(|G|.|S|)$. In the case of interordinal scaling, density of derived context $\mathbb{K}_2$ is

Table 3. $\mathbb{K}_2 = (G, S, I_2)$ for the attribute s_1

	$s_1 \leq 4$	$s_1 \leq 5$	$s_1 \leq 6$	$s_1 \geq 4$	$s_1 \geq 5$	$s_1 \geq 6$
g_1		×	×	×	×	
g_2			×	×	×	×
g_3	×	×	×	×		
g_4	×	×	×	×		
g_5		×	×	×	×	

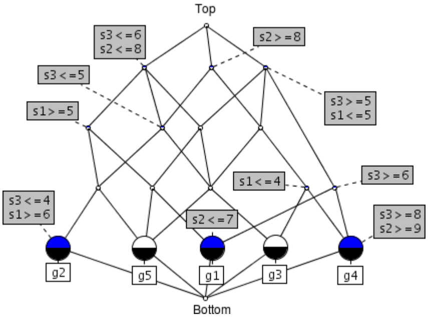


Fig. 1. Concept lattice of formal context $\mathbb{K}_2 = (G, S, I_2)$

$d = \frac{\sum_{i=1}^{i \leq p} |W_i| + 1}{2 \cdot \sum_{i=1}^{i \leq p} |W_i|}$. When $|W|$ grows, d tends towards 50%. Moreover, the number of derived attributes is $2 \cdot \sum_{i=1}^{i \leq p} |W_i|$ and $|g'| = |W| + 1$ for all $g \in G$. This makes the derived contexts dense, large and difficult to process. For comparison, density of binary data in [4] never exceeds 6% and the number of derived attributes remains the same after scaling.

Then the concept lattice of $\mathbb{K}_2$ is given in Figure 1. Concept extents near Bottom concept contain few genes, since the corresponding intents are related to the smallest intervals. Top concept extent is composed of all genes as its intent correspond to intervals of maximal length. The higher a concept lies in the diagram, the larger is the interval corresponding to its intent. Concepts near Top are not interesting: they allow for almost all possible values of attributes. In Section 5.3 we discuss how to select most interesting concepts.

5 Mining a GED with Interval Pattern Structures

Now we suggest how to equivalently mine a gene expression data as a pattern structure avoiding scaling. First we give a general intuition of interval pattern structures that are formally defined and illustrated latter.

For a many-valued context (G, S, W, I), an object $g \in G$ admits a unique description $\langle [a_1, b_1], \ldots, [a_i, b_i], \ldots, [a_p, b_p] \rangle$, where $p = |S|$. Each attribute (or situation in gene expression analysis) value is an interval (may be consisting of one point) given by its left and right limits. In our example, description of the object g_1 is $\langle [5, 5], [7, 7], [6, 6] \rangle$. A set of objects also admits a description of the form $\langle [a_1, b_1], \ldots, [a_i, b_i], \ldots, [a_p, b_p] \rangle$, where for all objects of the set, the values of all attributes lie within respective intervals. For our example, description of the set $\{g_1, g_2\}$ is $\langle [5, 6], [7, 8], [4, 6] \rangle$. This description is shared by all objects having attribute values in respective intervals, in our example by objects of the set $\{g_1, g_2, g_5\}$. The object g_3 is not contained in this set, because $g_3(s_1) = 4$ and $4 \notin [5, 6]$. The whole set of object sharing a description is closed w.r.t. a closure operator. To formalize this construction one starts from interval algebra on numbers and respective partial order on intervals. Two descriptions are comparable if all intervals of one description are contained in those of the other one, incomparable otherwise.

5.1 General Definition of Pattern Structures

Formally, let G be a set (interpreted as a set of objects), let $(D, \sqcap)$ be a meet-semilattice (of potential object descriptions) and let $\delta : G \longrightarrow D$ be a mapping. Then $(G, \underline{D}, \delta)$ with $\underline{D} = (D, \sqcap)$ is called a *pattern structure*, and the set $\delta(G) := \{\delta(g) \mid g \in G\}$ generates a complete subsemilattice $(D_\delta, \sqcap)$, of $(D, \sqcap)$. Thus each $X \subseteq \delta(G)$ has an infimum $\sqcap X$ in $(D, \sqcap)$ and $(D_\delta, \sqcap)$ is the set of these infima. Each $(D_\delta, \sqcap)$ has both lower and upper bounds, resp. 0 and 1. Elements of D are called *patterns* and are ordered by subsumption relation $\sqsubseteq$: given $c, d \in D$ one has $c \sqsubseteq d \Longleftrightarrow c \sqcap d = c$.

A pattern structure $(G, \underline{D}, \delta)$ gives rise to the following derivation operators $(\cdot)^\square$:

$$A^\square = \bigsqcap_{g \in A} \delta(g) \qquad for\ A \subseteq G,$$

$$d^\square = \{g \in G | d \sqsubseteq \delta(g)\} \qquad for\ d \subseteq D.$$

These operators form a Galois connection between the powerset of G and $(D, \sqsubseteq)$. *Pattern concepts* of $(G, \underline{D}, \delta)$ are pairs of the form (A, d), $A \subseteq G$, $d \in D$, such that $A^\square = d$ and $A = d^\square$. For a pattern concept (A, d) the component d is called a *pattern intent* and is a description of all objects in A, called *pattern extent.* Intuitively, (A, d) is a pattern concept if adding any element to A changes d through $(\cdot)^\square$ operator and equivalently taking $e \supset d$ changes A. Like in case of formal contexts, for a pattern structure $(G, \underline{D}, \delta)$ a pattern $d \in D$ is called *closed* if $d^{\square\square} = d$ and a set of objects $A \subseteq G$ is called *closed* if $A^{\square\square} = A$. Obviously, pattern extents and intents are closed.

5.2 Interval Patterns

In a pattern structure, objects have descriptions from a complete semilattice $(D, \sqcap)$, where the operation $\sqcap$ is idempotent, commutative and associative and returns “similarity” of its arguments. Here we consider a similarity operation $\sqcap$ based on an interval algebra on real numbers. For two intervals $[a, b]$ and $[c, d]$, with $a, b, c, d \in \mathbb{R}$ and $a \leq c$, we define their meet $[a, b] \sqcap [c, d]$ as $[min(a, c), max(b, d)]$. Then

$$[a, b] \sqsubseteq [c, d] \Longleftrightarrow [a, b] \sqcap [c, d] = [a, b]$$
$$\Longleftrightarrow [min(a, c), max(b, d)] = [a, b] \Longleftrightarrow a \leq c \quad and \quad b \geq d.$$

Note that in contrast to usual intuition, this definition means that smaller intervals subsume larger intervals.

An interval pattern structure $(G, (D, \sqcap), \delta)$ for a many-valued context (G, M, W, I) with $W \subseteq \mathbb{R}$ is composed of a set of objects G, a meet semilattice $(D, \sqcap)$ and $\delta : G \longrightarrow D$ a mapping that associates a description to a set of genes. The elements of D, called interval patterns, are vectors of p intervals, each interval staying for an attribute or situation. The order on elements of D is given by the natural subsumption order. For interval pattern descriptions $c = \langle [a_i, b_i] \rangle_{i \in [1,p]}$ and $d = \langle [c_i, d_i] \rangle_{i \in [1,p]}$:

$$c \sqsubseteq d \Longleftrightarrow c \sqcap d = c$$
$$\Longleftrightarrow a_i \leq c_i \quad and \quad b_i \geq d_i, \forall i \in [1, p]$$

The first operator of the Galois connection takes a set of objects (genes in our application) to their common description, which is a vector of intervals (of gene expression values). Consider two objects g_1 and g_2 with $\delta(g_1) = \langle [a_i, b_i] \rangle_{i \in [1,p]}$ and $\delta(g_2) = \langle [c_i, d_i] \rangle_{i \in [1,p]}$, then

$$\{g_1, g_2\}^{\square} = \bigsqcap \delta(\{g_1, g_2\}) = \delta(g_1) \sqcap \delta(g_2)$$
$$\{g_1, g_2\}^{\square} = \langle [min(a_i, c_i), max(b_i, d_i)] \rangle_{i \in [1,p]}$$

The second derivation operator takes a description (vector of intervals of gene expression values) to the set of all objects sharing this description (set of genes whose expression values are within intervals of the description for each attribute). Consider $d \in D$, a pattern such that $d = \langle [a_i, b_i] \rangle_{i \in [1,p]}$, then

$$d^{\square} = \{g \in G \mid d \sqsubseteq \delta(g)\}$$
$$d^{\square} = \{g \in G \mid d \sqcap \delta(g) = d\}$$
$$d^{\square} = \{g \in G \mid \delta(g) = \langle [c_i, d_i] \rangle_{i \in [1,p]}, a_i \leq c_i \ and \ b_i \geq d_i, \ \forall i \in [1, p]\}$$

For a many-valued context (G, M, W, I) with $W \subset \mathbb{R}$ consider the respective pattern structure $(G, (D, \sqcap), \delta)$ on intervals, the interordinal scaling $\mathbb{I}_{W_s} = (W_s, W_s, \leq) \mid (W_s, W_s, \geq)$ from the previous Section, and the context K_I resulting from applying interordinal scaling $\mathbb{I}_{W_s}$ to (G, M, W, I). Consider usual derivation operators $(\cdot)'$ in context K_I. Then the following obvious proposition establishes an isomorphism between the concept lattice of K_I and the pattern concept lattice of $(G, (D, \sqcap), \delta)$.

Proposition. Let $A \subseteq G$. Subset A is an extent of the interval pattern structure $(G, (D, \sqcap), \delta)$ iff $A \subseteq G$ is a concept extent of the context K_I. Moreover, $A^{\square} = \langle [\underline{m}_i, \overline{m}_i] \rangle_{i \in [1,p]}$ iff for all $i \in [1, p]$ $\underline{m}_i$ is the largest number n such that the attribute $\geq n$ is in A' and $\overline{m}_i$ is the smallest number n such that the attribute $\leq n$ is in A'.

Consider an example of pattern concept: $(\{g_1, g_2, g_5\}, \langle[5,6],[7,8],[4,6]\rangle)$, the equivalent concept of the interordinally scaled context is $(\{g_1, g_2, g_5\}, \{s_1 \leq 6, s_1 \geq 4, s_1 \geq 5, s_2 \geq 7, s_2 \leq 8, s_2 \leq 9, s_3 \leq 6, s_3 \leq 8, s_3 \geq 4\})$. The top pattern concept is $(G, \langle[4,6],[7,9],[4,8]\rangle)$. The higher is a concept in the lattice diagram, the larger are the intervals of its pattern intent, in particular, the top pattern concept has maximal intervals. In the next section we consider the problem of selecting most interesting concepts given by small intervals.

5.3 Interestingness of a Pattern Concept

The main goal of GED analysis is extracting homogeneous groups of genes, i.e. groups of genes having similar expression values. Descriptions of homogeneous groups should be composed of intervals with "small" sizes. This can be easily expressed in terms of interval-based patterns. Consider parameter max_{size} that specifies the maximal length of an interval to allow for the whole description represent a homogeneous group of genes. Then in our experiments we may restrict only to pattern concepts with pattern intents $c = \langle[a_i, b_i]\rangle_{i\in[1,p]} \in D$ satisfying the constraint: $\exists i \in [1,p]\ (b_i - a_i) \leq max_{size}$. A stricter constraint would be $\forall i \in [1,p]\ (b_i - a_i) \leq max_{size}$.

Since both constraints are monotone (if an extent does not satisfy it, than a larger intent does not satisfy it too), the subsets of patterns satisfying any of these constraints make an order ideal of the lattice of pattern intents. In terms of computation, using any of these constraints means that only some lower part of the pattern lattice is computed, with patterns satisfying the constraints.

Another possibility is to consider additional $*$-values of interval descriptions replacing intervals, whose lengths exceed threshold max_{size}. So, if we choose $max_{size} = 1$ in our example, then $\{g_1, g_2\}^\square = \langle[5,6],[7,8],*\rangle$ and $\{g_1, g_4\}^\square = \langle[4,5],*,*\rangle$.

6 Computation

Many algorithms for generating formal concepts from a formal context are known, see e.g. a performance comparative [15]. Two families of algorithms are distinguished: incremental and batch ones. At the i^{th} step an incremental algorithm builds the set of concepts for i first objects. Batch algorithms generate sets of concepts from scratch, in a top-down way (resp. bottom up) or from maximal to minimal intents (resp. from minimal to maximal intents). Experimental results of [15] highlight *Norris* (incremental), *CloseByOne* and *NextClosure* (both bottom up) algorithms as best algorithms when the context is dense and large, which is the case of interordinal derived formal contexts.

To compute pattern concepts, one needs generating infima of subsets of D_δ. To this end we have chosen the standard FCA algorithms *Norris*, *CloseByOne*, and *NextClosure*, which need only slight modifications for computing in pattern structures [13]. Computing $A^\square$ for a set $A \subseteq G$ is realized by taking *min* (resp. *max*) of all left (resp. right) limits of the intervals of each object description.

For a pattern $d \in D$, $d^{\square}$ is computed by testing for each object $g \in G$ if each interval of its description is included in the corresponding interval of d.

Worst-case upper bound time complexity of the three highlighted algorithms for computing in a formal context (G, M, I) is $O(|G|^2 \cdot |M| \cdot |L|)$ with G the set of genes, M the set of attributes and L the set of generated concepts [15]. The worst-case time complexity of computing the set of interval pattern structures is $O(|G|^2 \cdot p \cdot |L|)$, where p is the number of components in a description. In both cases, the sets G and L are the same, thus relative efficiency of processing both data representations depends on the number of different attribute values. For a large number of values the time for computing concepts for the interordinally scaled context may be too large. A projection should reduce the number of different attribute values, and also the number of concepts. A simple way is to round real attribute values to n digits after the comma or to a multiple of 10. A direct consequence of this transformation is uncontrolled loss of information which we would like to avoid. However, in this case we just loss precision on attribute values that has limited consequences compared to the binary transformations presented in Related work.

7 Experiments and Results

7.1 Data

Biologists at the UMR IAM (INRA) study interactions between fungi and trees. They recently published the complete genome sequence of the fungus *Laccaria bicolor* [20]. This fungus lives in symbiosis with many trees of boreal, montane and temperate forests. The fungus forms a mixt organ on tree roots and is able to exchange nutrients with its host in a specific symbiotic structure called ectomycorrhiza, contributing to a better tree growth and enhancing forest productivity. On the other hand, the plant retributes its symbiotic partner by providing carbohydrates, allowing the fungus to complete its biological cycle by producing fruit-bodies (e.g. mushrooms). It is thus a major interest to understand how the symbiosis performs at the cellular level. The genome sequence of *Laccaria bicolor* contains more than 20,000 genes [20]. It remains now to study their expression in various biological situations in order to help to understand their roles and functions in the biology of the fungus. Microarray-based gene expression study of different situations is a solution of choice. For example, it enables to compare expression values of all the genes between contrasted situations like free-living cells of the fungus (i.e. mycelium), cells engaged in the symbiotic association (i.e. ectomycorrhiza), and specialized cells forming the fruit-body structure (i.e. mushroom). A *Laccaria bicolor* gene expression dataset is available at the Gene Expression Omnibus of the National Center for Biotechnology Information (NCBI)[3]. It is composed of 22,294 genes in lines and 5 various biological situations in columns, reflecting cells of the organism in various stages of its biological cycle, i.e. free living mycelium (situation FLM), symbiotic tissues (situations MP and MD) or fruiting bodies (situations FBe and FBl).

[3] `http://www.ncbi.nlm.nih.gov/geo/` as series GSE9784

7.2 Biological Experiments

First, a selection from the 22,294 genes is processed. It consists by removing genes having no significant difference of expression across all situations. For each couple of situation, a t-test is performed and a p-value is attributed. If the p-value > 0.05 (cut-off classically applied in biology) for all couples of situations then the current gene is removed from the dataset. Indeed, a gene that shows similar expression values in all situations presents less interest to the biologist than a gene with high differences of expression. Significant changes in gene expression may reflect a role in a biological process and such genes can help the biologist to draw hypotheses. The CyberT tool, available at http://cybert.microarray.ics.uci.edu/, was used to filter the dataset and obtain 10,225 genes.

Another classical processing in GED analysis is to turn values into log_2. Indeed, it allows to capture small expression values into intervals that should be larger for high expression values. Finally, the projection consists in rounding log_2 expression values to one digit after the comma.

We ran the *CloseByOne* algorithm on the obtained interval pattern structure. In this settings, we set $max_{size} = 0.35$. We choose to retain a concept iff its minimal support is greater than 10. Indeed, let us recall that concepts near Bottom, i.e. in the lowest levels of the concept lattice, are composed of a few genes. We also choose to retain a concept iff its description d has a size $|d| \geq 4$: its extent is composed of genes having similar values in at least four situations. Processing lengths 4.2 hours and returns $91,681$ concepts.

Here we present two extracted patterns that group genes with high expression levels in the fruit-bodies situations, whereas their expression remains similar between the mycelium and symbiosis situations (Figure 2). In both patterns, the levels measured are about twice higher in the fruit-body compared to the other situations indicating that these genes correspond to biological functions of importance at this stage.

The first pattern contains 7 genes, of which only 3 possess a putative cellular function assignment based on similarity in international gene databases at NCBI. Interestingly, these genes all encode enzymes involved in distinct metabolic pathways. A gene encodes a 1-pyrroline-5-carboxylate dehydrogenase

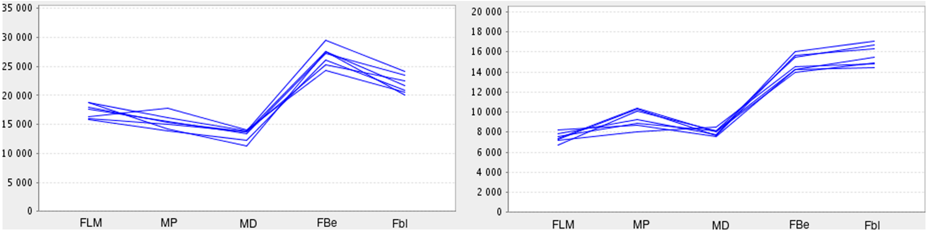


Fig. 2. Graphical visualisation of two extracted concepts. X-axis is composed of situations, Y-axis is the expression values axis. Each line denotes the expression profile of a gene in the concept extent. Values are taken before the logarithmic transformation.

which is involved in amino-acid metabolism, another correspond to an acyl-coA dehydrogenase, involved in fatty acid metabolism and a last gene encodes a transketolase, an enzyme involved in the pentose phosphate pathway of carbohydrate metabolism. All these metabolic functions are essential for the fungus and reflect that the fruit-body is a highly active tissue. The fruit-body is a specific fungal organ that differentiate in order to produce spores and that further ensure spore dispersal in nature [21]. Previous gene expression analyses of the fruit-body development conducted in the ectomycorrhizal fungus *Tuber borchii* also reported the strong induction of several genes involved in carbon and nitrogen metabolisms [22] as well as in lipid metabolism [23]. The present results are consistent with these observations and supports an important mobilization of nutrient sources from the mycelium to the fruit-body. It seems obvious that the primary metabolism requires to be adapted to use these sources in order to properly build spores and provide spore-forming cells with nutrients [21].

The second pattern also contains 7 genes, of which only 3 possess a putative biological function. Interestingly, one of these genes encodes a pseudouridylate synthase, an enzyme involved in nucleotide metabolism that might also be involved in remobilization of fungal components from the mycelium to spore-forming cells and spores. The 2 other genes encode a cytoskeleton protein (actin) and a protein related to autophagy (autophagy-related 10 protein), a process that can contribute to the recycling of cellular material in developing tissues. Both functions participate to reconstruction cellular processes [21], which is consistent with involvement of metabolic enzymes in remobilization of fungal resources towards the new organ in development.

Analysis of these two patterns that present a high expression level in the fruit-body situation is highly informative, comforts existing knowledge in the field and highlights the importance of remobilization in the developing organ. These co-expressed genes share related roles in a particular process. This could indicate that they are under the control of common regulators of gene expression. Interestingly, these patterns also contained a total of 8 genes of unknown functions, i.e. for which no functional assignment was possible in international gene databases. There were 4 genes encoding hypothetical proteins with an homology in databases but no detailed function and 4 genes not previously described in fungi or other organism and which are considered specific to *Laccaria bicolor*. There are about 30% of such genes specific to this fungus and these may play specific roles in the biology of this soil fungus [20]. All these genes show consistent profiles with those encoding metabolic functions. Thus, these genes are interesting investigation leads as they may contain new enzymes not previously described of the pathways or eventual regulator of the cellular process. Altogether, these results contribute to a better understanding of the molecular processes underlying the fruit-body development.

7.3 Computer Experiments

Now we compare time performance and memory usage of three algorithms to equivalently mine interordinal formal contexts (Section 4) and interval pattern structures (Section 5). We have implemented the *Norris*, *NextClosure*, and *CloseByOne* algorithms, for both processing formal contexts and pattern structures. We have added the Charm algorithm [24] that extracts closed itemsets, i.e. concept intents, in a formal context. FCA algorithms have been implemented in original versions, plus the stack optimization for *NextClosure* and *CloseByOne* as described in [15]. For interval pattern structures, we operate sligth modifications. Charm algorithm is run with the Coron Toolkit [25]. All implementations are in Java: sets of objects and binary attributes are described with the BitSet class and interval descriptions with standard double arrays. Computation was run on Ubuntu 8.10 OS with Intel Core2 Quad CPU 2.40 Ghz and 4 Go RAM.

We tried to compare algorithms on the data presented in biological experiments, i.e. an interval pattern structure from a many-valued context (G, S, W, I_1) where $|G| = 10,225$ and $|S| = 5$. Even with projections, computation is infeasible. Indeed we do not consider here constraints like the maximal interval size: we compute all infima of subsets of D_δ. Then we randomly selected samples of the data, by increasing the number of objects. As attribute values are real numbers with about five digits after the comma, the size of W is large. In worst case, $|W| = |G| \times |S|$, i.e. each attribute value is different in the dataset. This implies very large formal contexts to process and a large number of concepts. We compare time usage for this case, see Table 4. *Norris* algorithm draws best results in formal contexts, which meets conclusions of [15] for large and dense contexts. However, *CloseByOne* in pattern structures is better, and most importantly is the only one that enables computation of very large collection of concepts.

When strongly reducing the size of W by rounding attribute values to the integer, i.e. $|W| << |G| \times |S|$, the *Charm* algorithm outperforms the others. The *Norris* algorithms is still the best FCA-algorithms in formal contexts and *CloseByOne* the best in pattern structures (see Table 5).

Table 4. Generation time in both data representations (no projection)

Datasets							
$\|G\|$	10	20	30	40	50	75	100
$\|W\|$	50	100	150	199	249	374	252
density	51.00%	50.50%	50.33%	50.25%	50.20%	50.13%	50.20%
Generation time in formal contexts (in milliseconds)							
Charm	60	916	16,469	N/A	N/A	N/A	N/A
Next Closure	5	145	1,299	12,569	68,969	N/A	N/A
Norris	**2**	**90**	**609**	**5,180**	**28,831**	N/A	N/A
Close By One	3	106	906	7944	41,238	N/A	N/A
Generation time in pattern structures (in milliseconds)							
Next Closure	6	100	763	5,821	35,197	N/A	N/A
Norris	6	172	1982	15,522	83,837	N/A	N/A
Close By One	**2**	**85**	**585**	**3,094**	**18,320**	**1,004,073**	**2,288,200**
Concept set L							
$\|L\|$	280	9,587	78,173	455,008	1,857,725	40,325,176	64,571,385

Table 5. Generation time in both data representations. Attribute values are rounded.

Datasets							
$\|G\|$	25	50	75	100	125	150	200
$\|W\|$	34	37	44	53	58	62	66
Generation time for formal contexts (in milliseconds)							
density	51.47%	51.35%	51.14%	50.94%	50.86%	50.81%	50.76%
Charm	55	**154**	**184**	**243**	**394**	**936**	**1856**
Next Closure	100	933	3,333	22,973	30,854	78,790	593,416
Norris	**38**	320	861	2,697	5,954	15,359	46,719
Close By One	84	483	2,424	8,452	22,173	59,070	227,432
Generation time for pattern structures (in milliseconds)							
Next Closure	59	372	1,924	6,215	15,417	42,209	143,501
Norris	44	479	2,602	7,243	16,257	40,991	109,814
Close By One	**40**	**220**	**1,084**	**3,832**	**9,289**	**23,989**	**89,804**
Concept set L							
$\|L\|$	1,165	5,928	23,962	48,176	73,463	163,316	252,515

Then, when the number of attribute values w.r.t. $|G| \times |S|$ is low, computing concepts in formal contexts is more efficient. For large datasets with many different attribute values, it is more efficient to compute with pattern structures. The explanation is that for formal concepts the object intent representation is a bit string whose length increases with the growth of $|W|$. Object descriptions in pattern structure are arrays of constant size w.r.t. $|W|$. Therefore, processing them uses less memory for datasets with high number of attribute values.

8 Conclusion

In this paper we discussed FCA-based methods for mining complex data like gene expression data. We compared two mathematically equivalent methods for processing numerical intervals: the first one using interordinal scaling and classical FCA algorithms, and the second one which relies on interval pattern structures. Pattern structures offer more concise representation, better scalability, and better readability of the (pattern) concept lattice. Experiments show that when the number of distinct attribute values is large, adaptation of *ClosebyOne* to pattern structures is most efficient. We also confirmed a general conclusion of [15] for the case of interordinal scaled contexts of our dataset, stating that the *Norris* algorithm is more efficient than *NextClosure* and *CloseByOne* when only the set of concepts needs to be generated, not the covering relation of the lattice.

We have shown how algorithms for processing interval pattern structures can be adapted for particular data and goals. Indeed, the first introduced order of description elements generates all possible descriptions w.r.t. the similarity operation. For GED analysis we have made some propositions to retain "best" concepts. Many other possibilities should be investigated. Another direction of further research may be models accounting for domain knowledge. The semilattice of descriptions $(D, \sqcap)$ may be viewed as an attribute hierarchy, where domain knowledge (e.g. known functions of genes) may be encoded.

Acknowledgments

The third author was supported by the project of the Russian Foundation for Basic Research, grant no. 08-07-92497-NTsNIL_a. Other authors were supported by the "Contrat de Plan Etat Région – Modélisation des biomolécules et leurs intéractions" (Lorraine, France, 2007–2013).

References

1. Stoughton, R.B.: Applications of DNA microarrays in biology. Annual Review of Biochemistry 74(1), 53–82 (2005)
2. Jiang, D., Tang, C., Zhang, A.: Cluster analysis for gene expression data: a survey. IEEE Trans. on Knowledge and Data Engineering 16(11), 1370–1386 (2004)
3. 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)
4. Prelic, A., Bleuler, S., Zimmermann, P., Wille, A., Buhlmann, P., Gruissem, W., Hennig, L., Thiele, L., Zitzler, E.: A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics 22(9), 1122–1129 (2006)
5. Blachon, S., Pensa, R., Besson, J., Robardet, C., Boulicaut, J.F., Gandrillon, O.: Clustering formal concepts to discover biologically relevant knowledge from gene expression data. In Silico Biology 7(4–5), 467–483 (2007)
6. Kaytoue, M., Duplessis, S., Napoli, A.: Using formal concept analysis for the extraction of groups of co-expressed genes. In: An, L.T.H., Bouvry, P., Tao, P.D. (eds.) Modelling, Computation and Optimization in Information Systems and Management Sciences (2008)
7. Motameny, S., Versmold, B., Schmutzler, R.: Formal concept analysis for the identification of combinatorial biomarkers in breast cancer. In: Medina, R., Obiedkov, S. (eds.) ICFCA 2008. LNCS (LNAI), vol. 4933, pp. 229–240. Springer, Heidelberg (2008)
8. Cheng, Y., Church, G.M.: Biclustering of expression data. In: Proc. 8th International Conference on Intelligent Systems for Molecular Biology (ISBM), pp. 93–103 (2000)
9. Tanay, A., Sharan, R., Shamir, R.: Discovering statistically significant biclusters in gene expression data. Bioinformatics 18(suppl. 1), S136–S144 (2002)
10. Robardet, C., Pensa, R.G., Besson, J., Boulicaut, J.-F.: Using classification and visualization on pattern databases for gene expression data analysis. In: Proceedings of the Intl. Workshop on Pattern Representation and Management, Heraklion, Hellas, March 18 (2004)
11. Ganter, B., Wille, R.: Formal Concept Analysis. Springer, Berlin (1999)
12. Choi, V., Huang, Y., Lam, V., Potter, D., Laubenbacher, R., Duca, K.: Using formal concept analysis for microarray data comparison. J. Bioinform. Comput. Biol. 6(1), 65–75 (2008)

13. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: Delugach, H.S., Stumme, G. (eds.) ICCS 2001. LNCS, vol. 2120, pp. 129–142. Springer, Heidelberg (2001)
14. Kuznetsov, S.O.: JSM-method as a machine learning method. Itogi Nauki i Tekhniki, ser. Informatika 15, 17–50 (1991)
15. 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)
16. Pensa, R.G., Besson, J., Boulicaut, J.F.: A methodology for biologically relevant pattern discovery from gene expression data. Discovery Science, 230–241 (2004)
17. Messai, N., Devignes, M.D., Napoli, A., Smaïl-Tabbone, M.: Many-valued concept lattices for conceptual clustering and information retrieval. In: 18th biennial European Conference on Artificial Intelligence (2008)
18. Chaudron, L., Maille, N.: Generalized formal concept analysis. In: International Conference on Conceptual Structures (ICCS), pp. 357–370 (2000)
19. Ferré, S., Ridoux, O.: A logical generalization of formal concept analysis. In: International Conference on Conceptual Structures (ICCS), pp. 371–384 (2000)
20. Martin, F.: The genome of laccaria bicolor provides insights into mycorrhizal symbiosis. Nature 452(7183), 88–92 (2008); 69 co-authors wrote this paper
21. Busch, S., Braus, G.H.: How to build a fungal fruit body: from uniform cells to specialized tissue. Molecular Microbiology 64, 873–876 (2007)
22. Lacourt, I., Duplessis, S., Abba, S., Bonfante, P., Martin, F.: Isolation and characterization of differentially expressed genes in the mycelium and fruit body of tuber borchii. Applied and Environmental Microbiology 68, 4574–4582 (2002)
23. Gabella, S., Abba, S., Duplessis, S., Montanini, B., Martin, F., Bonfante, P.: Transcript profiling reveals novel marker genes involved in fruiting body formation in tuber borchii. Eukaryotic Cell 4, 1599–1602 (2005)
24. Hsiao, C.J., Zaki, M.J.: Efficient algorithms for mining closed itemsets and their lattice structure. IEEE Trans. on Knowl. and Data Eng. 17(4), 462–478 (2005)
25. Szathmary, L.: Symbolic Data Mining Methods with the Coron Platform. PhD Thesis in Computer Science, Univ. Henri Poincaré – Nancy 1, France (November 2006)

Ontology-Based Formal Concept Differences Analysis in Radiology Report Impact by the Adoption of PACS

Telung Pan and Kwoting Fang

Information Management Department
National Yunlin University of Science & Technology
123 University Road, Section 3, Douliou, Taiwan
telung@mac.com, fangkt@yuntech.edu.tw

Abstract. Following the advent of information technology and the rapid growth of its application in the medical field, the picture archiving and communication system (PACS) became very popular in mid- to large-scale hospitals. This study aimed to compare the concept lattice of radiology report content before and after the adoption of PACS. This study proposes a formal concept analysis process to produce different levels of ontological concepts from the radiology report. Ontology concept lattice diagrams are presented along with the correlation of concepts and their corresponding significance. The conceptual clustering method is applied to obtain a reasonable number of concepts. To find differences, the structure of ontology is compared in different phases by using a distance measurement. The study results showed a delicate change in radiology report terms before and after the adoption of the system. Given the quantitative analysis results, a qualitative-focused ethnography will be further conducted with several radiologists.

Keywords: Formal concept analysis, radiology report, medical imaging system, PACS.

1 Introduction

Following the advent of information technology and the rapid growth of its application in the medical field, the picture archiving and communication system (PACS) gradually gained popularity in mid- to large-scale hospitals. Medical image storage, management, transmission, and even diagnosis now heavily rely on computer and network technology. Yet, given the vigorous development of medical information technology, there may be an impact on the medical behavior and treatment of physicians or other medical experts when they diagnose.

One of the main activities of a hospital radiology department is to provide reports of radiology opinions and interpretations of medical images to physicians. Several studies [1-4] have investigated the effect of the reporting process, but most of these studies focused on the impact of the computerized radiography reporting system at its initial adoption. The call for understanding the impact not only on the radiology reporting process but also on the diagnosis of physicians has become loud and clear. Some studies [5-8] have addressed the impact of voice-entry radiology reporting

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 267–283, 2009.

systems, but the effect of PACS in is not clear. Balassy et al. [6] found that Liquid Crystal Display (LCD) is significantly superior to Cathode Ray Tube (CRT) for lung images; however, some studies [8-11] found no significant difference. These studies showed mixed results and were mainly focused on report time. Thus, explicit knowledge about the content variation of radiology reports and diagnosis behavior differences after the adoption of PACS has yet to be developed.

On the basis of the assumption that the conceptual knowledge of radiologists' reporting is original from his/her network model, an analysis with a network model can reveal the characteristics of cognition for each individual. Semantic structure or conceptual knowledge can be obtained by elaborating on conceptual structures inherent in the text data given by radiologists. Such conceptual structures can be determined by mathematical methods using Formal Concept Analysis (FCA) [22].

This study aimed to build ontology from the radiology report text in a different time phase through FCA and the ontology comparison method. It strives to identify the variations and factors arising after the adoption of PACS in a hospital. In this study, we will lay the groundwork for understanding how PACS affects radiology diagnosis and reporting.

1.1 PACS Technology

With the exception of face-to-face diagnosis, the radiology image and report is one of the primary means of communication, as reported by radiologists, between the patients and the referring physician. Significant resources are devoted to the radiology reporting service and the quality of medical image. Computers have been used in various ways to assist in radiology [4], including image digitalization, report entry, and image quality control. In this situation, PACS offers a solution to the problems of image and information management.

Essentially, PACS handles medical images from various imaging modalities (machines), including X-ray, ultrasound, endoscopy, computerized tomography (CT), magnetic resonance imaging (MRI), and positron emission tomography (PET). PACS replaces hard copies of images with their digital forms, for example, film archives not only provide a new way to manage medical images but also expand the possibilities of off-site viewing, reporting, tele-diagnosis, and distance discussion. Given the sharp drop in the price of digital equipment and storage, PACS undoubtedly provides effective cost and space advantages as compared to conventional file archives.

The major objective of PACS is to clearly and definitively present patients' health situations, which could involve improving image diagnosis accuracy and reducing patients' radiation exposure. From the viewpoint of strategy, PACS has been recognized as a way to improve a hospital's competitiveness. Up to 2008, an increasing number of medical centers and regional hospitals in Taiwan have adopted PACS.

The actual influence of PACS on medical behavior, as given by physicians when they implemented PACS, is still unclear. Jean et al. [1] and Choplin et al. [3] considered the impact of a computerized radiography reporting system on film reporting, turnaround time, costs, and understanding of work patterns. Leaming et al. [13] assessed the impact of voice-entry radiology reporting systems on the accuracy of radiologic reports. From a technological perspective, several studies reviewed different computer methods designed to assist in the production of radiology reports [13, 14].

Several other studies have investigated users' views of the quality of radiology services and the impact of PACS. Bryan et al. [15] evaluated a hospital-wide PACS by a "before and after" comparison. They pointed out that PACS greatly reduced the perceived problem of image non-availability but did not increase the availability of radiological reports. Moreover, the time junior doctors spent in image searching was dramatically reduced by the introduction of PACS. In two studies on radiologists, Franken et al. [16] indicated that radiologists spent twice as long interpreting PACS images as they did interpreting film images, but there was no significant difference in the time spent for both approaches; furthermore, Wilcox et al. [18] surveyed radiologists to assess their perception of the value of access to images and found the impact of PACS is positive.

Occasionally, the radiology report is the only means of communication between radiologists and referring physicians. It must clearly describe the nature of the examination, pertinent findings, considerations of likely diagnoses and their relative possibilities, and suggestions for further diagnostic evaluation if indicated [17]. In fact, the way in which radiologists perceive radiology reports is generally unclear.

1.2 Ontology and Domain Knowledge Concept Representation

In the field of computer science, ontology is the description of the concepts and relationships that play the role of an agent or a community of agents [36]. In general, any content of formally represented knowledge is based on a conceptualization: the objects, concepts, and other entities that are presumed to explicitly or implicitly exist in some area of interest and the relationships that hold them [21], and ontology is an explicit specification of a conceptualization [22]. Kahn et al. [19] suggested that ontology could represent radiological and clinical knowledge in order to integrate PACS to support radiology interpretation.

Some studies [37-39] attempted to build further on domain ontology using the FCA method. A number of studies [24-25] and related technologies in other fields apply the benefits of domain ontology construction and computation for knowledge representation and applications, including the comparison of different ontologies. [26-27].

There are two main approaches for domain ontology construction [38]. One facilitates manual ontology engineering by providing natural language processing, such as lexical entry. This method has its limitations and is ineffective in large text mining [28-31]. The other approach relies on machine learning and automated language-processing techniques [7]. There are text clustering, association rules and knowledge base. Text clustering dealing with a large number of dimensions and a large amount of data can be problematic because of time complexity [32]. The use of association rules or knowledge base methods as construction methods for ontology still causes several restrictions, especially in combinations of different conceptual relationships [33], and the prior construction of knowledge bases in related domain and human effort is required [34].

As the meaning of words and the cognition of concepts differ in different times and environments, a variation might appear in the same domain (same category of radiology reports). Because of different factors, the same radiologist might use different words for the same object or use the same word for different objects, or radiologists might have different perceptions about specific concepts. Such potential for heterogeneity causes problems in a critical medical environment in which consistency and quality are pursued. Differences in ontology can be established from radiology report text in different time phases of a hospital—that is, before and after the adoption of a specific system, which in this study is PACS.

2 Research Process and Pilot Analysis

In this section, we will describe the goals of this study in more detail. Figure 1 shows an overview of the process of this study. At the beginning, a pilot study was conducted to enhance the knowledge of environmental changes after the adoption of PACS. Following this and after a term parsing process, a formal concept analysis, will deal mainly with radiology report text in different time periods (before and after the adoption of PACS). Term frequency and many-valued contexts will be considered. On the basis of the ontology lattice acquired from a formal context, comparison among different ontologies associated with different conditions will be performed using a distance measurement method. Concepts will be clustered to minimize the number of concepts for interpretation and discussion purposes.

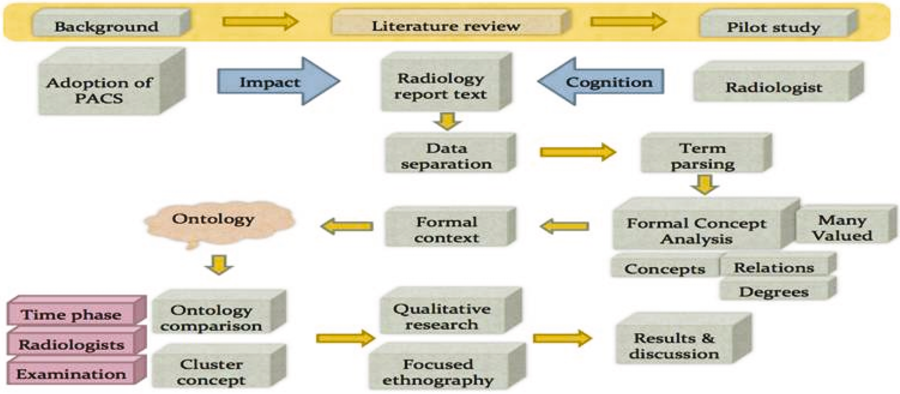


Fig. 1. Research process of this study

2.1 Settings and Data Collection

Research data were derived from the Radiology Information System (RIS) of a teaching hospital in central Taiwan. This hospital adopted PACS technology on May 16, 2006. It was implemented on radiology modalities, including projection (plain) radiography, fluoroscopy, CT scanning, and MRI. Because of function and cost considerations, some modalities, such as mammography, DXA (dual energy X-ray absorptiometry), and some ultrasound machines are still not connected to PACS.

There are about 7000 radiology-related diagnoses made each month at this teaching hospital. The analysis data of this study covered data from March 16, 2005 to March 15, 2007 and was split into two portions. The first portion of the data covers the first year and is called the traditional film phase. The portion covers the second year and is called the PACS phase. Figure 2 presents an example of the content of an abdomen CT scan radiology report. The content of the report was mainly given by one of the radiologists from the radiology department according to his personal judgment of the medical images.

The related attributes of record data for every radiology diagnosis in the RIS database are shown in Table 1, along with attribute names and descriptions. These include report text, examination item code, examination area, date, the International Classification of Diseases 9th revision code (ICD9) for diagnosis expression, and the PACS ID number.

Table 1. Related attributes of radiology data

Attribute Name (Abbreviation)	Description
Report text (REP_TEXT)	Content of the radiology diagnosis report.
Examination item code (ITEM_CODE)	Code of the radiology examination item.
Examination area (AREA)	Examination area of patient's body.
Examination date (PHO_DATE)	Examination date of patient.
Radiologist code (REP_DOC_NO)	Code of the radiologist in the department of radiology.
ICD9 codes of diagnosis (ICDH_CODE)	Diagnostic disease ICD9 code from attending physician, maximum 5 different ICD9 codes (from ICDH_CODE1 to ICDH_CODE5). These codes are provided by attending physicians and referenced only by radiologists.
PACS ID number (ORDERNO)	Only when this attribute has the ID number to show this record has PACS image.

2.2 Pilot Study

In order to enhance knowledge about the possible variations in diagnosis quantity and modification on radiology examination and report after the adoption of PACS, a pilot study was conducted using the same data that is used in this proposed research. There are three main steps in the pilot study. First, for comparison, based on the time of adoption of PACS, some statistic analyses were performed between the two phase groups; the results are presented in Table 2. There are over 150 different radiology examination items in this teaching hospital, and here, we list only few, including the code, examination quantity, and variations between the two phases.

We further combine these radiology diagnosis items by the type of modalities shown in Table 3. With regard to the modality type, there has been a significant reduction in non-PACS modalities in the PACS phase group (–3.94% for Fluoroscopy and –12.45% for DXA). On the basis of these statistical results, a qualitative study is required in order to obtain more details. One radiographer and a radiology physician were interviewed and asked questions on the decrease in the examination number for non-PACS modalities after the adoption of PACS. Furthermore, possible reasons for this decrease were investigated.

Table 2. Example of statistical data for the two phase groups

Item Item Code	Examination Item	2005-2006 Film Phase	2006-2007 PACS Phase	Difference
70001	Chest View	29,411	28,431	–3.33%
700011	Clavicle	11		–100.00%
70003	Sternum	6		–100.00%
70040	Abdomen	1,166	934	
70055	K.U.B Examination	10,384	10,930	5.26%
700601	Foot	1,178	1,379	17.06%
700602	Ankle	1,324	1,531	15.3%
700603	Tibia & Fibula	969	990	2.17%
...	…	… …	… …	…

The qualitative interview results for the first step of the pilot study are as follows: (1) The examination quantity for every modality should be increased to coincide with the volume of business for this hospital; thus, the main reason for the decrease in examination numbers for non-PACS modalities is clearly connected to PACS. (2) PACS' influence on the convenience and quality of medical images and on the willingness of physician to use specific radiology examinations is self-evident. (3) The decrease in non-PACS modalities examination numbers has no relationship with medical demands.

Table 3. Statistical data for modality types in the two phase groups

Diagnosis Type	Film Phase	PACS Phase	Difference	Non-PACS
Projection Radiography	67289	67595	0.45%	
Mammography	1978	2514	27.10%	
Fluoroscopy	2183	2097	-3.94%	*
DXA	795	696	-12.45%	*
CT	3494	4048	15.86%	
MRI	392	943	140.56%	

On the basis of the primary results of the first step of the pilot study, we further analyze whether there are variations in disease diagnosis code (ICD9) for each attending physician. Because there are complex and numerous variables that influence a physician, we limited all inspections to the same order physician, same modality (CT), same examination code (70500, CT scan for no developer), same examination area (brain), and same radiology reporter. The statistical results are shown in Table 4. The unknown status for the diagnosis code increased after the adoption of PACS. According to the decrease in the ICD9 code ending with 9, the indeterminate diagnoses clearly reduce.

Table 4. Counting of diagnosis category

Distinct diagnosis categories	Diagnosis I	Diagnosis II	Diagnosis III	Diagnosis IV	Diagnosis V
Film Phase	24	24	23	13	6
Unknown	0	9	18	31	41
PACS Phase	27	22	19	13	3
Unknown	2	17	27	38	46

(‘Unknown’ implies that no disease diagnosis code was designated before radiology examination.)

At the last step of the pilot study, the diagnosis code was revised after the radiology examination was inspected

The pilot study had some influence on the medical behavior related to the adoption of PACS in this teaching hospital. Physicians’ reliance on radiology examination increased, the indeterminate disease codes decreased, and revisions of the diagnosis code increased. These results stimulated our strong motivation to carry on the analysis of the content of radiology texts from a cognitive point of view.

3 Radiology Report Content Analysis

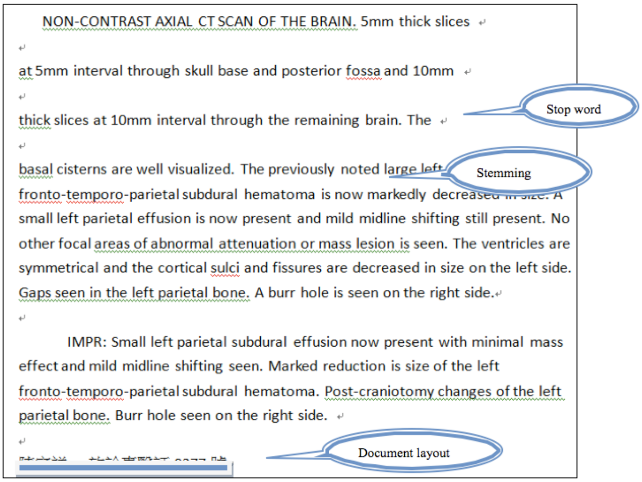
The next process is the formal concept analysis, used to construct conceptual maps for radiology reports. The lattice structures of these conceptual maps are actually a form of ontology, which reflects the characteristics of radiology reports before and after the adoption of PACS. Differences between ontologies can then be investigated by comparing their corresponding lattices according to distance-based algorithms. The research process is introduced as follows.

3.1 Term Parsing

Different radiologists prepare radiology report documents; these reports must pass through different preprocessing methods in order to fulfill subsequent requirements. Yet, the complexity and time consumption of the term parsing process are indispensable and may influence the analysis results. In order to address these concerns, Delphi is used to develop the necessary parsing tools and lexical databases for related processes.

Document layout elimination process: Disregard all irrelevant information, including typesetting format, annotation, signature, Chinese characters, and other information. The output of this phase is a data stream of characters. Figure 4 is an example of an original report text, while the result of the term parsing process is shown in Figure 5.

Stop words elimination process: The stop words referred to here are generally titles, prepositions, conjunctions, and other terms that do not constitute the main idea or concept of the document. For examples, words such as “is,” “a,” “and,” “at,” “in,” “the,” and “of” are eliminated from the report text.



NON-CONTRAST AXIAL CT SCAN OF THE BRAIN. 5mm thick slices

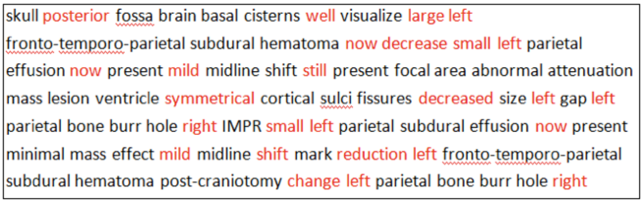
at 5mm interval through skull base and posterior fossa and 10mm

thick slices at 10mm interval through the remaining brain. The

basal cisterns are well visualized. The previously noted large left fronto-temporo-parietal subdural hematoma is now markedly decreased in size. A small left parietal effusion is now present and mild midline shifting still present. No other focal areas of abnormal attenuation or mass lesion is seen. The ventricles are symmetrical and the cortical sulci and fissures are decreased in size on the left side. Gaps seen in the left parietal bone. A burr hole is seen on the right side.

IMPR: Small left parietal subdural effusion now present with minimal mass effect and mild midline shifting seen. Marked reduction is size of the left fronto-temporo-parietal subdural hematoma. Post-craniotomy changes of the left parietal bone. Burr hole seen on the right side.

Fig. 2. Report text of abdomen CT scan



skull posterior fossa brain basal cisterns well visualize large left fronto-temporo-parietal subdural hematoma now decrease small left parietal effusion now present mild midline shift still present focal area abnormal attenuation mass lesion ventricle symmetrical cortical sulci fissures decreased size left gap left parietal bone burr hole right IMPR small left parietal subdural effusion now present minimal mass effect mild midline shift mark reduction left fronto-temporo-parietal subdural hematoma post-craniotomy change left parietal bone burr hole right

Fig. 3. Term parse result

Stemming words elimination process: Different radiologists use different writing styles. Hence, there is inevitably a chance that slight variations in the context will occur, depending on the ways in which a particular term is used. Plurality, verbal nouns, and tenses can alter the basic form of a word. The standard form of the word or root word is used to replace its different forms. For example, the word “appear” has variations that include “appearing,” “appearance,” “appearances,” etc. The principle of eliminating the stemming word is using the root word to replace all other variations of the same word.

Lexical analysis: Lexical analysis in the process of converting a sequence of characters into a sequence of strings. A string is a categorized block of text, usually consisting of indivisible terms. This is the last but most important step in the term parsing process, because lexical analysis will decide what words or terms will enter the follow-up analysis.

3.2 Conceptual Relationships and Hierarchy Construction

This study uses FCA to establish a set of conceptual relationships and a hierarchy of terms. Three kinds of relationships exist among concepts: independence, intersection, and inheritance. In order to establish the concept relationships and hierarchy of the different terms, the following steps are designed.

First, produce a binary relationship matrix between reports and terms. In every radiology report, a term that will best represent the main concept of the report must be obtained in the term retrieval. This can be done by referring to the document set and the term set. If a term appears in a document, the corresponding entries of the matrix are labeled “X” and generate the binary relationship matrix between reports and terms. The starting point in FCA is the setup of a context.

$$\text{A context is a triple } X = (D, T, I) \tag{1}$$

In this study, the context of the ontology is identified as *L*, the related report set of the ontology is represented by *D*, the related terms set of the ontology is marked as *T*, and *I* is a binary relationship between *D* and *T*:

$$I \subseteq D \times T \tag{2}$$

Secondly, one must generate the concept set *C*.
Let *X* be the partial set of *D*, and *Y* as the partial set of *T*, that is,

$$X \subseteq D, Y \subseteq T \tag{3}$$

The mappings:

$$\sigma(X) = \{t \in T \mid \forall d \in X : (t, d) \in I\} \tag{4}$$

the common terms of *X*, and

$$\tau(Y) = \{d \in D \mid \forall t \in Y : (t, d) \in I\} \tag{5}$$

the common reports of *Y*. On the basis of the above definitions, a concept is defined. A concept is a pair of sets: a set of reports and a set of terms (X, Y):

$$Y = \sigma(X) \text{ and } X = \tau(Y) \tag{6}$$

A concept is a maximal collection of documents sharing common terms. Thus, taking concept *c* as an example, the biggest report set that contains the common terms is in the maximal rectangle by all the relationships *I* in the binary relation matrix. The set of all the concepts of *c* is represented by *C*.

The third step is to calculate the hierarchical relationship of concepts. The set of all of the concepts of a given context forms a complete partial order. Thus, we define that a concept *(X_0, Y_0)* is a subconcept of concept *(X_1, Y_1)*, denoted by

$$(X_0, Y_0) \in (X_1, Y_1) \tag{7}$$

In the event that the document set X1 of a term set Y1 is contained in the document set X2 of another term set Y2, denoted by X1⊆X2, (X1, Y1) becomes the subconcept of (X2, Y_2), denoted by $(X_1, Y_1) \subseteq (X_2, Y_2)$. For concept C, it means $C_1(X_1, Y1)$ becomes the subconcept of $C_2(X_2, Y_2)$.

The last step in the conceptual construction is generating the entire hierarchy of concepts. It is possible for concept *c* to have various father concepts as well as subconcepts. Hence, the computation of various hierarchy relationships for different concepts is required in order to obtain the entire hierarchy of concepts. Each node in the hierarchy represents a concept. Given two elements *(D$_1$, T$_1$)* and *(D$_2$, T$_2$)* in the concept hierarchy, their supremum or join is defined as:

$$(D_1,T_1)\cup(D_2,T_2)=(\tau(T_1\cap T_2),T_1\cap T_2) \tag{8}$$

Let *c$_1$(X$_1$, Y$_2$)* and *c$_2$(X$_2$, Y$_2$)* be two concepts, the supreme of the two concepts is computed in order to determine their respective positions in the concept hierarchy.

3.3 Many-Valued Contexts

In Figure 3, the perception term, such as "well," "small," "now," "right," and "have" do have values. We call these perception terms "many-valued attributes," in contrast to "one-valued attributes."

A many-valued context (*G, M, W, I*) consists of sets *G, M* and *W* and a ternary relation I between *G, M* and *W* [22].

$$(g,m,w)\in I \text{ and } (g,m,v)\in I \text{ always imply } w=v \tag{9}$$

The elements of G are called objects, those of *M* (many-valued) attributes and those of *W* attribute values.

(*G, M, W, I*) is called an n-valued context, if *W* has *n* elements. The many-valued attributes can be regarded as partial maps from G in W. Thus, one must write *m(g)* = *w* instead of $(g, m, w) \in I$. The domain of an attribute *m* is defined as:

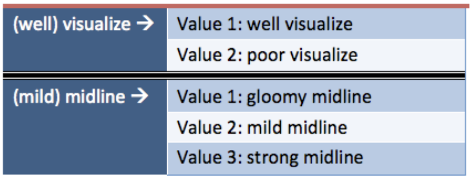
$$dom(m) := \{g\in G \mid (g,m,w)\in I \text{ for some } w\in W\} \tag{10}$$

This study transforms a many-valued context into a one-valued context in accordance with certain rules, which will be explained below. This interpretation process is called conceptual scaling.

In the process of scaling, each attribute of a many-valued context is first interpreted by means of a conceptual scale. A scale for the attribute *m* of a many-valued context is a (one-valued) context $Sm := (Gm, Mm, \mathrm{Im})$ with $m(G)\subseteq Gm$. The objects of scale are called scale values; the attributes, scale attributes.

In actuality, the choice of scale for the attribute *m* is not a mathematical process, but a matter of interpretation, and will thus be based on the opinion of the domain expert. In a lexical process, terms with perception status will be treated as a many-valued context and transformed into a one-valued context. We explain the transformation rule with the example that follows.

In Figure 3, two perception terms are observed: "well visualize" and "mile midline." Searching for the report text and discussing it with the domain expert define scales of many-valued attributes for these perception terms. One example is shown in Figure 4.



(well) visualize →	Value 1: well visualize
	Value 2: poor visualize
(mild) midline →	Value 1: gloomy midline
	Value 2: mild midline
	Value 3: strong midline

Fig. 4. Transformation of many-valued contexts

3.4 Interrelationship and Degree of Concepts

After constructing the hierarchy of relationships among concepts, the next step is to identify the interrelationships of these concepts. Let $c_1(X_1, Y_1)$ and $c_2(X_2, Y_2)$ be two concepts; if $Y_1 \subset Y_2$ and $Y2 \subset Y1$, since the two concepts are partially contained by one another, it allows us to identify the interrelationship between c_1 and c_2.

It is obvious that a repeated concept will be more important than a singly appearing concept in a single report. We can express this characteristic by using the concept of TF-IDF (term frequency-inverse document frequency) [35]. The formula and related variables are as follows:

$$r(C_j, C_k) = \frac{\sum_{i=1}^{n} tfi_{ijk}}{\sum_{i=1}^{n} tfi_{ij}} \times WF(C_k) \tag{11}$$

$$tfi_{ijk} = tf_{ijk} \times \log_{10}\left(\frac{N}{df_{jk}} \times w_j\right) \tag{12}$$

N is the total number of keywords, $\mathrm{tf}_{\mathrm{ijk}}$ is the co-occurrence of term j, k in document I, w is weight for inverse document frequency, and $\mathrm{df}_{\mathrm{jk}}$ is document frequency of term j, k.

$$tfi_{ij} = tf_{ij} \times \log_{10}\left(\frac{N}{df_j} \times w_j\right) \tag{13}$$

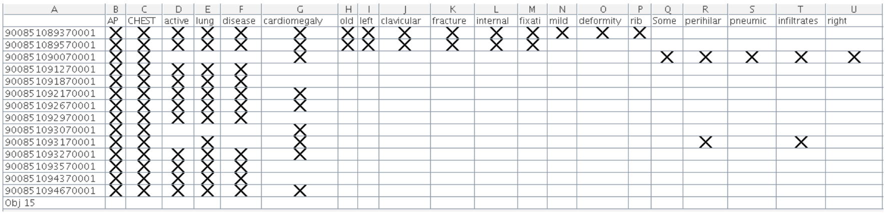
$\mathrm{tf}_{\mathrm{ij}}$ is frequency of term j in document i, and df_{j} is document frequency of term j.

$$WF(C_k) = \frac{\log_{10}\frac{N}{df_k}}{\log_{10} N} \tag{14}$$

Formula *(11)* describes the degree of relevancy between the two concepts. All degrees of relevancy have a corresponding direction. In formula *(11)*, the central point of the calculation is C_j as the correlation between C_k and C_j is being established. In formula *(12)*, d_{ijk} is decided by the frequency that C_k and C_j both appear and inverse the frequency of report. $\mathrm{tf}_{\mathrm{ijk}}$ represents the frequency that C_j and C_k both appear in document *i*, df_{jk} represents the total document number that C_j and C_k appear together. When both concepts have a higher relevance, the frequency with which C_k and C_j appear in the same report should also be high and is centralized in some specific reports. In formula *(14)*, *WF*(C_k) corresponds to the specificity of C_k against the reports. As the concept C_k becomes more general, the value of *WF* (C_k) decreases.

3.5 Ontological Concept Lattice Building

After the process consisting of the above methods, a matrix table denoting the frequency of terms that appear in radiology reports can be completed, as in Figure 5, in which the conceptual hierarchies generated by the FCA constitute the ontology lattices. Figure 6 is an example result of ontology lattices that comes from the matrix table of Figure 5.



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
	AP	CHEST	active	lung	disease	cardiomegaly	old	left	clavicular	fracture	internal	fixati	mild	deformity	rib	Some	perihilar	pneumic	infiltrates	right
900851089370001	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X					
900851089570001	X	X	X	X	X	X	X	X	X	X	X	X								
900851090070001	X	X				X										X	X	X	X	X
900851091270001	X	X	X	X	X															
900851091870001	X	X	X	X	X															
900851092170001	X	X	X	X	X	X														
900851092670001	X	X	X	X	X	X														
900851092970001	X	X	X	X	X															
900851093070001	X	X				X														
900851093170001	X	X		X		X											X		X	
900851093270001	X	X	X	X	X	X														
900851093570001	X	X	X	X	X															
900851094370001	X	X	X	X	X															
900851094670001	X	X	X	X	X	X														
Obj 15																				

Fig. 5. Formal context produced from lexical attributes of a radiology report

The upper part of the concept lattice of Figure 6 expresses the most common concept to appear in different reports, while in the lower part the change will be more sensitive in different reports. The ontology concept lattice of radiology reports will be constructed according to different conditions, physicians, phases, and examination items for further comparison analysis.

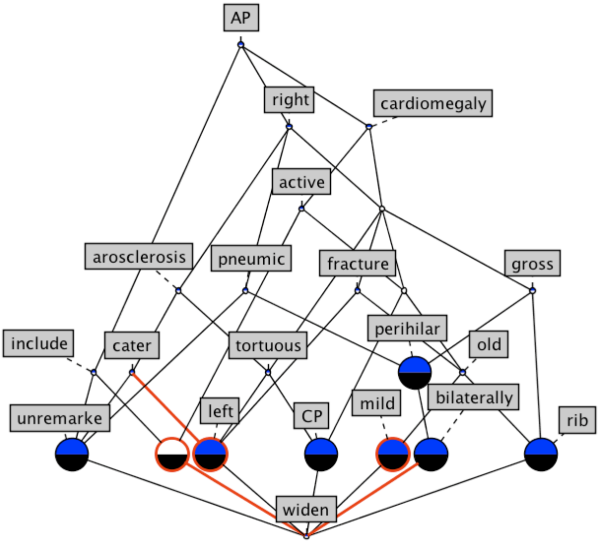


Fig. 6. Ontological concept lattice generated from formal context of Figure 5

4 Radiology Report Difference Evaluation

In order to evaluate our approach, we need to access the way in which differences exist in different phases of a radiology report as reflected by its ontology. There are many ways to measure the difference between two given objects. In this study, all concepts were normalized by the FCA method. We can treat any different concept as a different node in the ontology lattice, meaning that the ontology similarity will become a weighted graphic matching problem.

Given two ontology O_j and O_k, the formula to measure the similarities can be defined as follows:

$$Comp(O_j, O_k) = \sum_{i=1}^{N} (Con(C_{ij}, C_{ik}) \times (RC_j + RC_k)) \tag{15}$$

N is the concept number of ontology with fewer concepts, RC_j and RC_k are degree of relevancy and $Con(C_{ij},C_{ik})$ defined as:

$$Con(C_j, C_k) = \sum_{i=1}^{n} (Rt_{ij} - Rt_{ik}) \tag{16}$$

If we compare different ontology lattices in different conditions, we can find the ontology with the most obvious difference. Figures 7 and 8 are the preliminary results of a chest examination performed by a radiologist before and after PACS implementation.

In a traditional concept lattice, the concept lattice generated by FCA is sometimes quite complicated due to the large number of formal concepts generated. Since the formal concepts are generated mathematically, objects that have small differences in terms of attribute values are classified into distinct formal concepts. At a higher level, such objects should belong to the same concept when a domain expert interprets them.

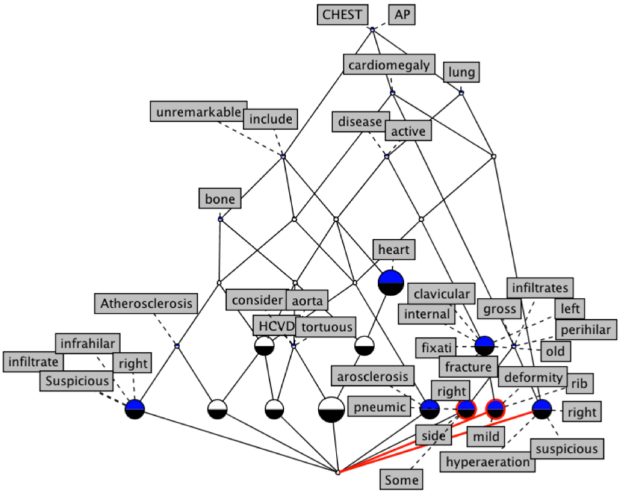


Fig. 7. Preliminary results of a chest radiology report after the adoption of PACS

On the basis of this observation, this study simplified formal concepts into conceptual clusters by using the conceptual clustering method. Each conceptual cluster is a sub-lattice extracted from the concept lattice. A formal concept must belong to at least one conceptual cluster, but it can also belong to more than one conceptual cluster.

We can use a similarity confidence threshold to determine whether two concepts are similar. Figure 9 shows the cluster result of Figure 7, which is more concise and easier to interpret. Outliers can also be detected in a cluster analysis.

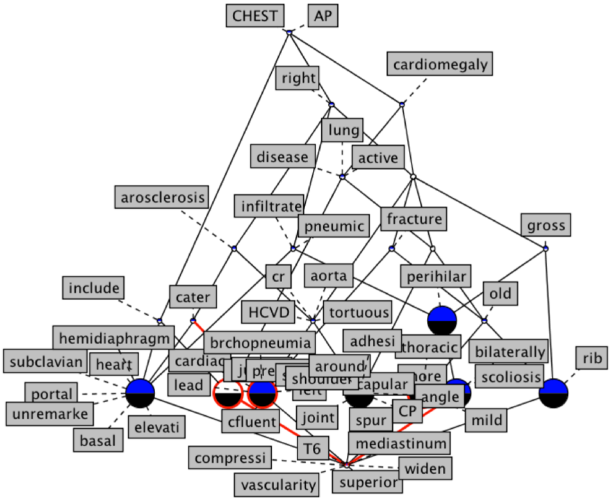


Fig. 8. Preliminary results of a chest radiology report before the adoption of PACS

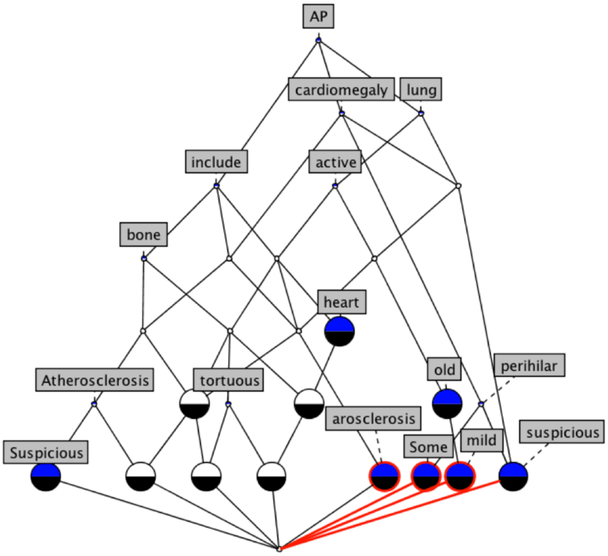


Fig. 9. A cluster result of an ontological concept lattice

5 Discussion and Conclusion

This study analyzes radiology report text in different time phases through the methods of term parsing, formal concept analysis, and cluster analysis. The important variations in radiology report content in different time phases and conditions are elucidated through the comparison of ontologies. In the results shown in Figures 7 and 8, ontologies were constructed from the analysis of this study. These ontologies represent the knowledge characteristics of the content of a radiological chest examination report prepared by a radiologist before and after the adoption of the PACS system. Figure 8, before the adoption of PACS, contains more terms (labels in the lattice chart) than Figure 7, after the adoption of PACS. Nevertheless, the actual concept number in the ontology before the adoption of PACS is 23, and is fewer than after the adoption of PACS - 29.

The interpretation of the study results shows that the radiologist used more concepts in the composing of radiology report. Observing the cluster analysis result from Figure 7 that show in Figure 9, the variant concepts include "suspicious," "some," and "mild." From the level of semantics, these concepts express the fact that the radiologists have more details and uncertainty when composing radiology reports. Our findings suggest that PACS usage may have an impact on radiologist physicians in the composition of radiology reports and the addition of more detailed concepts to the content. This is consistent with the broad concept that the adoption of PACS has the potential to improve the quality of medical images and enhance diagnoses.

The limitation of this study is that our result depends upon a single examination item and a single radiologist. Ontology construction and quantified comparison for different examination items and radiologists are still being conducted; variations in these concepts will be more complete in future studies. Future FCA analysis results will combine a focus on ethnography qualitative study hope to achieve a richer meaning from the point of view of clinic.

The adoption of PACS in hospitals changed the content of radiology reports. This study suggests that such a system may improve radiologists' clinical decisions and reflected in the radiology report. Future studies that construct and compare the ontology of different examination items from more radiologists are needed. Combining the results of FCA analysis with qualitative research on clinical radiologists will make the results more significant at the clinical level.

References

1. Jean, W.D., Danton, R.M., Kilburn, A.R.: An Assessment of a Computerized Reporting System (SIREP). The British Journal of Radiology 53, 421–427 (1980)
2. Gur, D., Stalder, J.S., Hardesty, L.A., Zheng, B., Sumkin, J.H., Chough, D.M., Shindel, B.E., Rockette, H.E.: Computer-aided Detection Performance in Mammographic Examination of Masses: Assessment. Radiology 233, 418–423
3. Choplin, R.H., Boehme, J.M., Cowan, R.J., Gelfand, D.W., Maynard, C.D., Pack, W.C., Volberg, F.M., Williams, R.C.: A Computer-assisted Radiologic Reporting System. Radiology 150, 345–348 (1984)

4. Jost, R.G., Trachtman, J., Hill, R.L., Smith, B.A., Evens, R.G.: A Computer System for Transcribing Radiology Reports. Radiology 136, 63–66 (1980)
5. Razavi, M., Sayre, J.W., Taira, R.K., Simons, M., Huang, H.K., Chuang, K.S., Rahbar, G., Kangarloo, H.: Receiver-operating-characteristic Study of Chest Radiographs in Children: Digital Hard-copy Film V.S. 2K X 2K Soft-copy Images. American Journal of Roentgenology 158, 443–448 (1992)
6. Balassy, P.M., Weber, M., Sailer, J., Herold, C., Schaefer-Prokop, C.: Flat-Panel Display (LCD) Versus High-Resolution Gray-Scale Display (CRT) for Chest Radiography: An Observer Preference Study. American Journal of Roentgenology 184, 752–756 (2005)
7. Navigli, R., Velardi, P., Gangemi, A.: Ontology Learning and Its Application to Automated Terminology Translation. IEEE Intelligent System 18, 22–31 (2003)
8. Lee, W.J., Lee, H.T., Ching, Y.T., Tsai, C.H., Liu, H.M., Chen, S.J.: Tablet PC as PACS Workstation: Observer Performance Evaluaton. Chin J. Radiol. 30, 277–281 (2005)
9. Pavlicek, W., Owen, J.M., Peter, M.B.: Active Matrix Liquid Crystal Displays for Clinical Imaging: Comparison with Cathode Ray Tube Displays. Journal of Digital Imaging 13, 155–161 (2000)
10. Frate, C.D., Bestagno, A., Londero, V., Mucelli, R.P., Salomoni, V., Bazzocchi, M.: Comparision Between CRT and LCD Displays for Full-Field-Digital-Mammography (FFDM) Interpretation. In: Astley, S.M., Brady, M., Rose, C., Zwiggelaar, R. (eds.) IWDM 2006. LNCS, vol. 4046, pp. 576–584. Springer, Heidelberg (2006)
11. Hitomi, M., Takizawa, M., Uchida, K., Osanai, K., Wada, T., Shiojima, M., Yamamoto, A., Kasahara, E., Yasuda, E.: Performance of a Plat-Panel Display System for Proximal Caries Detection. Oral Radiology 16, 67–72 (2006)
12. Parker-Pope, T.: Mamograms, New and Old. The New York Times (2008)
13. Hundt, W., Adelhard, K., Hundt, C., Nissen-Meyer, S., Kohz, P., Fink, U., Reiser, M.: A Computer-Based Reporting System in Radiology of the Chest. European Radiology 8, 1002–1008 (2004)
14. Dreyer, K.J., Kalra, M.K., Maher, M.M., Hurier, A.M., Asfaw, B.A., Schultz, T., Halpern, E.F., Thrall, J.H.: Application of Recently Developed Computer Algorithm for Automatic Classification of Unstructured Radiology Reports: Validation Study. Radiology 234, 323–329 (2005)
15. Bryan, S., Weatherburn, G., Watkins, J., Roddie, M., Keen, J., Murise, N., Buxton, M.: Radiology Report Times: Impact of Picture Archiving and Communication Systems. AJR 170, 1153–1159 (1998)
16. Franken, E.A., Smith, W.L., Berbaum, K.S., Kao, S.C.S., Sato, Y.: Comparision of a PACS Workstation with Conventional Film for Interpretation of Neonatal Examination: A Paired Comparision Study. Pediatric Radiology 21, 336–340 (2005)
17. Clinger, N.J., Hunter, T.B., Hillman, B.J.: Radiology Reporting: Attitudes of Referring Physicians. Radiology. 169, 825–826
18. Wilcox, et al.: Picture Archiving and Communication System and its Impact on Image Viewing in Physical Therapy Practice. Journal of Digital Imaging 19, 346–350 (2006)
19. Kahn, et al.: An Ontology for PACS Integration. Journal of Digital Imaging 19, 316–327 (2006)
20. Ganter, R., Wille, B.: Formal Concept Analysis – Mathematical Foundations. Springer, Heidelberg (1999)
21. Genesereth, M.R., Nilsson, N.J.: Logical Foundations of Artifical Intelligence. Morgan Kaufman Publishers, San Mateio (1987)
22. Ding, Y., Fensel, D., Klein, M., Omelayenko, B.: The Semantic Web: yet another hip? Data and Knowledge Engineering 41, 205–227 (2002)

23. Jiang, G., Ogasawara, K., Endoh, A., Sakurai, T.: Context-Based Ontology Building Support in Clinical Domains using Formal Concept Analysis. International Journal of Medical Informatics 71, 71–81 (2003)
24. Stevens, R., Goble, C.A., Bechofer, S.: Ontology-based Knowledge Representation for Bioinformatics. Briefings in Bioinformatics 1, 398–414 (2000)
25. Broekstra, J., Klein, M., Decker, S., Fensel, D., Harmelen, F., Horrocks, I.: Enabling Knowledge Representation on Web by Extending RDF Schema. Computer Networks 39, 609–634 (2002)
26. Golbreich, C., Dameron, O., Gibaud, B., Burgun, A.: Web Ontology Language Requirements w.r.t Expressiveness of Taxonomy and Axioms in Medicine. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 180–194. Springer, Heidelberg (2003)
27. Patel-Schneider, P., Horrocks, I., van Harmelen, F.: Reviewing the Design of DAML+OIL: An Ontology Language for the Semantic Web. In: Proceedings of the 18th National Conference on Artificial Intelligence (2000)
28. Lee, S.J., Kang, J.H.: Semi-Automatic Practical Ontology Construction by Using a Theasurus, Computational Dictionaries, and Large Corpora. In: Association for Computational Linguistics 2001, vol. 2001, Toulouse, France (2001)
29. Ahmad, R., AI-Sayed, K.: Knowledge Sharing in a Community of Practice: Text-Based Approach in Emergent Domains. The Electronic Journal of Knowledge Management 4, 99–108 (2006)
30. Philpot, A., Fleischman, M., Hovy, E.H.: Semi-automatic Construction of a General Purpose Ontology. In. Proceeding of the International LISP Conference, New York (2003)
31. Stamou, S., Krikos, V., Kokosis, P., Ntoulas, A., Christodoulakis, D.: Web Directory Construction Using Lexical Chains. In: Montoyo, A., Muńoz, R., Métais, E. (eds.) NLDB 2005. LNCS, vol. 3513, pp. 138–149. Springer, Heidelberg (2005)
32. Doddi, S., Marathe, A., Ravi, S.S., Torney, D.C.: Discovery of Association Rules in Medical Data. Medical Information & The Internet in Medicine 26, 25–33 (2001)
33. Hanash, C., Creighton, S.: Mining Gene Expression Database for Association Rules. Bioinformatics. 19, 79–86
34. Alani, H., Kim, S., Millard, D.E., Weal, M.J., Hall, W., Lewis, P.H., Shadbolt, N.R.: Automatic Ontology-Based Knowledge Extraction from Web Documents. In: IEEE Intelligent Systems, pp. 14–21 (2003)
35. Salton, G.: Introduction to Modern Information Retrieval. McGraw Hill, New York (1983)
36. Gruber, T.R.: Toward Principles for the Design of Ontologies Used for Knowledge Sharing. International Journal of Human-Computer Studies 43, 907–928 (1995)
37. Weng, S.S., et al.: Ontology Construction for Information Classification. Expert System with Application 31, 1–12 (2006)

Revisiting the Potentialities of a Mechanical Thesaurus

Uta Priss and L. John Old

Napier University, School of Computing
u.priss@napier.ac.uk, j.old@napier.ac.uk

Abstract. This paper revisits the lattice-based thesaurus models which Margaret Masterman used for machine translation in the 1950's and 60's. Masterman's notions are mapped onto modern, Formal Concept Analysis (FCA) terminology and three of her thesaurus algorithms are formalised with FCA methods. The impact of the historical and social situatedness of Roget's Thesaurus on such algorithms is considered. The paper concludes by discussing connections between Masterman's research and modern research.

1 Introduction

In the 1950's and 60's Margaret Masterman, a pioneer in natural language processing and AI, conducted research using a thesaurus for machine translation. Her paper "Potentialities of a Mechanical Thesaurus" describes a lattice-based model of Roget's Thesaurus (Masterman, 1956). Since lattice-based models of Roget's Thesaurus have also been described in recent Formal Concept Analysis (FCA) research (Priss & Old, 2004, 2005, 2006, 2007, 2008), the idea for this paper is to revisit Masterman's ideas, to formalise her algorithms in FCA notation and to compare her ideas to modern research.

Until recently, it was quite difficult to access Masterman's research because many of her ideas were published in technical reports which were never widely distributed. Furthermore, her writing is influenced by the 1950's intellectual background, and is sometimes difficult to comprehend from a modern perspective. In 2005, however, an edition of her work was published by Wilks (Masterman, 2005) and supplemented with editorial comments and explanations which now make it possible to revisit Masterman's ideas.

Masterman was not a mathematician. Therefore her lattice-theoretical thesaurus notions are described mostly in prose and exemplified by descriptions of computer algorithms. Because the 1950's computers were quite different from modern technology, even the algorithms are sometimes obscure from a modern perspective. An example of the changes in computational technology is demonstrated by the fact that Masterman's group (the Cambridge Language Research Unit, CLRU) believed that "no computer could hold a coded thesaurus" (Masterman, 2005, p.105) because of the storage space required.

Instead of storing a coded thesaurus on a computer, the raw and uncoded thesaurus data was stored on punched cards. It is not clear, however, how much of the data was stored and in what format. Furthermore, it is not clear how much human invention was required in order to execute any of the thesaurus-based calculations. Perhaps the computer was only used to compare thesaurus categories, but the initial processing of the

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 284–298, 2009.

input data was done by hand? Masterman (2005, p.156) mentions that the “person operating the thesaurus must use his or her own judgement”. Masterman’s early experiments were only simulations of punched card programs because “the experiments could not be carried out automatically, as the CLRU had no computer” (ibid. p.159). But the lattice program, which was coded by A. F. Parker-Rhodes, “will in the near future actually go through a computer” (ibid. p.86). Wilks (1998) explains that Masterman later “had the thesaurus card punched (twice for error checking), which then formed the basis for a range of experiments ... performed on Hollerith sorting machines”.

Another difficulty is the difference between editions of Roget’s Thesaurus. If one had an exact copy of the thesaurus used by Masterman, one could approximate her algorithms by running them on the same data and comparing the results. Masterman mentions that “Roget’s Thesaurus with additions was used” (Masterman et al., 1959) indicating that her thesaurus was amplified with further terms and edited for the purposes of her research. In some cases, the additions may have been ad hoc in response to a particular translation task (Masterman, 2005, p.153). Wilks (1998) comments that Masterman “had the whole of Roget’s Thesaurus compacted from a thousand or more heads to eight hundred”. He further states that Masterman’s experiments “were not very successful because the heads were too sparse to give sufficient repetition.” Thus, using a modern edition of Roget’s Thesaurus is a disadvantage for the purpose of understanding Masterman’s work, but it is actually an advantage with respect to examining the general validity of her work because the modern editions contain more words.

The next section[1] provides an overview of Masterman’s work of using thesauri in machine translation. Section 3 describes Masterman’s thesaurus model. Section 4 provides an FCA formalisation of three of her thesaurus algorithms. Section 5 discusses the changes between different editions of Roget’s Thesaurus. Section 6 reconstructs one of Masterman’s examples using a modern edition of the thesaurus and modern software. Section 7 traces the connections between Masterman’s work and modern research.

2 Lattices and Thesauri in Mechanical Translation

In 1956, at the International Conference on Mechanical Translation at MIT, four researchers from the CLRU (Masterman (1956), Richens (1956), Parker-Rhodes (1956), Halliday (1956)) reported on their research of using a thesaurus as an interlingua in “mechanical translation” (MT), the term then used for “machine translation”. The group’s founder, Masterman, envisioned using mathematical lattice theory for building a thesaurus, i.e. a hierarchical structure with groupings of synonyms or near synonyms. She thought that a “multilingual MT dictionary is analogous, in various respects, to a thesaurus” and that “the entries form, not trees, but algebraic lattices, with translation points at the meets of the sublattices” (Masterman, 1956). The advantage of this approach is that instead of having to consider different pairs of languages separately, each language needs to be translated only once (into the thesaurus). Adding a new language, then, does not require any changes to the previously added languages. Masterman stated that “the

[1] A draft version of Sections 2 and 7 is contained in: Priss & Old (2008). “Lattice-based Modelling of Thesauri.” Lattice-Based Modeling Workshop, Palacky University, Czech Republic.

complexity of the entries need not increase greatly with the number of languages, since translation points can, and do, fall on one another" (ibid. p.36).

Of course, computational research in the 50s and 60s was influenced by the limitations of computers at that time. Considerations about computational speed and storage problems determined the algorithms. Parker-Rhodes (1956) extended Masterman's ideas by describing a mechanical translation program for interlingual thesauri using Boolean operations that "can be performed with very great speed". The storage problem would be solved by storing "all the relevant information ... in the input and output dictionaries". Richens (1956) described the algebraic interlingua, NUDE, its code and an overview of its translation operations.

MT algorithms at that time often started with a chunk-by-chunk literal translation (Masterman et al., 1959). Every word stem and every grammatical indicator was translated from the input language to the output language using a dictionary and some rules. This was also referred to as "pidgin translation" (Masterman, 2005, p.161). Masterman's use of lattices was novel because other linguists at that time (for example Lehmann (1978)) saw translation as a mapping between trees. A sentence from the input language was parsed into a tree structure. Each branch of the input language was mapped onto a branch of the output language. The branches in the output language formed another tree which had the output sentence as their root. Masterman argued that from a semantic viewpoint, lattices are a better model than trees. In a lattice, pairs of elements can have different numbers of parents and children, instead of having only one parent each in a tree structure. Thus, combinations of meanings can be represented more naturally. In 1965, Wilks commented that word-for-word pidgin translation was too limited because phrases, not words, are the semantic units of a sentence (Masterman, 2005, p.186). But this does not contradict the idea of using a thesaurus as a means of translating the semantic units.

In particular, Masterman (1957) was interested in Roget's Thesaurus (RT). Her idea was that each of the 1000 categories in RT could be used as a "head" which described the core meaning of a word. Because words that have more than one sense occur more than once in RT, a word can have several heads. This leads naturally to a lattice, not tree structure. Of course, this implies that the meets and joins need to be calculated; without meets and joins, a thesaurus would be just a partially ordered set, not a lattice. Multiple occurrences of a word in the thesaurus might correspond to different meanings of the word or even homographs (such as "lead" the verb and "lead" the metal). If one determines the heads of all the words of a sentence, the heads provide an indication of what the sentence is about. Individual words can be disambiguated by comparing their heads to the other heads in the sentence. If a word has two different heads and only one of these also occurs for other words in the same sentence, then it is quite likely that that head corresponds to the meaning of the word in this sentence.

Masterman et al. (1959) saw a relationship between MT and information retrieval because in both cases a thesaurus could be used: either for retrieval, or as an interlingua. Even grammar and syntax were dealt with by the thesaurus (Masterman, 1957) because grammatical indicators in the "intralinguistic context" relate to structures in the "extralinguistic context" that are shared across languages. For example, some languages have no genders (English), others have two (French), three (German) or six (Icelandic).

But the distinction between "male" and "female" is extralinguistically motivated (ibid. p.39). Masterman et al. (1959) see an interlingua as consisting of a "logical system giving the structural principle on which all languages are based". In modern terminology, the thesaurus represents the "conceptual structures" that underly information retrieval and natural languages. Because different languages share conceptual structures, they could share a thesaurus as a representation of conceptual structures. Thus, Masterman's ideas anticipated modern research into semantics, AI, knowledge representation and formal ontologies. The next section shows how her thesaurus notions can be mapped to their modern equivalents.

3 Masterman's Thesaurus Model

Masterman describes a thesaurus in terms of *heads*, *fans* and *tags*. Fig. 1 summarises her notions. Heads correspond to *contexts* (Masterman, 2005, p.109) or *situations* (ibid. p.194). Different situations are distinguished from each other by *similarity* and *contrast* (idid. p.189). In Roget's Thesaurus (RT), heads correspond to the approximately 1000 categories. The notion of *fan* refers to the polysemy of a word. A fan consists of a word and its different uses or meanings which could be changing over time (ibid. p.39). In RT, the index at the back of the book (or what Masterman et al. (1959, p.925) call the *cross-reference dictionary*) shows a fan for each word. Heads are further subdivided into *lists*, which are mutually exclusive, such as "spade, hoe, rake"; and *rows* which are quasi-synonymous, but non-exclusive, such as "coward, faint-heart, poltroon" (Masterman, 2005, p.109). Extralinguistically, the elements of a row can denote the same object, whereas the elements of a list denote different objects. Thus, the elements of a list "are sub-species of a genus, and not synonyms at all" (Wilks, 1998) or might form an extension of a natural language concept. In some editions of RT, lists and rows are visually differentiated because lists are printed in a one-word-per-line format. Rows are the smallest groupings of the words in RT: each category is subdivided into paragraphs which contain semicolon-delimited sets of words.

Archeheads or *aspects* are additional classifications that crosscut the classification by the heads. Aspects are also distinguished from each other by similarity and contrast in the same manner as situations (ibid. p.189). Neither situations, nor aspects are permanently fixed, but instead both change over time and if different principles are

Masterman's notion	in Roget's Thesaurus	general notion
head (context/situation)	category	concept
similarity/contrast	(implicit)	attributes
list	list	extension
row	paragraph or semicolon group	synonymy
fan	index	polysemy
tag (aspect)	part of speech etc	facet
archeheads (aspect)	antonymy (in table of contents)	facet/frame/role

Fig. 1. Masterman's main thesaurus notions

applied (ibid. p.189). Masterman uses the metaphor of "re-sorting a pack of cards" in order to illustrate the flexibility of classifications (ibid. p.190). In older versions of RT, the main archeheads are "pleasing" and "non-pleasing" (ibid. p.110), because the categories alternate between a positive head followed by its antonymous counterpart. This is predominantly displayed in the table of contents of older versions of RT. In modern information science terminology, archeheads are *facets*. Masterman further uses the notion *(semantic) tag* to refer to aspects (ibid. p.202). In this case, this seems to include part of speech tags, which is a facet that is applied to all categories in RT. The other examples of tags given by Masterman (ibid. p.202 and p.204) appear to be similar to modern thematic roles, frames or case relations. These do not systematically occur in any of the printed editions of RT, but must have been manually constructed by Masterman's group.

In summary, Masterman's notions correspond to modern notions and closely relate to the structures that can be observed in Roget's Thesaurus. Her notion of *double classification* (ibid. p.190) using situations/heads and aspects/tags is similar to the modern notion of faceted classification. Her notions of heads, similarity/contrast and lists can be mapped onto FCA terminology. Heads, words and fans provide the basic formal contexts for an FCA modelling of thesauri as shown in the next section.

4 Three Algorithms

This section describes three of Masterman's thesaurus algorithms in FCA terminology. All three algorithms follow the same core structure and are similar to what are called *restricted neighbourhood lattices* (Priss & Old, 2004) in modern research. A (modern) neighbourhood context in Roget's Thesaurus starts with a word, looks up all the senses (rows or heads) of this word, and then looks up all the other words that share the same senses. This process can be started with either words or senses and is stopped after several iterations. The usual FCA operator for finding all attributes that belong to a set of objects is the *prime operator*[2]. The *plus operator*[3] used for constructing neighbourhood lattices differs from the prime operator in that it looks for attributes belonging to *any* (instead of *all*) of the objects. Thus, it usually enlarges the sets. This enlargement can be reduced by requiring that attributes need to be shared by at least two (or any other number) of objects[4]. Such neighbourhood lattices are called *restricted* (Priss & Old, 2004). Applied once to a single object (or single attribute), the prime and plus operators yield the same result ($g' := G_1' = G_1^+ =: g^+$ for $G_1 = \{g\}$).

The first algorithm considered here is a translation of a Latin sentence "Agricola in curvo terram dimovit aratro" into English[5] (Masterman et al. (1957) and Masterman (2005, p. 149)). It involves the following stages:

1. dictionary matching: for each chunk of the input language a set of English heads is found representing semantic, syntactic and grammatical elements

[2] For a formal context $(G, M, I), G_1 \subseteq G, M_1 \subseteq M$, $G_1' := \{m \in M \mid gIm$ for all $g \in G_1\}$. $M_1' := \{g \in G \mid gIm$ for all $m \in M_1\}$. To avoid ambiguity, G_1' can also be written as G_1^I.

[3] $G_1^+ := \{m \in M \mid gIm$ for one $g \in G_1\}$. $M_1^+ := \{g \in G \mid gIm$ for one $m \in M_1\}$.

[4] $G_1^{(I,\geq n)} := \{m \in M \mid gIm$ for at least n elements $g \in G_1\}$. Thus, $G_1^+ = G_1^{(I,\geq 1)}$.

[5] For example: "The farmer parts the earth with his curved plough."

2. operations on semantic heads: basic translation
3. operations on syntactic heads: syntactically complete, unparsed output
4. operations on grammatical heads: parsed and ordered output
5. cleaning up operations

Masterman argues that, in principle, grammar and syntax can be dealt with in the same manner (i.e., at a conceptual level using a thesaurus) as semantics (Masterman, 2005, p201). But in Masterman et al. (1957), only the application of Roget's Thesaurus to Stage 2 is shown, which involves operations on semantic heads. Translated into an FCA notation, a thesaurus contains a formal context (W, H, F) of words $w \in W$, heads $h \in H$ and fans, which describe the relationship between a word and its heads. Masterman's algorithm starts by calculating $S^F = \bigcup_{w \in S} w^F$ which contains the heads w^F for each chunk w of the input sentence $S \subseteq W$. Each head that does not occur in at least one other set of heads is eliminated: $\mathcal{H}(S) := \bigcup_{w \in S} w^{F/S}$ with $w^{F/S} := \bigcup_{v \in S, w \neq v} v^F \cap w^F$. This step eliminates homographic or polysemous senses that do not relate to the main topic of the sentence. It follows that $w^{F/S} = w^F \cap \mathcal{H}(S)$ and $\mathcal{H}(S) = S^{(F, \geq 2)}$. In modern terminology, Masterman's algorithm produces a restricted neighbourhood context $(S, \mathcal{H}(S), F \cap S \times \mathcal{H}(S))$.

If $w^F \cap v^F = \emptyset$ for all $v \in S, w \neq v$, none of the heads in w^F are in $\mathcal{H}(S)$. Thus, in order to determine a translation of w, the next higher grouping above the heads according to the table of contents of RT is considered. This can be represented as a formal context (H, A, C) of heads H, higher level classes A and the classification relationship C. Two plus operators are applied: first, the higher level classes are obtained, then all other heads that belong to these higher level classes are selected. This can be formally represented[6] as $w^{(F \circ C, \geq 1)(C, \geq 1)} = \{h \in H \mid \exists_{a \in A} \exists_{i \in H} : wFi, iCa, hCa\}$. This contains all heads that are in the same higher level classes as the original heads in w^F. Now restriction can again be applied: $w^{F/S*}$ is defined as $w^{F/S}$ if this is not $\emptyset$ and $\bigcup_{v \in S, w \neq v} v^F \cap w^{(F \circ C, \geq 1)(C, \geq 1)}$, otherwise. The result is a set $\mathcal{H}(S)^* := \bigcup_{w \in S} w^{F/S*}$ and a restricted neighbourhood context $(S, \mathcal{H}(S)^*, F \cap S \times \mathcal{H}(S)^*)$.

Each head in $\mathcal{H}(S)^*$ can now be translated into the target language. The target language is represented by a formal context (W_1, H, F_1) corresponding to the words W_1 and fans F_1 of the target language and the same heads as in the source language. Possible translations are elements of $\mathcal{H}(S)^{*F_1} \subseteq W_1$. More specifically, the translation of a word $w \in W$ with respect to S is provided by $T(w) := (w^{F/S*})^{(F_1, \geq 2)} = \{v \in W_1 \mid vF_1h$ for at least 2 elements $h \in w^{F/S*}\}$. Only those translations are selected that occur in at least two heads. The whole algorithm resembles the calculation of restricted neighbourhood lattices, with some modifications.

Some entries in Roget's Thesaurus contain cross-references. This is not to be confused with Masterman's "cross-reference dictionary" in the back of the book. Cross-references in the first half of the book are mostly a space-saving mechanism in RT: if a group of words occurs in two paragraphs, a cross-reference is used in one of the categories instead of duplicating the words in both places. In general, the use of cross-references is inconsistent in RT. Masterman argues that RT is a lattice where the joins are given by the hierarchy that is contained in the table of contents of the book,

[6] $F \circ C$ denotes the relational composition. $w^{(F \circ C, \geq 1)} = \{a \in A \mid \exists_{i \in H} : wFi, iCa\}$.

and the meets are given by the cross-references (Masterman, 2005, p.210). Thus, for the purposes of the translation algorithm, Masterman considers both words and cross-references that are shared between two heads. If a cross-reference is shared, it is added to its $w^{F/S*}$ and included in the computation of $T(w)$ as an iterative process. Masterman's reasoning for this is that "the numerical cross-references are to be interpreted ... as the overlap in meaning between the heads" (ibid. p.204).

In the Latin example sentence, this procedure leads to between one and eight possible translations for each word of the sentence (ibid. p.157). Similar results can be obtained with a modern thesaurus (see Section 6). Masterman explains that syntactic and grammatical operations would need to be applied in order to obtain the full translation, but these operations are not further explained. Incidentally, we ran the Latin sentence through a free online InterTran translation website[7] yielding the translation: "farmer upon to bow earth dimovit plowman". This is significantly worse than Masterman's head translations: "farm, farmer, bend/crook, ground/soil, plough/till, ploughman" (ibid. p.157). Unfortunately, we were not able to compare the result to Google's and Yahoo's translations (which yield good translation results in some cases) because these do not translate from Latin.

The second algorithm is very similar to the first but is applied to the translation of a whole passage of text instead of just a single sentence. It was originally described in Masterman (1956), but also appears in a slightly amended version in the appendix of Masterman et al. (1959). This algorithm consists of four stages:

1. Using a bilingual pidgin dictionary, chunks of the input language are translated into chunks of the output language.
2. Lattice position indicators are used for identifying the syntactic elements.
3. The thesaurus cross-reference dictionary is used to find the heads for the chunks.
4. The thesaurus procedure is applied.

Unfortunately, stage 2 is not explained in the paper. It is not clear how the syntax lattice is constructed. The main purpose of stage 2 appears to be to determine which heads to include in the calculation of $w^{F/S}$ (using the FCA notation from above). When dealing with a whole paragraph of text, it would not be useful to compare all heads simultaneously. Instead, only those heads are compared which are equal or subordinate to each other, based on the syntax lattice. As stated before, this is not completely clear in the paper, but it probably means that the syntax lattice is similar to a tree-parsed structure. Heads are compared if they are in a parent-child relationship in the tree. For example, the head of an adjective is compared with the head of its noun, and so on.

Otherwise, the algorithm is very similar to the previous ones, except that more procedures are added for dealing with cross-references and for adding words to the thesaurus. Most likely the reason for this is that the thesaurus used by Masterman was fairly sparse and did not yield any results for some of the head comparisons. Masterman argues that these procedures are not ad hoc, but instead make use of cross-references and of interpreting phrases and metaphors that are implicitly contained in the thesaurus. She asserts that "all possible chains of meanings are somewhere in Roget's Thesaurus if they can be

[7] www.stars21.com and www.translation-guide.com

found" (Masterman, 2005, p.93)[8]. The conclusion of Masterman's paper is that the algorithm is able to improve the initial pidgin translation by replacing some of the words with more appropriate choices.

The third algorithm (Masterman, 1961) translates an English sentence into a thesaurus representation, which serves as an interlingual or conceptual representation. Even though we argue that Masterman's algorithms resemble restricted neighbourhood lattices, she does not represent the results as lattices herself, though her 1961-paper contains a graphical representation of fans which are quite similar to our representations of neighbourhood lattices. *Jointed fans* (Masterman, 2005, p.207) are partially ordered sets that have a word at the top, and the different uses or meanings of the word below (as given in the index of Roget's Thesaurus). In the diagram, each meaning is indicated by the number and the name of the head in which this word occurs. The resulting diagrams strongly resemble our representation of neighbourhood lattices (as in Figure 2). Jointed fans are trees, not lattices, but the smallest neighbourhood lattices (consisting of a word, its senses and other words that share these senses) tend to be mostly tree-like. Non-tree like structures emerge in neighbourhood lattices when the plus operator is applied more than twice, or when the algorithm starts with more than one word or sense.

The fans are further subdivided by the different parts of speech of the word under each head. The part of speech indicators are examples of *tags*. Masterman discusses the manual addition of further tags (ibid. p.204), which assign aspects to each occurrence of each word in the thesaurus. Although the words used for the tags are idiosyncratic, the resulting structure resembles a formal ontology. Masterman (ibid. p.209) argues that this fan structure, consisting of heads and tags, corresponds to dictionary entries and that, for translation purposes, dictionaries of different languages can be connected to an interlingual thesaurus using such fan structures.

The algorithm for translating the English sentence into a thesaurus representation given by Masterman (1961) is as follows (translated into FCA notation): for the thesaurus context (W, H, F) and a sentence S with words $w \in S$, the set $S^F = \bigcup_{w \in S} w^F$ contains the sets of heads for each word. For each element in $\mathcal{H}(S)$, the translation is provided as: $T(w) := (w^F)^{(F, \geq 2)} = \{v \in W \mid vFh$ for at least 2 elements $h \in w^F\}$. This is basically the same procedure as described above (Masterman et al., 1957), again resembling neighbourhood lattices. In this case, Masterman provides special rules for the instances where $T(w)$ contains more than one word or w^F contains zero or one head, but these rules rely on other non-thesaurus resources (such as special dictionaries).

5 Changes between the Different Thesaurus Editions

If Masterman's experiments are to be replicated, it is of interest to determine how far they depend on the particular versions of the thesaurus. This section provides an

[8] We believe that her assertion is correct. We have conducted experiments with changing the degree of restriction in neighbourhood lattices (Priss & Old, 2008) and found that it is possible to use heuristics to enlarge the lattices in cases where not much overlap is found. But, of course, the implementation of such techniques would have been difficult to imagine with 1950s computers.

excursus into the differences between the thesaurus editions in order to highlight the historical changes of the thesaurus categories.

There are two main versions of Roget's Thesaurus (RT) publications: the British line, published by Longman/Penguin from 1852-2002; and the US American line, branching from the British line in 1911, published by Crowell/Harper Row, and commonly referred to as the International version (RIT). Both contain the hierarchical classification system, the Synopsis of Categories, based on Roget's 1000 categories, with six to eight classes at the highest level. By default, the US thesaurus uses American spellings for common words such as color and jail, while the British version lists colour and gaol in the index. It is notable that the current International version does not list gaol, even as a Britishism, while the British version does list jail. This could reflect the influence of American spelling on British English. Masterman used RT (1953) for her experiments. Our own experiments are based on an amended version of RIT (1962)[9]. Because of copyright reasons, RT (1911) is the most current edition in the public domain.

With each new edition of RT, categories may be renamed, added, and sometimes combined or split (very, very rarely deleted). Among the reasons to change the name of a category, modernisation is the most usual. For example, Cicuration (RT, 1852-1953; RIT 1911-1922), meaning the act of taming animals. This was modernised to Animal Culture (RIT, 1946), then Animal Husbandry (RIT, 1962; RT, 1972). A further example is Preterition (RT 1852-1972; RIT 1911-1946), meaning passing, or passed. This was modernised to The Past (RIT, 1962) and Past Time (RT, 1972). Categories may be split where distinct ideas were obviously combined under one head (John L. Roget, 1933, Editors Preface); or expanded, for example, from "Earliness" (RIT, 1922) to "Earliness; Punctuality" (RIT, 1946).

Both versions of RT have in the past borrowed entries from the other, and occasionally, category names: Non-addition; Subduction (RT, 1852-1953; RIT, 1911-1922), Deduction (RIT,1946); Subduction (RT, 1962), Subtraction (RT, 1982; RIT 1992). Or perhaps these changes simply reflect the changing face of modern English (a term which is always current, no matter in which decade it is used).

The pattern of category changes is different from the British version to the American version. For example the updating of equivalent categories occurs earlier for the American editions. Also, the British version is more tolerant of obsolete (from daily usage) Latin terms. In recent years the addition of new words has mainly been in the areas of science and technology. The addition of these terms also differs between the versions. The British version continues to add new scientific and technical terms to existing Categories, while the American version has added new categories. For example, electronics is found under 160 Power in the British version (RT, 2002), along with strength, force and energy; while the American version has its own category for 342 Electronics (RIT, 1962).

The addition of categories can reflect cultural and political attitudes, just as Roget's original categories reflected the attitudes of his day. For example, category 986 Pseudo-revelation (original 1852 edition) contrasts the heathen Koran, Buddha, the

[9] Which was converted to electronic form by Sedelow & Sedelow (1993) under NSF grants and later converted to a relational format by the second author.

Upanishads, and others, with orthodox Judeo-Christian beliefs of the time (category 985 Revelation).

The 1953 British edition was chosen by Masterman and the CLRU at a time when the paranoia of Macarthyism was rampant in the USA. Not coincidentally, much of their machine translation research was supported by the US military (National Science Foundation, U.S. Air Force Office of Scientific Research, and the Office of Naval Research (Washington, D.C), among others.

This had no apparent direct effect on RT, but had a massive influence on the ensuing US edition (RIT, 1962). That edition was used directly in the machine translation efforts of the US military to translate Russian military strategy. The following categories were some of those which were either added, or radically expanded: 277 Aeronautics, 280 Rockets and Flying Missiles, 281 Astronautics, 326 Radiation and Radioactivity, 342 Electronics, 345 Radar and Radiolocators, and 348 Automation. A sample of the type of words added is:

> 277 Aeronautics: aircraft hydraulics, jet engineering, kinetics, micrometry, rocket engineering, supersonics, supersonic aerodynamics; aviation medicine, Air Force School of Aviation Medicine, Air Force Department of Space Medicine; aerial navigation, celestial navigation, electronic navigation, automatic electronic navigation, navar, teleran, loran, shoran
>
> 348 Automation: robotic control, cybernetics, automatic electronic navigation, automatic guidance, missile guidance; guided missile, thinking machine, chess-playing machine; ENIAC UNIVAC, IBM 702

Some of the more cryptic words are acronyms, probably completely unknown to normal English speakers. For example, loran (Long Range Aid to Navigation) and teleran (Television Air Radar Navigation).

As this was also the beginning of the space race, many related words were added to categories such as 374 Universe (to do with astronomy, star systems, constellations, along with some navigational terminology). 279 Aircraft, in the 1946 (post-war) edition, had extensive lists of Second World War US, British, German, Italian and Japanese military aircraft. In 1962 these were replaced by terms such as “air-sea rescue amphibian, anti-submarine patrol; constant-chordrotor helicopter, intermeshing-rotor helicopter; high-altitude reconnaissance plane, long-range patrol bomber, photo-reconnaissance plane”.

This excursus demonstrates the historical and social situatedness of thesaurus versions. Any thesaurus algorithm will be influenced to some degree by these versional changes. The actual labelling of the heads is not so important for neighbourhood lattices, but the types and numbers of words in the thesaurus and the degree of granularity in the lowest level groupings are relevant. Because the modern versions are copyrighted and not freely available in electronic format, it is difficult to conduct experiments that evaluate the impact of the versional changes. Our experiments have show that RT (1911) is too sparse compared to RIT (1962). Masterman’s RT (1953) is similarly sparse as RT (1911).

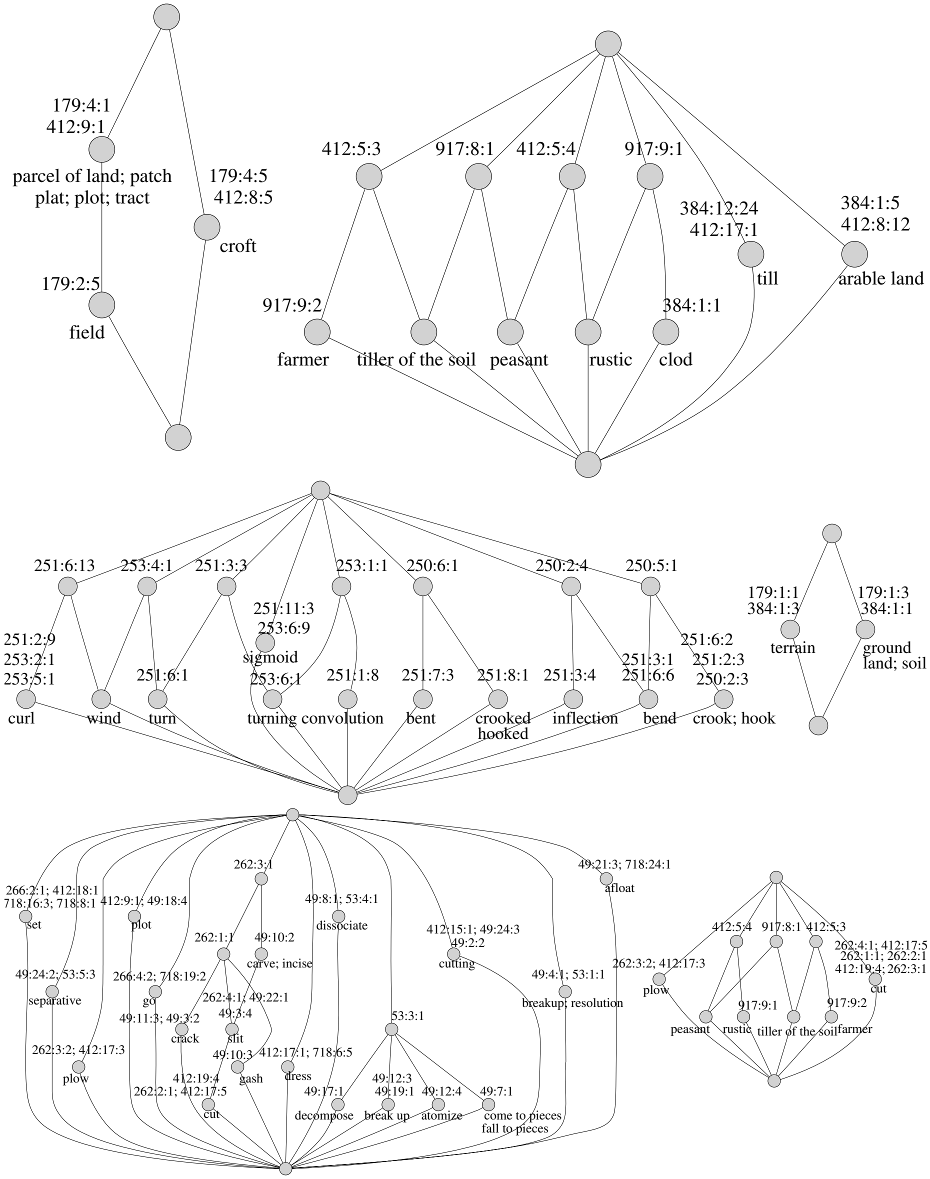


Fig. 2. Neighbourhood lattices for the translation of “Agricola in curvo terram dimovit aratro”

6 Revisiting one of Masterman’s Examples

In this section, we revisit Masterman’s treatment of the Latin sentence “Agricola in curvo terram dimovit aratro” as described in Section 4. We use Masterman’s heads ($\mathcal{H}(S)^*$), but manually map them to the categories of RIT (1962) instead of RT (1953).

In most cases, the category names are the same in both thesauri, only the numbers are different. For example, Agriculture is category 412 in RIT (1962) and category 371 in RT (1953). We then use our software as described in previous papers (Priss & Old 2004, 2007 and 2008) to calculate neighbourhood lattices for each $T(w)$. The six lattices in Figure 2 correspond to the translations of the six chunks "AGRI", "COL", "INCURV", "TERR", "DIMOV" and "AR". Most of the objects in these lattices are the words that Masterman retrieved as well. RIT (1962) retrieves slightly more words, than RT (1953).

This example confirms that even with the differences between thesaurus editions, the main structures are still similar. Because we started with Masterman's heads ($\mathcal{H}(S)^*$), this example only confirms the last part of Masterman's algorithm. With respect to the first part (the initial selection of heads), one would need a thesaurus of Latin words. Masterman conducted this part manually. The second part (the use of higher level classes and cross-references) is possible with RIT (1962), but so far we have not yet implemented it. Another confirmation of Masterman's research is the fact that her algorithms are similar to techniques that were independently developed by other researchers as discussed in the next section.

7 Modern Descendants

Masterman's research influenced many people, including Karen Spärck Jones who is considered to be one of the pioneers in information retrieval and natural language processing. Spärck Jones used Roget's Thesaurus, but as far as we know had not much interest in lattices. Similarly, Yarowsky (1992) described an implementation of the use of Roget's for word-sense disambiguation which was very similar to Masterman's ideas (although he does not cite her), but he uses statistical methods instead of lattices.

In 1960s in the US, Sally Yeates Sedelow obtained funding to convert the American edition of Roget's (1962) into a machine readable format with the purpose of aiding machine translation. The initial abstract models that she and her husband, Walter Sedelow Jr., used did not rely on lattice theory (Dillon (1971), Bryan (1973), Bryan (1974), Talburt & Mooney (1989)). But Bryan's model describes a binary relation between words and senses which is very similar to a formal context as used in FCA. Thus, when the Sedelows met Rudolf Wille, the founder of FCA, in the early 1990s, they were enthusiastic about the possibilities that lattice theory had to offer for their research. Their paper about the concept "concept" (Sedelow & Sedelow, 1993) derives semantic neighbourhoods for words from the thesaurus which are then represented as "neighbourhood lattices". Our own research has used and elaborated this technique in a variety of papers (Priss & Old, 2004) and has recently led to the implementation of an on-line interface[10] that allows users to interactively generate such lattices. As explained above, Masterman's thesaurus algorithms are a form of "restricted neighbourhood lattices". Thus, one can argue that this modern research is an implementation of Masterman's ideas, although the thesaurus research (of the Sedelows) was initially separated from lattice research and was only recombined through FCA.

Another modern instantiation of Masterman's ideas is Helge Dyvik's (2004) research, although, as far as we know, he was not directly influenced by, or aware of,

[10] http://www.roget.org

either FCA or Masterman. Dyvik's lattices are feature lattices in the sense of componential semantics. Dyvik's Semantic Mirrors Method extracts semantic information from bilingual corpora. His assumption is that if the same sentence is expressed in two different languages, then it should be possible to align words or phrases in one language with the corresponding words or phrases in the other language using statistical processes or semi-automated processes. Once the corpora are aligned the "translational images" of words in the other language are computed. This process can be repeated several times. Next, the translational images are algorithmically assigned to separate senses. The resulting structures can be represented either graphically as lattices, or as a thesaurus (using a WordNet-style representation). Both structures can be generated interactively through an on-line interface[11]. Priss & Old (2005) have shown that this procedure is similar to creating neighbourhood lattices in FCA, though Dyvik's research was developed independently of FCA.

It could be argued that Dyvik's Semantic Mirrors method is a proof of concept for Masterman's vision. Masterman's (1956) statement that a "multilingual MT dictionary is analogous in various respects, to a thesaurus" and that "the entries form, not trees, but algebraic lattices, with translation points at the meets of the sublattices" prescribes exactly what Dyvik has implemented. Of course, it would not have been possible to implement a system like Dyvik's in the 1950s or 60s due to the limits of computers at that time. It seems to us, however, that perhaps not all of Masterman's ideas have fully been explored using modern technology. For example, the "Twenty questions method of analysis" (Masterman et al., 1959) that was used for extracting extralinguistic (or "semantic") information via an intralingual analysis, appears to be similar to attribute exploration in FCA (Ganter & Wille, 1999). But this relationship has not yet been further investigated.

8 Conclusion

In summary, Masterman's theoretical notions can easily be mapped to modern FCA notions. Her lattice-thesaurus research has been independently rediscovered (Dyvik (2004) and Sedelow & Sedelow (1993)) and has been implemented in recent FCA research (Priss & Old, 2004, 2005, 2006, 2007, 2008). The core of the algorithms and graphical representations described by Masterman resembles restricted neighbourhood lattices, which have been shown to be the most useful lattices for Roget's Thesaurus (Priss & Old, 2008). Several of her ideas are novel with respect to neighbourhood lattices of Roget's Thesaurus and could inspire future research:

- The use of cross-references. In library thesauri, "related term" links are similar to cross-references in Roget's Thesaurus. Thus, Masterman's approach for using cross-references is relevant for such thesauri.
- The use of an additional aspect classification (or faceted classification). Masterman suggests to modify the hierarchy of heads that is represented in the table of contents of Roget's Thesaurus so that it becomes a lattice structure in order to obtain a more detailed semantic representation. This can be achieved by adding semantic tags, and would be quite easy to implement for modern thesauri.

[11] http://ling.uib.no/helge/mirrwebguide.html

- Semantic tags for the individual occurrences of the words in the thesaurus. Masterman explains this procedure with a manually constructed example for one head. Clearly, it would be difficult to implement this for the whole thesaurus. But perhaps there are modern natural language processing algorithms that could be used for such purposes. At least the tags that already occur in the thesaurus (mostly part of speech tags) could be used in the process of constructing neighbourhood lattices.

References

1. Bryan, R.: Abstract Thesauri and Graph Theory Applications to Thesaurus Research. In: Sedelow, S.Y. (ed.) Automated Language Analysis, Report on research 1972–73. University of Kansas, Lawrence (1973)
2. Bryan, R.: Modeling in Thesaurus Research. In: Sedelow, S.Y. (ed.) Automated Language Analysis, Report on research 1973-74. University of Kansas, Lawrence (1974)
3. Dillon, M., Wagner, D.J.: Models of thesauri and their applications. In: Sedelow, S.Y. (ed.). Automated Analysis of Language Style and Structure in Technical and other Documents. Technical Report, University of Kansas, Lawrence (1971)
4. Dyvik, H.: Translations as semantic mirrors: from parallel corpus to wordnet. Language and Computers 49(1), 311–326 (2004)
5. Ganter, B., Wille, R.: Formal Concept Analysis. Mathematical Foundations. Springer, Heidelberg (1999)
6. Lehmann, W.P., Pflueger, S.M., Hewitt, H.-J.J., Amsler, R.A., Smith, H.R.: Linguistic Documentation of Metal System. Final technical report, Rome Air Development Center, RADC-TR-78-100 (1978)
7. Masterman, M.: Potentialities of a Mechanical Thesaurus. MIT Conference on Mechanical Translation, CLRU Typescript [Abstract]. In: Report on research: Cambridge Language Research Unit. Mechanical Translation, vol, 3(2), p. 36 (1956); Full paper in: Masterman (2005)
8. Masterman, M.: The Thesaurus in Syntax and Semantics. Mechanical Translation 4(1,2), 35–43 (1957)
9. Masterman, M., Needham, R.M., Spärck-Jones, K., Mayoh, B.: Agricola in curvo terram dimovit aratro. In: Masterman. CLRU memo ML-84 (1957) (2005)
10. Masterman, M., Needham, R.M., Karen, S.-J.: The Analogy between Mechanical Translation and Library Retrieval. In: Proceedings of the International Conference on Scientific Information (1958), vol. 2, pp. 917–935. National Academy of Sciences - National Research Council, Washington, D.C (1959)
11. Masterman, M.: Translation. Aristotelian Society Supplementary, vol. 35, pp. 169–216 (1961); In: Masterman (2005)
12. Masterman, M.: Language, Cohesion and Form. Edited by Yorick Wilks. Cambridge University Press, Cambridge (2005)
13. Parker-Rhodes, A.F.: Mechanical Translation Program Utilizing an Interlingual Thesaurus [Abstract]. In: Report on research: Cambridge Language Research Unit. Mechanical Translation, vol. 3(2), p. 36 (1956)
14. Priss, U., Old, L.J.: Modelling Lexical Databases with Formal Concept Analysis. Journal of Universal Computer Science 10(8), 967–984 (2004)
15. Priss, U., Old, L.J.: Conceptual Exploration of Semantic Mirrors. In: Ganter, B., Godin, R. (eds.) ICFCA 2005. LNCS, vol. 3403, pp. 21–32. Springer, Heidelberg (2005)
16. Priss, U., Old, L.J.: An application of relation algebra to lexical databases. In: Schärfe, H., Hitzler, P., Øhrstrøm, P. (eds.) ICCS 2006. LNCS (LNAI), vol. 4068, pp. 388–400. Springer, Heidelberg (2006)

17. Priss, U., Old, L.J.: Bilingual Word Association Networks. In: Priss, U., Polovina, S., Hill, R. (eds.) ICCS 2007. LNCS (LNAI), vol. 4604, pp. 310–320. Springer, Heidelberg (2007)
18. Priss, U., Old, L.J.: Data Weeding Techniques Applied to Roget's Thesaurus. Knowledge Processing in Practice (in preparation) (2008)
19. Roget, P.M.: Roget's International Thesaurus, 3rd edn. Thomas Crowell, New York (1962)
20. Sedelow, S., Sedelow, W.: The Concept concept. In: Proceedings of the Fifth International Conference on Computing and Information, Sudbury, Ontario, Canada, pp. 339–343 (1993)
21. Talburt, J.R., Mooney, D.M.: The Decomposition of Roget's International Thesaurus into Type-10 Semantically Strong Components. In: Proceedings of 1989 ACM South Regional Conference, Tulsa, Oklahoma, pp. 78–83 (1989)
22. Wilks, Y.: Language processing and the thesaurus. In: Proceedings National Language Research Institute, Tokyo, Japan (1998)
23. Yarowsky, D.: Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. In: Proc. COLING 1992, Nantes, France (1992)

FCA-Based Two Dimensional Pattern Matching

Fritz Venter, Derrick G. Kourie, and Bruce W. Watson

Department of Computer Science, FASTAR Research Group
University of Pretoria, Pretoria, South Africa
fritz.venter@gmail.com,dkourie@cs.up.ac.za,bwatson@cs.up.ac.za
www.up.ac.za

Abstract. We propose a concept lattice-based approach to multiple two dimensional pattern matching problems. It is assumed that a pattern can be described as a set of vertices (or pixels) and that a small set of vertices around each vertex corresponds to an attribute in a concept lattice. Typically, an attribute should be a succinct characterisation of domain-dependent relevant information about the neighbourhood of the vertex. The set of objects in the lattice is the set of 2D patterns to be matched. Searching in the 2D image is carried out in reference to the intents of lattice concepts. Thus, by searching a small region of the text, one can efficiently identify which sets of pattern objects may potentially be found in a larger environment of that region. As experimental data, we use 2D images derived from microchip design layouts and 2D matching patterns derived from microchip design rules.

Keywords: 2d pattern matching, formal concept analysis, microchip design layout, microchip design rules.

1 Introduction

In this paper we propose that a *concept lattice* presents a promising avenue for two dimensional pattern matching. More specifically, we investigate the idea that a *concept lattice* (a data structure developed in the field of *Formal Concept Analysis*(FCA) - introduced in [2]) can be used to generate a highly integrated set of matching sequence paths to search for positions where *multiple* pixel matrices (images) match in a target image. As experimental data, we use two dimensional images derived from chip design layouts and design rules. The paper describes important steps of the experimental matching process. This process starts with the transformation of geometric relationships in microchip design rules to two dimensional images and ends with the actual matching algorithm that employs traversal of and operations on lattices built from these design rule pixel matrices. The experimental tools developed and results of this matching approach as applied to some existing design rules are discussed and some promising findings on the efficiency of this approach are presented.

S. Ferré and S. Rudolph (Eds.): ICFCA 2009, LNAI 5548, pp. 299–313, 2009.

2 Defining the 2D Pattern Matching Domain

For the purpose of this paper we define the domain of research as matching multiple two dimensional match images in a target image. Specifically we use images derived from layouts and design rules as used in electronic design automation (EDA) software.

2.1 Layouts and Design Rules

To simplify pattern matching of design rules in layouts, layouts and design rules are treated structurally equivalent. For brevity we refer to a *layout* or *design rule* or any subset of a layout or design rule using the collective term *vset*. A vset is a set of vertices. We define a vertex v in such a set as the 5-tuple $\langle x_0, x_1, g, h, k\rangle$ where x_0 (x_1) is a the first (second) dimension displacement in the vertex plane and g is the index of the polygon to which the vertex belongs; h, the numeric identifier of the vertex in $polygon_p$; and k, the material identifier (with respect to some table of materials) of $polygon_p$. We also define utility functions to access elements of v as follows:

$$\begin{aligned} x(v) &= x_0 \\ y(v) &= x_1 \\ p(v) &= g \\ n(v) &= h \\ m(v) &= k \end{aligned}$$

To understand many of the formal definitions in further sections we briefly introduce *quantifications* using the following notation:

$$(\oplus a : R(a) : f(a))$$

where $\oplus$ is the associative and commutative *quantification operator* (with unit $e_\oplus$), a is the *dummy variable* introduced, R is the *range predicate* on the dummy, and f is the *quantified expression*. By definition, we have:

$$(\oplus a : false : f(a)) = e_\oplus$$

The following table lists some of the most commonly quantified operators, their quantified symbols, and their units:

Operator	$\vee$	$\wedge$	$\cup$	**min**	**max**	$+$
Symbol	$\exists$	$\forall$	$\bigcup$	**MIN**	**MAX**	Σ
Unit	$false$	$true$	$\emptyset$	$+\infty$	$-\infty$	0

As an example of quantification on set union, we can define the set E of all even numbers as the following expression:

$$E = (\bigcup x : x = 2z \wedge z \in \mathbb{Z} : \{x\})$$

where $\mathbb{Z}$ is the set of all integers. One can thus say that E is the set that is created by concatenating all the single member sets $\{x\}$, where $x = 2z$ and z is a member of all integers $\mathbb{Z}$. In this example the *quantification operator* is $\bigcup$, the *range predicate* is $x = 2z \wedge z \in \mathbb{Z}$ and the *quantified expression* is $\{x\}$. The dummy variables used are x and z. Further elaboration on quantifications can be found in [1].

Applying the quantification notation introduced above, we can now define a vset s as the set of vertices $(\bigcup\ v : v \in \mathbb{V} : \{v\})$ where $\mathbb{V}$ is the set of all vertices. We define the set $\mathbb{S}$ as all vsets. We define a function *polygon* as a query on a vset s that gives the sub-vset of s that contains only the vertices from s that belong to a specific polygon.

$$\begin{aligned} &\mathit{polygon} : \mathbb{N} \times \mathbb{S} \to \mathbb{V} \ \text{ satisfying} \\ &\qquad\qquad \mathit{polygon}(a, s) = (\bigcup\ v : v \in s \wedge p(v) = a : \{v\}) \end{aligned}$$

2.2 From Vsets to Vimages

Although the option to match on vsets exists, this research applies matching to the output of a pre-processing phase that converts vsets to two dimensional matrices of pixels called *vimages.* Many existing 2D pattern matching algorithms can thus be applied for matching on these vimages. Examples of such algorithms can be found in [3], [4], [5], [6].

We define the *vimage* i as a set of pixels forming a two dimensional matrix. A pixel p is defined as the triple $\langle i_0, i_1, u\rangle$, where $\langle i_0, i_1\rangle$ are the coordinates of a cell of the matrix corresponding to i and u is the value of the pixel defined as the triple $\langle x_0, x_1, t\rangle$. The element t is a set of material identifiers corresponding to all polygons that a rectangle r covers geometrically. The coordinates x_0 (x_1) correspond to the first (second) dimension displacement in the vertex plane of the corner of r that is closest to the vertex plane origin. By convention this is normally the top-left corner of the rectangle. The rectangles corresponding to the pixels of a vimage form a non-overlapping cover of the rectangular area bounding all vertices in the vset from which the vimage is derived.

Figure 1 shows an example of the top-left corners of the tiled rectangles and material identifiers mapped onto a vset containing two polygons.

As with vertices we define utility functions to access elements of a pixel $p = \langle i_0, i_1, u\rangle$ with $u = \langle x_0, x_1, t\rangle$ as follows:

$$\begin{aligned} i(p) &= i_0 \\ j(p) &= i_1 \\ x(p) &= x_0 \\ y(p) &= x_1 \\ z(p) &= t \end{aligned}$$

We define $\mathbb{P}$ as all pixels. We define $\mathbb{I}$ as all vimages. The algorithm that derives a vimage from a vset (s) utilises the function $s2i_0 : \mathbb{S} \to \mathbb{I}$ satisfying

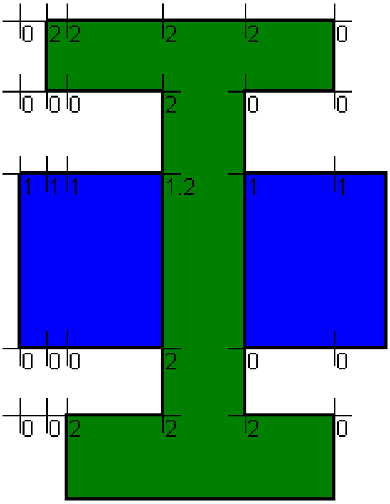


Fig. 1. Example of an image overlaid on a vset

$$s2i_0(s) = (\bigcup v_1, v_2 : v_1 \in s \wedge v_2 \in s :$$
$$\{v2p(v_1, s)\} \bigcup \{v2p(v_2, s)\} \bigcup \{v2p_r(v_1, v_2, s)\} \bigcup \{v2p_r(v_2, v_1, s)\})$$

The function $s2i_0$ gives a set of pixels (vimage) as a union of the following derived four pixels for every pair of vertices v_1 and v_2 in an input vset s.

- Two pixels that are directly derived from v_1 and v_2 using the function $v2p : \mathbb{V} \times \mathbb{S} \rightarrow \mathbb{P}$.
- Two pixels that are derived from v_1 and v_2 using a rotated combination of their vertex space coordinates, using the function $v2p_r : \mathbb{V} \times \mathbb{V} \times \mathbb{S} \rightarrow \mathbb{S}$

The functions $v2p$ and $v2p_r$ satisfy

$$v2p(v, s) = \langle x2i(x(v), s), y2j(y(v), s), \langle x(v), y(v), \emptyset \rangle \rangle$$
$$v2p_r(v_1, v_2, s) = \langle x2i(x(v_1), s), y2j(y(v_2), s), \langle x(v_1), y(v_2), \emptyset \rangle \rangle$$

Two new functions are introduced here. The function $x2i : \mathbb{N} \rightarrow \mathbb{N}$ ($y2j : \mathbb{N} \rightarrow \mathbb{N}$) takes the vertex plane x (y) ordinate of a vertex v and gives the index if the first (second) dimension of a vimage matrix derived from the vset s. Further definition of these functions is out of scope for this paper as they are considered implementation details of the experimental system that implements $s2i_0$.

Note that pixels in the vimage given by $s2i_0$ have empty lists of material numbers. This means that for a vset s:

$$(\forall p \; : p \in s2i_0(p) : z(p) = \emptyset)$$

To achieve the fully defined vimage from a vset s we use the function $s2i : \mathbb{S} \rightarrow \mathbb{I}$, satisfying

$$s2i(s) = imp(s2i_0(s), s)$$

where *imp* uses information of neighbours of each pixel in the input vimage to create a new vimage of which the pixels have fully populated material lists.

2.3 The Matching Requirement

The post-condition (*Post*) of the matching algorithm is to find the output set O. *Post* is thus defined as follows:

$$O = (\bigcup s_m, s_{mm} : s_{mm} \in s_m \wedge (\exists x, y :: match_s(s_{mm}, s, x, y)) : \{\langle s_{mm}, x, y \rangle\})$$

where s_m is the set of vsets to match and s is the vset against which to match. The function $match_s$ has the signature

$$match_s : \mathbb{S} \times \mathbb{S} \times \mathbb{N} \times \mathbb{N} \to \mathbb{B}$$

As discussed above we are not matching directly on vsets. Therefore we rewrite this post-condition in terms of vimages as follows;

$$O = (\bigcup i_m, i_{mm} : i_{mm} \in i_m \wedge (\exists x, y :: match_i(i_{mm}, i, x, y)) : \{\langle i_{mm}, x, y \rangle\})$$

This means that the matching system needs to store the triple $\langle i_{mm}, x, y \rangle$ in the output set O for each vset i_{mm} from the set of vimages i_m that matches i at offset $\langle x, y \rangle$ in the pixel matrix corresponding to i.
The function $match_i : \mathbb{I} \times \mathbb{I} \times \mathbb{N} \times \mathbb{N} \to \mathbb{B}$ satisfies

$$\begin{aligned} match_i(i_m, i, x, y) = (\forall p_m : p_m \in i_m : (\exists p_i : p_i \in i :\\ & i(p_m) + x = i(p_i) \wedge \\ & j(p_m) + y = j(p_i) \wedge \\ & z(p_m) = z(p_i))). \end{aligned}$$

3 A Context of 2D Patterns

The matching mechanism that achieves the above-mentioned post-condition is now considered. A naïve approach is to check every vimage $i_{mm} \in i_m$ pixel by pixel and to record matches accordingly. However, since one expects that there will be some i_{mm} that (partially) match each other, intuition would suggest that such matching subsets of vimages should only be checked once to optimise the process of matching all $i_{mm} \in i_m$. Put differently, if a partial order exists amongst the i_{mm}, then such an order could suggest a more optimal search order.

The partial order that a *lattice of formal concepts* exhibits offers a promising "experimental mechanism" to test the hypothesis that order amongst elements of i_m may be exploited to yield more efficient search strategies.

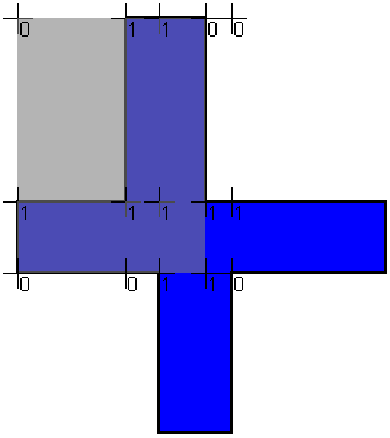


Fig. 2. An attribute pattern depicted by the shaded area of a vimage

3.1 Mapping the 2D Pattern Matching Domain to Concept Lattices

To apply the matching problem introduced thus far to *Formal Concept Analysis* we define the set of objects in the *formal context* as string identifiers (names) of the set of vimages to match (i_m) in some vimage i.

The attributes of the *formal context* are small rectangular (sub)vimages derived from each $i_{mm} \in i_m$ (ie each object in the context. These sub-images are called *attribute patterns*. The value of such an attribute is a one-dimensional string derived as follows:

$$c = \langle i \rangle, r = \langle j \rangle \backslash \langle materiallist \rangle$$

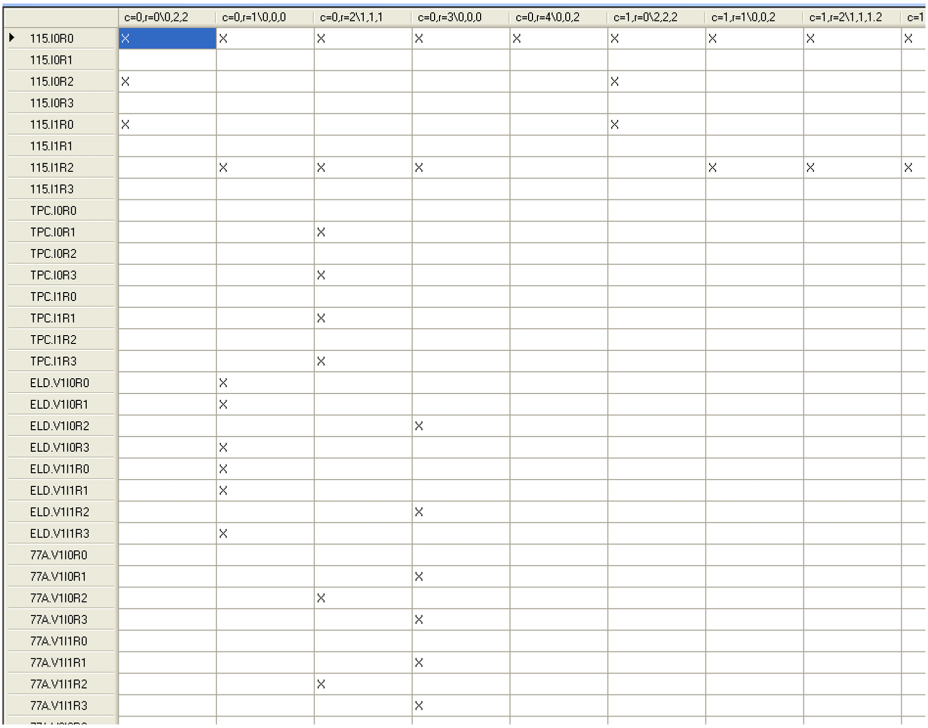
where $\langle i \rangle$ ($\langle j \rangle$) is an offset in the first (second) dimension of the matrix associated with the object (vimage). The suffix $\langle materiallist \rangle$ is a string derived from the 2D sequence of pixel values (material identifiers) that occur in the attribute pattern. Each row consists of a sequence of comma-delimited material identifier lists and rows are delimited by a "\".

As an example, the attribute pattern $c = 0, r = 0\backslash 0, 1, 1\backslash 1, 1, 1$ has been derived from the shaded section of the vimage in figure 2.

Figure 3 shows a section of a context derived from experimental vimage matching data.

4 Experimental Tools and Initial Results

The above sections provide the formal basis for a system that transforms a set of *vsets* (m_s) (to match in some target *vset* (s)) to a corresponding set of *vimages* (i_m) to match on a corresponding target *vimage* (i). Such a system can then derive a *formal context* from the set i_m using *attribute patterns* derived from i_m.



	c=0,r=0\0,2,2	c=0,r=1\0,0,0	c=0,r=2\1,1,1	c=0,r=3\0,0,0	c=0,r=4\0,0,2	c=1,r=0\2,2,2	c=1,r=1\0,0,2	c=1,r=2\1,1,1.2	c=1
115.I0R0	×	×	×	×	×	×	×	×	×
115.I0R1									
115.I0R2	×					×			
115.I0R3									
115.I1R0	×					×			
115.I1R1									
115.I1R2		×	×	×			×	×	×
115.I1R3									
TPC.I0R0									
TPC.I0R1			×						
TPC.I0R2									
TPC.I0R3			×						
TPC.I1R0									
TPC.I1R1			×						
TPC.I1R2									
TPC.I1R3			×						
ELD.V1I0R0		×							
ELD.V1I0R1		×							
ELD.V1I0R2				×					
ELD.V1I0R3		×							
ELD.V1I1R0		×							
ELD.V1I1R1		×							
ELD.V1I1R2				×					
ELD.V1I1R3		×							
77A.V1I0R0									
77A.V1I0R1				×					
77A.V1I0R2			×						
77A.V1I0R3				×					
77A.V1I1R0									
77A.V1I1R1				×					
77A.V1I1R2			×						
77A.V1I1R3				×					

Fig. 3. Example of a context of *vimages* (as objects) and *attribute patterns* (as attributes)

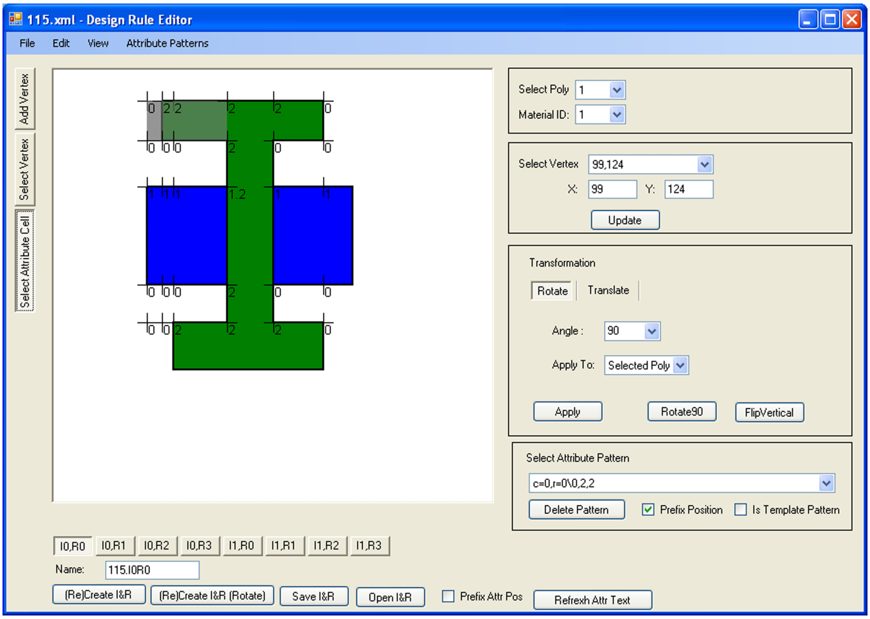
4.1 Experimental Software Tools

We developed two experimental tools that form part of such a system.
The first tool is a *"Design Rule Editor"* that provides the following features:

- A graphical vset editor - to create rectilinear polygons that form part of a microchip design rule with the ability to assign material identifiers to polygons.
- A vimage creation module that implements an algorithm for the function $s2i$ defined above.
- A module that creates four orthogonal rotations for two inflections (also known as "horizontal flips") of the vset and corresponding vimage created in the graphical vset editor.
- A module that makes it possible to define attribute patterns
- A module that uses an attribute pattern as a template to create all attribute patterns of the same dimensions as the template pattern that fits into every possible position in every inflection and rotation variation of the user defined vset and vimage.
- A facility to store all inflections / rotation variations and attribute patterns created during a session of usage of the "Design Rule Editor" in XML format.

The reason why we created the mechanism to create all the inflection / rotation variations of vimages and their associated attribute patterns is to create concept lattices that may suggest inflection / rotation independent "higher level rules" for use by chip design engineers to refine chip design rule sets. As will be shown later in the results discussion, it is also possible—under certain conditions—to stop the search process at such a concept, reducing the number of steps required by the search process itself.

Figure 4 shows the main user interface of the "Design Rule Editor".



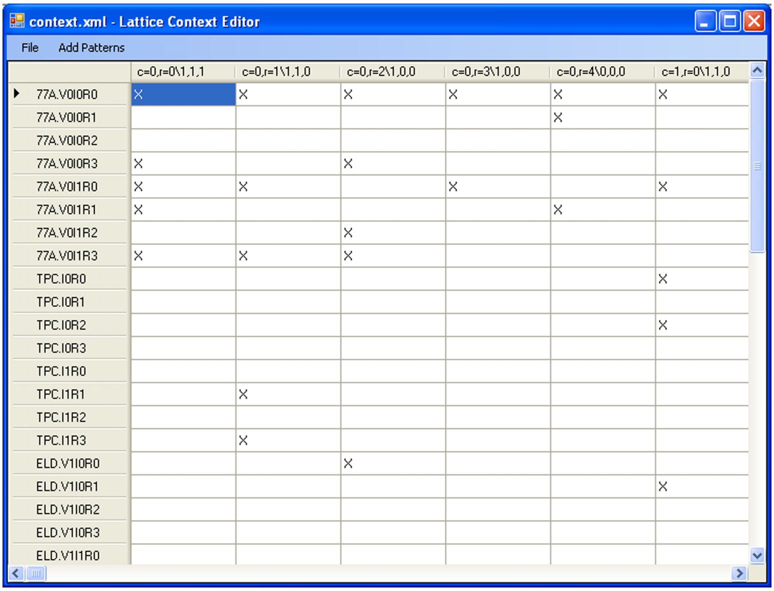
Select Poly: 1
Material ID: 1
Select Vertex: 99,124
X: 99 Y: 124
Angle : 90
Apply To: Selected Poly
Select Attribute Pattern: c=0,r=0\0,2,2
☑ Prefix Position □ Is Template Pattern
Name: 115.I0R0
□ Prefix Attr Pos

Fig. 4. Main user interface of the Design Rule Editor

The second tool that we developed is a "Lattice Context Editor" and provides the following features:

- A rule editor file import mechanism that reads a file created with the "Design Rule Editor" and (a) adds an attribute for every new attribute pattern in the file and (b) adds an object for every new vimage in the file.
- A facility to save the context in XML format
- A facility to export the context into the format accepted by the open source "Concept Explorer" tool.

Figure 5 shows the main user interface of the "Lattice Context Editor".



	c=0,r=0\1,1,1	c=0,r=1\1,1,0	c=0,r=2\1,0,0	c=0,r=3\1,0,0	c=0,r=4\0,0,0	c=1,r=0\1,1,0
77A.V0I0R0	X	X	X	X	X	X
77A.V0I0R1					X	
77A.V0I0R2						
77A.V0I0R3	X		X			
77A.V0I1R0	X	X		X		X
77A.V0I1R1	X				X	
77A.V0I1R2			X			
77A.V0I1R3	X	X	X			
TPC.I0R0						X
TPC.I0R1						
TPC.I0R2						X
TPC.I0R3						
TPC.I1R0						
TPC.I1R1		X				
TPC.I1R2						
TPC.I1R3		X				
ELD.V1I0R0			X			
ELD.V1I0R1						X
ELD.V1I0R2						
ELD.V1I0R3						
ELD.V1I1R0						

Fig. 5. Main user interface of the Lattice Context Editor

4.2 Initial Results

Experimental Data. The results that we achieved using the experimental tools mentioned above are discussed here in relation to a specific example data set. Using the *Design Rule Editor* we created the vsets associated with some existing chip design rules as defined by engineers in the industry.

Figure 6 shows the vsets as they were created in the *Design Rule Editor*.

Figure 7 shows the vsets that are orthogonal rotations and inflections of the vset “ELD VO”. The vimage of each vset is also shown as an overlay.

Figure 5 shows a section of the *formal context* derived from this data set.

The concept lattice line diagram associated with this context is extremely large. A section of this line diagram is shown in figure 8.

Matching Using Concept Exploration. To illustrate how a matching algorithm will use the above-mentioned lattice we try to match all objects (vimages) of the context in the vimage shown in figure 9.

The bands of sequences that have solid borders show the horizontal and vertical image matrix indices. The values to the left and above the these bands show indices of potentially matching vimages as identified during the matching process.

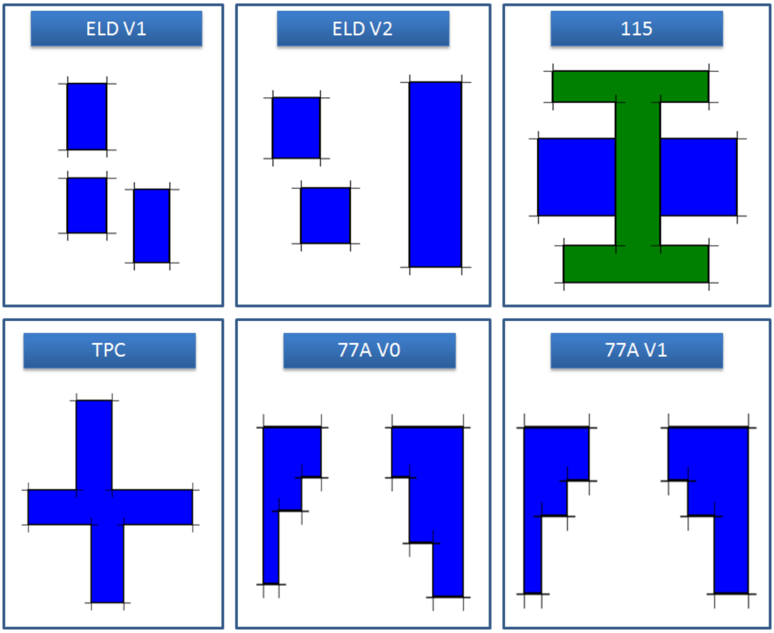


Fig. 6. Vsets of chip design rules

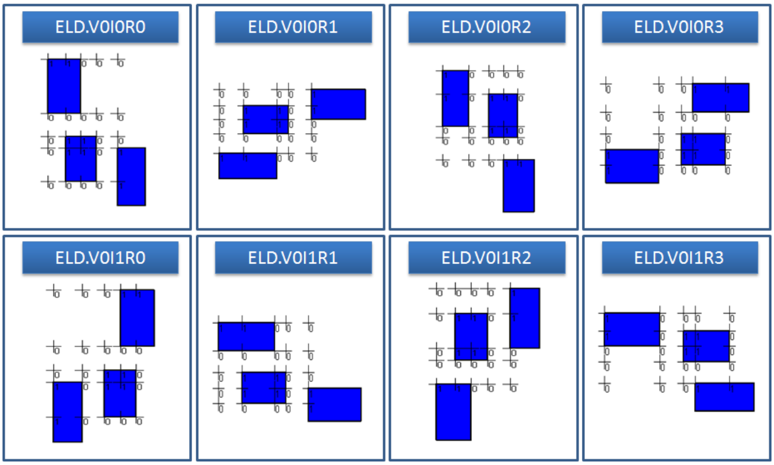


Fig. 7. Rotations and Inflections of the vimage ELD VO

The matching algorithm repeats the following two high level phases until the end of the target vimage is reached:

- *Match-Top-Level*
- *Match-Concepts-Below*

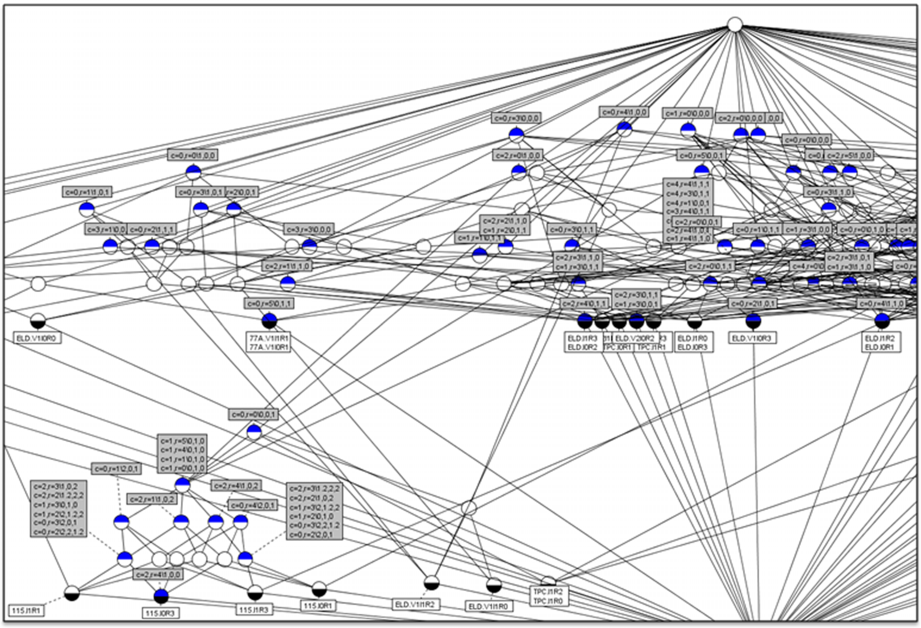


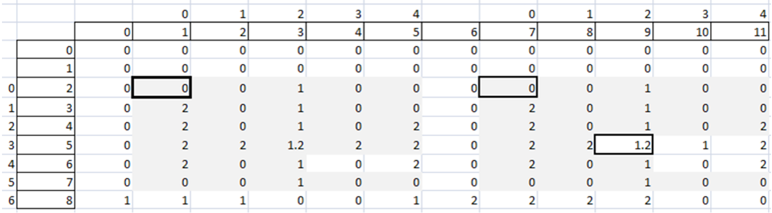
Fig. 8. Lattice Diagram of Experimental Context

The *Match-Top-Level* phase executes the following high level sequence:

For each pixel p (accessed by traversing the target vimage i from left to right and top to bottom) check for an initial match using the function $match_{B0}$ applied to all top level concepts.

$$match_{B0}(B, i, x, y) = (\exists b : b \in B : isor(b) \wedge match_i(i, ml2i(b), x, y))$$

where B is the intent of a top level concept, and $x = i(p)$ and $y = j(p)$ are indices into the vimage i and the function $ml2i$ transforms the $\langle materiallist \rangle$ part of the string associated with attribute b "back" to a vimage. The function $isor$ ("Is Origin") simply checks if the attribute string value contains the prefix $c = 0, r = 0$. This means that the algorithm that implements $match_{B0}$ looks for an attribute



			0	1	2	3	4		0	1	2	3	4
		0	1	2	3	4	5	6	7	8	9	10	11
	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	2	0	0	0	1	0	0	0	0	0	1	0	0
1	3	0	2	0	1	0	0	0	2	0	1	0	0
2	4	0	2	0	1	0	2	0	2	0	1	0	2
3	5	0	2	2	1.2	2	2	0	2	2	1.2	1	2
4	6	0	2	0	1	0	2	0	2	0	1	0	2
5	7	0	0	0	1	0	0	0	0	0	1	0	0
6	8	1	1	1	0	0	1	2	2	2	2	0	0

Fig. 9. Target image to match

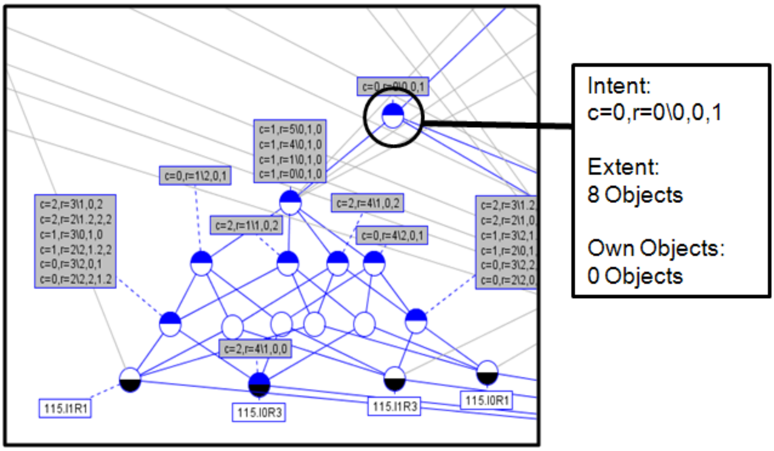


Fig. 10. Top level concept selected

in the intent of some concept that represents a matching neighbourhood in the target vimage. Using the the experimental vimage and concept lattice introduced above, this phase will result in the selection of a top level concept depicted by a concept with a black circle around it in figure 10.

In figure 9 above, the origin (top-left pixel) of this matching neighbourhood is shown as a shaded rectangle with a thick border. The *Match-Concepts-Below* phase executes the following high level sequence: Starting with the concept c (with intent B) found in the previous phase, verify that the $\langle materiallist \rangle$ part of the string associated with each attribute $b \in B$ transformed to a neighbourhood vimage i_n matches the part of vimage i located at the offset coordinate $\langle x + b2i(b), y + b2j(b) \rangle$ where $\langle x, y \rangle$ is the origin of the mapping process as determined in the *Match-Top-Level* phase and $b2i(b)$ ($b2j(b)$) is a function that gives the $\langle i \rangle$ ($\langle j \rangle$) part of the $c = \langle i \rangle, r = \langle j \rangle$ prefix of the string value of b. The function $match_B$ formally satisfies:

$$match_B(B, i, x, y) = (\forall\ b, i_n : b \in B \wedge i_n = ml2i(b) : \\ match_i(i, i_n, x + b2i(b), y + b2j(b)))$$

In a depth-first order, apply this function to all concepts in the downward closure (ie “below”) c. For all concepts that have *own objects* and for which $match_B$ returns *true*, add the names of all such *own objects* to the output set O defined in the post-condition of the matching algorithm as defined above.

Two optimisations have been made on this phase. The *first optimisation* of this phase is to not traverse the downward closures of any concepts for which $match_B$ returns $false$. The *second optimisation* of this phase is that only the set of attributes D that is the difference between the intent of the concept being processed (B) and the intent of previous (successfully matched) concept (B_p) needs to be matched.

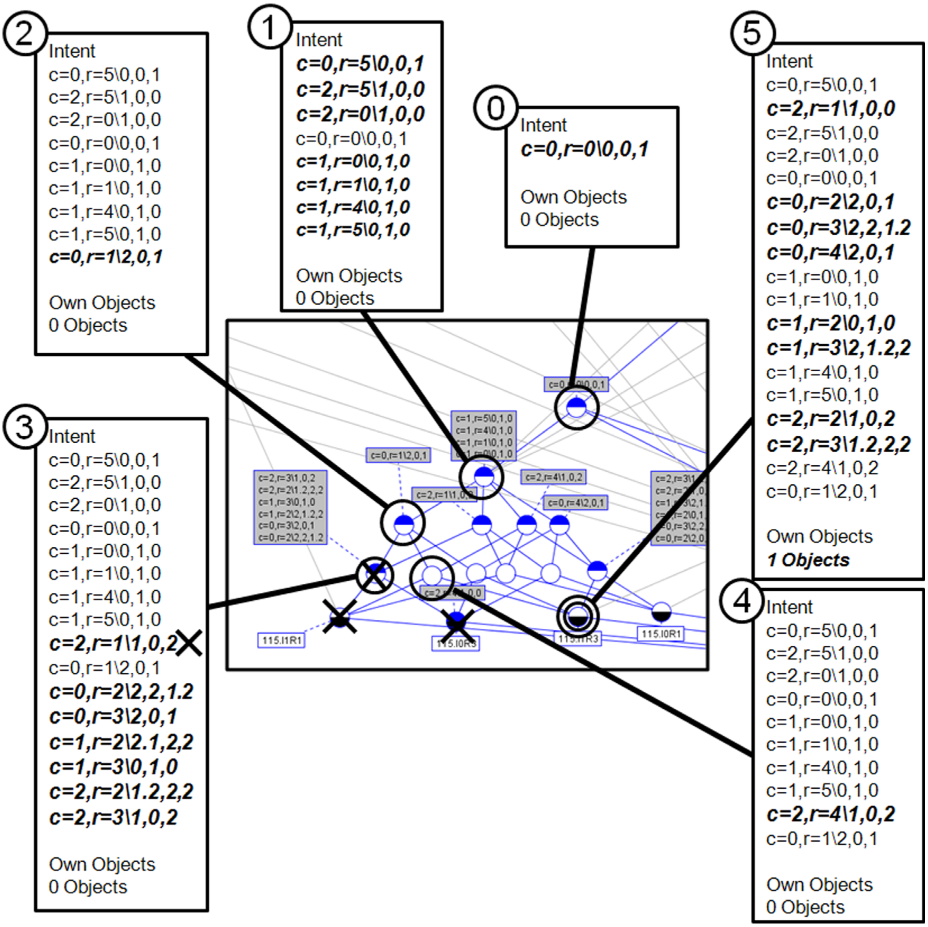


Fig. 11. Matching steps in the Match-Concepts-Below phase

To illustrate this phase of the algorithm, we show in figure 11 each matching step by circling the respective "visited" concept and showing its intent (B_n) (the intent pertaining to the concept visited in step n) and own objects in a rectangle linked to the respective concept node with a thick line. The step number (n) is shown inside a circle to the top-left of the respective rectangle. The set of attributes ($D_n = B_n \setminus B_{n-1}$) matched on the vimage in figure 9 in step n is shown in ***bold italics***. Note that for completeness we show the result of the *Match-Top-Level* as *Step 0*. Concepts that do not match and concepts in the downward closures of such concepts are marked by overlaying a"**X**" shape on the respective concept node.

A few promising observations can be made from this example. *Firstly*, after finding the first matching concept in the Match-Top-Level phase, the number of attempted attribute pattern matches are very close to the number of attribute patterns of the successfully matched vimage (115.I1R3). This means that very little "unnecessary" matches were done. The optimisation of eliminating from the search space the downward closures of concepts that do not match, contributes to the efficient behaviour shown by this approach. *Secondly*, the second concept visited during the matching process (highlighted in *Step 1* in figure 11) only contains objects in its extent that are variations of the "115" rule. In the microchip design context, the attributes (attribute patterns) pertaining to this concept could be transformed to a "generic" new design rule that can be searched for in future instead of all the objects in its extent or such a new design rule can be be used in future as a replacement of all the rotations and inversions (objects) in its extent, thus simplifying the design rule set.

5 Conclusion

In this paper we formally introduce vsets and vimages. We also formally introduce the matching requirement for these domains. Experimental tools have been used to test the hypothesis that formal concept analysis can be used to create an index of a set of vimages for very efficient matching in a target vimage. Results obtained from using this technique on microchip design data show that this technique leads to a very efficient order in which attribute patterns pertaining to multiple vimages (to match) are checked at relevant locations in the target vimage. Further work will include

- Further analysis of the execution complexity of the matching algorithm.
- Comparison of this approach to current research in multiple 2D pattern matching techniques.
- More experiments to investigate this technique's efficiency in handling non-matching cases
- Analysis of the expected size of Concept Lattices created from vimages derived from chip design rules.
- Implementing the matching algorithm in software to test its execution efficiency.
- Comparison of this research into other work that applies FCA to a priori (eg geometrically) ordered attributes.
- Analysis of the characteristics of a data set that makes this approach more appropriate.

References

1. Ghries, D., Schneider, F.B.: A Logical Approach to Discrete Math, pp. 179–194. Springer, New York (1993)
2. Wille, R.: Restructuring lattice theory: an approach-based on hierarchies of concepts. In: Ordered Sets, pp. 445–470. Reidel, Dordrecht (1982)

3. Baeza-Yates, R., Regnier, M.: Fast Two Dimensional Pattern Matching (1993)
4. Bird, R.: Two Dimensionam Pattern Matching. Inf. Proc. Letters 6, 168–170 (1977)
5. Baker, T.: A technique for extending rapid exact string matching to arrays of more that one dimension. Inf. Proc. Letters 6, 168–170 (1978)
6. Zhu, R.F., Takaoka, T.: A technique for two-dimensional patern matching. CACM 32(9), 1110–1120 (1989)

RESTRUCTURING LATTICE THEORY:
AN APPROACH BASED ON HIERARCHIES OF CONCEPTS

Rudolf Wille
Fachbereich Mathematik
Technische Hochschule Darmstadt
6100 Darmstadt
Federal Republic of Germany

ABSTRACT

Lattice theory today reflects the general status of current mathematics: there is a rich production of theoretical concepts, results, and developments, many of which are reached by elaborate mental gymnastics; on the other hand, the connections of the theory to its surroundings are getting weaker and weaker, with the result that the theory and even many of its parts become more isolated. Restructuring lattice theory is an attempt to reinvigorate connections with our general culture by interpreting the theory as concretely as possible, and in this way to promote better communication between lattice theorists and potential users of lattice theory.

The approach reported here goes back to the origin of the lattice concept in nineteenth-century attempts to formalize logic, where a fundamental step was the reduction of a concept to its "extent". We propose to make the reduction less abstract by retaining in some measure the "intent" of a concept. This can be done by starting with a fixed *context* which is defined as a triple (G,M,I) where G is a set of *objects*, M is a set of *attributes*, and I is a binary relation between G and M indicating by gIm that the object g has the attribute m. There is a natural Galois connection between G and M defined by $A' = \{m \in M \mid gIm$ for all $g \in A\}$ for $A \subseteq G$ and $B' = \{g \in G \mid gIm$ for all $m \in B\}$ for $B \subseteq M$. Now, a *concept* of the context (G,M,I) is introduced as a pair (A,B) with $A \subseteq G$, $B \subseteq M$, $A' = B$, and $B' = A$, where A is called the *extent* and B the *intent* of the concept (A,B). The *hierarchy of concepts* given by the relation subconcept-superconcept is captured by the definition $(A_1,B_1) \leq (A_2,B_2) \Leftrightarrow A_1 \subseteq A_2 (\Leftrightarrow B_1 \supseteq B_2)$ for concepts (A_1,B_1) and (A_2,B_2) of (G,M,I). Let $L(G,M,I)$ be the

I. Rival (ed.), Ordered Sets, 314-339.

set of all concepts of (G,M,I). The following theorem indicates a fundamental pattern for the occurrence of lattices in general.

THEOREM: *Let (G,M,I) be a context. Then $(L(G,M,I),\leq)$ is a complete lattice* (called the *concept lattice* of (G,M,I)) *in which infima and suprema can be described as follows:*

$$\bigwedge_{i\in J}(A_i,B_i) = \left(\bigcap_{i\in J} A_i, \left(\bigcap_{i\in J} A_i\right)'\right),$$

$$\bigvee_{i\in J}(A_i,B_i) = \left(\left(\bigcap_{i\in J} B_i\right)', \bigcap_{i\in J} B_i\right).$$

Conversely, if L is a complete lattice then $L \cong (L(G,M,I),\leq)$ if and only if there are mappings $\gamma : G \to L$ and $\mu : M \to L$ such that γG is supremum-dense in L, μM is infimum-dense in L, and gIm is equivalent to $\gamma g \leq \mu m$ for all $g \in G$ and $m \in M$; in particular, $L \cong (L(L,L,\leq),\leq)$.

Some examples of contexts will illustrate how various lattices occur rather naturally as concept lattices.

(i) $(S,S,\neq)$ where S is a set.
(ii) $(\mathbb{N},\mathbb{N},1)$ where $\mathbb{N}$ is the set of all natural numbers.
(iii) $(V,V^*,\perp)$ where V is a finite-dimensional vector space.
(iv) $(V,\mathrm{Eq}(V),\models)$ where V is a variety of algebras.
(v) $(G\times G,\mathbb{R}^G,\sim)$ where G is a set of objects, $\mathbb{R}^G$ is the set of all real-valued functions on G, and $(g_1,g_2)\sim\alpha$ iff $\alpha g_1 = \alpha g_2$.

Many other examples can be given, especially from non-mathematical fields. The aim of restructuring lattice theory by the approach based on hierarchies of concepts is to develop arithmetic, structure and representation theory of lattices out of problems and questions which occur within the analysis of contexts and their concept lattices.

1. RESTRUCTURING LATTICE THEORY

Lattice theory today reflects the general status of current mathematics: there is a rich production of theoretical concepts, results, and developments, many of which are reached by elaborate mental gymnastics; on the other hand, the connections of the theory to its surroundings are getting weaker: the result is that the theory and even many of its parts become more isolated. This is not only a problem of lattice theory or of mathematics. Many sciences suffer from this effect of specialization. Scientists and philosophers are, of course, aware of the danger of this growing isolation. In [17], H. von Hentig extensively discusses the status of the humanities and sciences today. One consequence is his charge to "restructure" theoretical developments in order to integrate and rationalize origins, connections, interpretations, and applications. In particular, abstract developments should be brought back to the commonplace in perception, thinking, and action. Thus, *restructuring lattice theory* is understood as an attempt to unfold lattice-theoretical concepts, results, and methods in a continuous relationship with their surroundings (cf. Wille [40], [41]). One basic aim is to promote better communication between lattice theorists and potential users of lattice theory.

The important current monographs on lattice theory are by G. Birkhoff [3], P. Crawley and R.P. Dilworth [8], and G. Grätzer [15]. They are primarily concerned with "systematically developing a body of results at the heart of the subject" [8, p. V]. In an approach to restructure lattice theory we are first confronted with the question: *Why develop lattice theory?* G. Birkhoff in [3] refers to an earlier article "What can lattices do for you?" [4], where he attempts to rationalize the statement: "lattice theory has helped to simplify, unify and generalize mathematics, and it has suggested many interesting problems". In his first survey paper on "lattices and their applications" [2], G. Birkhoff set up a more general viewpoint for lattice theory: "lattice theory provides a proper vocabulary for discussing order, and especially systems which are in any sense hierarchies". Notwithstanding this broad point of view, it was the abstract level of accepting lattice theory as a theory of "substructures" in mathematics which brought the breakthrough in the 1930's and established lattice theory as a mathematical discipline (cf. Mehrtens [24]).

The approach to lattice theory outlined in this paper is based on an attempt to reinvigorate the general view of order. For this purpose we go back to the origin of the lattice concept in nineteenth-century attempts to formalize logic, where the study of hierarchies of concepts played a central rôle (cf. Schröder [35]). Traditional philosophy considers a *concept* to be determined by its "extent" [extension] and its "intent" [intension, comprehension]: the *extent* consists of all objects (or entities) belonging to the concept while the *intent* is the multitude of all attributes (or properties) valid for all those objects (cf. Wagner [36]). The *hierarchy* of concepts is given by the relation of "subconcept" to "superconcept", i.e. the extent of the subconcept is contained in the extent of the superconcept while, reciprocally, the intent of the superconcept is contained in the intent of the subconcept. Paradigmatic is the example: "human being" is a subconcept of the superconcept "being". Listing all objects and attributes belonging to a given concept is almost never possible. For practical reasons then, there is the proposal to limit a "context" by fixing the set of objects and the set of attributes (cf. Deutsches Institut für Normung [11], [12]). In set-theoretical language, this gives rise to lattices whose elements correspond to the concepts of the context and whose order comes from the hierarchy of concepts. The approach reported here takes these "concept lattices" as the basis and discusses how parts of arithmetic, structure and representation theory of lattices may be developed out of problems and questions which occur within the analysis of contexts and their hierarchies of concepts.

2. CONCEPT LATTICES

We start defining a *context* as a triple (G,M,I) where G and M are sets, and I is a binary relation between G and M; the elements of G and M are called *objects* [Gegenstände] and *attributes* [Merkmale], respectively. If gIm for $g \in G$ and $m \in M$ we say: the object g *has* the attribute m. For $A \subseteq G$ and $B \subseteq M$ we define

$$\begin{aligned} A' &:= \{m \in M \mid gIm \text{ for all } g \in A\}, \\ B' &:= \{g \in G \mid gIm \text{ for all } m \in B\}. \end{aligned}$$

The mappings given by $A \mapsto A'$ and $B \mapsto B'$ are said to form a *Galois connection* between the power sets of G and M, i.e. they fulfill the following basic properties (cf. Birkhoff [3; pp. 122-125]).

PROPOSITION. *For a context* (G,M,I):

(1) $A_1 \subseteq A_2$ *implies* $A_1' \supseteq A_2'$ *for* $A_1, A_2 \subseteq G$,

(1') $B_1 \subseteq B_2$ *implies* $B_1' \supseteq B_2'$ *for* $B_1, B_2 \subseteq M$,

(2) $A \subseteq A''$ *and* $A' = A'''$ *for* $A \subseteq G$,

(2') $B \subseteq B''$ *and* $B' = B'''$ *for* $B \subseteq M$.

Now, a *concept* [Begriff] of the context (G,M,I) may be defined as a pair (A,B) where $A \subseteq G$, $B \subseteq M$, $A' = B$, and $B' = A$; A and B are called the *extent* and the *intent* of the concept (A,B), respectively. The hierarchy of concepts is captured by the definition

$$(A_1,B_1) \leq (A_2,B_2) :\Leftrightarrow A_1 \subseteq A_2 \quad (\Leftrightarrow B_1 \supseteq B_2)$$

for concepts (A_1,B_1) and (A_2,B_2) of (G,M,I); (A_1,B_1) is called the *subconcept* of (A_2,B_2), and (A_2,B_2) is called the *superconcept* of (A_1,B_1). To formulate the basic theorem (cf. Banaschewski [1], Schmidt [34]) which indicates a fundamental pattern for the occurrence of lattices in general, we need some further definitions. Let $\mathfrak{B}(G,M,I)$ be the set of all concepts of the context (G,M,I) and let $\underline{\mathfrak{B}}(G,M,I) := (\mathfrak{B}(G,M,I),\leq)$. A subset D of a complete lattice L is called *infimum-dense* (*supremum-dense*) if $L = \{\bigwedge X \mid X \subseteq D\}$ $(L = \{\bigvee X \mid X \subseteq D\})$.

THEOREM. *Let (G,M,I) be a context. Then $\underline{\mathfrak{B}}(G,M,I)$ is a complete lattice, called the* concept lattice *[Begriffsverband] of (G,M,I) in which infima and suprema can be described as follows:*

$$\bigwedge_{j\in J} (A_j,B_j) = \Big(\bigcap_{j\in J} A_j, \big(\bigcap_{j\in J} A_j\big)'\Big),$$

$$\bigvee_{j\in J} (A_j,B_j) = \Big(\big(\bigcap_{j\in J} B_j\big)', \bigcap_{j\in J} B_j\Big).$$

Conversely, if L is a complete lattice then $L \cong \underline{\mathfrak{B}}(G,M,I)$ if and only if there are mappings $\gamma : G \to L$ and $\mu : M \to L$ such that γG is supremum-dense in L, μM is infimum-dense in L, and gIm is equivalent to $\gamma g \leq \mu m$ for all $g \in G$ and $m \in M$; in particular, $L \cong \underline{\mathfrak{B}}(L,L,\leq)$.

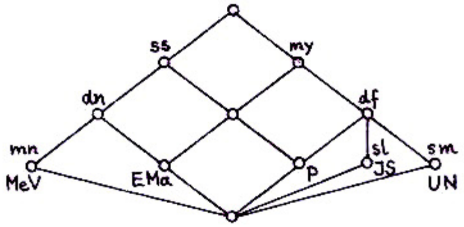
PROOF. Obviously, the relation $\leq$ is an order on $\mathfrak{B}(G,M,I)$. Let $(A_j,B_j) \in \mathfrak{B}(G,M,I)$ for $j \in J$. By the preceding proposition, $\bigcap_{j\in J} A_j \subseteq (\bigcap_{j\in J} A_j)''$ and $(\bigcap_{j\in J} A_j)'' \subseteq A_k'' = A_k$ for each $k \in J$ wherefore $(\bigcap_{j\in J} A_j)'' = \bigcap_{j\in J} A_j$ and hence $(\bigcap_{j\in J} A_j, (\bigcap_{j\in J} A_j)') \in \mathfrak{B}(G,M,I)$. If $(A,B) \in \mathfrak{B}(G,M,I)$ such that $(A,B) \leq (A_j,B_j)$ for all $j \in J$, i.e. $A \subseteq A_j$ for all $j \in J$, then $A \subseteq \bigcap_{j\in J} A_j$ and hence $(A,B) \leq (\bigcap_{j\in J} A_j, (\bigcap_{j\in J} A_j)')$. Thus, $(\bigcap_{j\in J} A_j, (\bigcap_{j\in J} A_j)')$ is the infimum of the (A_j,B_j) $(j \in J)$. The proof for the supremum goes analogously. Now, let φ be an isomorphism from $\underline{\mathfrak{B}}(G,M,I)$ onto a complete lattice L. We define $\gamma g := \varphi(\{g\}'',\{g\}')$ for $g \in G$ and $\mu m := \varphi(\{m\}',\{m\}'')$ for $m \in M$. Since $A = \bigcup_{g\in A} \{g\}''$ and $B = \bigcup_{m\in B} \{m\}''$ for every $(A,B) \in \mathfrak{B}(G,M,I)$, γG is supremum-dense in L and μM is infimum-dense in L, respectively. Furthermore, $gIm \Leftrightarrow \{g\}'' \subseteq \{m\}' \Leftrightarrow \gamma g \leq \mu m$. Conversely, let L be a complete lattice and let $\gamma : G \to L$ and $\mu : M \to L$ arbitrary mappings having the desired properties. We define a mapping $\varphi\colon \mathfrak{B}(G,M,I) \to L$ by $\varphi(A,B) := \bigvee\gamma A$ for all $(A,B) \in \mathfrak{B}(G,M,I)$. Obviously, φ is order-preserving. For $x \in L$, let $\psi x := (\{g \in G \mid \gamma g \leq x\}, \{m \in M \mid x \leq \mu m\})$. Since γG is supremum-dense in L and μM is infimum-dense in L, $\bigvee\gamma\{g \in G \mid \gamma g \leq x\} = x = \bigwedge\mu\{m \in M \mid x \leq \mu m\}$. Therefore, because of $gIm \Leftrightarrow \gamma g \leq \mu m$, ψx is a concept of (G,M,I) and $\varphi\psi x = x$. Clearly, $(A,B) \leq \psi\varphi(A,B)$. As $\gamma g \leq \bigvee\gamma A \leq \mu m$ always implies gIm, we even have $(A,B) = \psi\varphi(A,B)$. Furthermore, ψ is order-preserving. Now, we can summarize: φ is an isomorphism from $\underline{\mathfrak{B}}(G,M,I)$ onto L and $\varphi^{-1} = \psi$. If we choose $G := L$, $M := L$, $I := \leq$, γ and μ as the identity on L, we get $L \cong \underline{\mathfrak{B}}(L,L,\leq)$.

For a finite lattice L with $J(L)$ as its set of $\vee$-irreducible elements and $M(L)$ as its set of $\wedge$-irreducible elements, the theorem yields $L \cong \underline{\mathfrak{B}}(J(L),M(L),\leq)$, and the context $(J(L),M(L),\leq)$ is smallest with the property that its concept lattice is isomorphic to L. In general, the proposition and the theorem makes apparent a dual situation which often gets a concrete meaning by interchanging objects and attributes. Formally, we consider (M,G,I^{-1}) as the *dual context* of the context (G,M,I) where the mapping given by $(A,B) \mapsto (B,A)$ is an antiisomorphism from $\underline{\mathfrak{B}}(G,M,I)$ onto $\underline{\mathfrak{B}}(M,G,I^{-1})$.

3. EXAMPLES

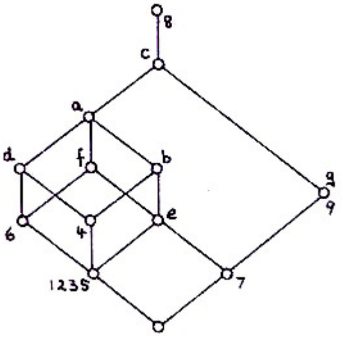
Some examples may illuminate how contexts give rise to concept lattices. The first example, a context of our planets, is described by the following table (cf. [25]) in which crosses indicate the relation between the objects and the attributes; the concept lattice is represented by its Hasse diagram where the labels indicate the mappings γ and μ of the preceding theorem.

	size			distance from sun		moon	
	small	medium	large	near	far	yes	no
Mercury	×			×			×
Venus	×			×			×
Earth	×			×		×	
Mars	×			×		×	
Jupiter			×		×	×	
Saturn			×		×	×	
Uranus		×			×	×	
Neptune		×			×	×	
Pluto	×				×	×	

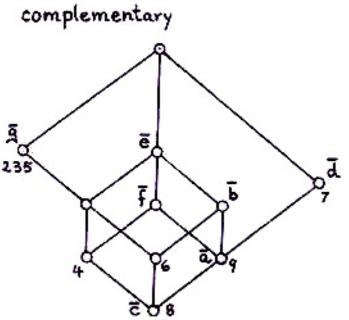


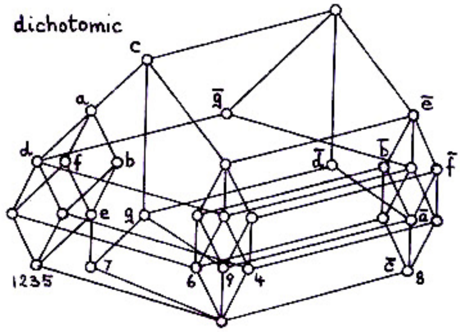
The second example is taken from the German national division of health and welfare (see Podlech [27]). The context indicates which personal information is needed by law for the different tasks of a local medical subdivision; tasks (objects) and information (attributes) are the following: 1. confirmation of requests for preventive care, 2. evaluations of regulations of insurance benefits, 3. confirmation of incapacity to work to insure success of treatment, 4. confirmation of correct diagnosis of incapacity to work, 5. work preliminary to rehabilitation, 6. verification of sickness, 7. advice to clients, 8. advice on general preventives, 9. carrying on the statistics of the medical service, a. name and address of the client ..., b. career history ..., c. kind of membership ..., d. name of responsible agency ..., e. family medical history, f. vocational education ..., g. number of certificates.

	a	b	c	d	e	f	g
1	×	×	×	×	×	×	
2	×	×	×	×	×	×	
3	×	×	×	×	×	×	
4	×	×	×	×			
5	×	×	×	×	×	×	
6	×		×	×		×	
7	×	×	×		×	×	×
8							
9			×				×



The concept lattice of a task-information-context (G,M,I) gives a natural hierarchical classification of the tasks and makes apparent the dependencies between information. But for certain purposes the concept lattice of the *complementary context* $(G,\overline{M},\overline{I})$ with $\overline{M} := \{\overline{m} | m \in M\}$ and $g\overline{I}\overline{m} :\Leftrightarrow \neg(gIm)$ is more suitable, for instance, if one wants to assign the tasks to different parties such that the access to personal information is mostly limited. Both lattices are combined in the concept lattice of the *dichotomic context* $(G,M\cup\overline{M},I\cup\overline{I})$, and it might be interesting to analyse how this lattice is dependent on the two smaller ones (cf. proposition in Section 4). Here we only present the Hasse diagrams with respect to the context above.





Experience has shown that concept lattices of "empirical" contexts which occur in rich multitude in many fields usually do not belong to the mostly studied classes of lattices as those of distributive, modular, semimodular, orthomodular etc. lattices. The situation becomes different if we consider contexts in mathematics (cf. Birkhoff [3]). For a set S, the concept lattice of $(S,S,\neq)$ is isomorphic to the Boolean lattice of all subsets of S since $(A,S\setminus A)$ is a concept of $(S,S,\neq)$ for every $A \subseteq S$. If $|$ is the divisibility relation on the set $\mathbb{N}$ of all natural numbers, $\mathfrak{B}(\mathbb{N},\mathbb{N},|)$ is distributive. Let V be a finite-dimensional vector space, let V^* be its dual space, and let $a \perp \varphi :\Leftrightarrow \varphi a = 0$ for $a \in V$ and $\varphi \in V^*$. Then $\mathfrak{B}(V,V^*,\perp)$ is isomorphic to the complemented modular lattice of all linear subspaces of V. If H is a Hilbert space and $\perp$ its orthogonality relation, then $\mathfrak{B}(H,H,\perp)$ is isomorphic to the orthomodular lattice of all closed linear subspaces of H. For a variety V of algebras and the set $Eq(V)$ of all equations valid in each algebra of V, the concept lattice of $(V,Eq(V),\models)$ is isomorphic to the coalgebraic lattice of all subvarieties of V. Many further examples could be given. These examples may suggest an extensive study of special classes of lattices. Here we want to continue with the general approach. We only mention one further class of examples which will be discussed more in Section 8: If $(P,\leq)$ is a (partially) ordered set then $\mathfrak{B}(P,P,\leq)$ is up to isomorphism the Dedekind-MacNeille completion of $(P,\leq)$.

4. THE DETERMINATION PROBLEM

The concept lattice can be understood as a basic answer to two fundamental questions concerning a context, namely the question of an appropriate classification of the objects and the question about the dependencies between the attributes. Hence an important problem is: *How can one determine the concept lattice of a given context* (G,M,I)? We get the most simple answer by forming (X'',X') for all $X \subseteq G$ (or (Y',Y'') for all $Y \subseteq M$). Of course, this method is far too costly and should only be used to determine single concepts. More successful, in particular if no computer is involved, is the approach using the following formulas for $(A,B) \in \mathfrak{B}(G,M,I)$:

$$A = \bigcap_{m\in B} \{m\}' \quad , \qquad\qquad B = \bigcap_{g\in A} \{g\}' \quad .$$

For instance, in the context of our planets one first determines the extents $\{ss\}'$, $\{sm\}'$, $\{sl\}'$, $\{dn\}'$, $\{df\}'$, $\{my\}'$, and $\{mn\}'$; then one gets all the other extents as intersections from those. From the extents it is easy to obtain the intents. For a larger context one may use both formulas simultaneously.

For the use of computers, it would be desirable to have "good" algorithms determining the concept lattice of a concept. Of course, designing those algorithms one has to consider how the concept lattice is handled afterwards. For instance, it might be presented by its Hasse diagram (see Section 4), it might be used to establish a measurement concerning the objects (see Section 8), or it might be a tool for a description of the dependencies between the attributes. In particular, one may elaborate the classification of objects further in the line of cluster analysis (for a recent survey on cluster analysis see Bock [6]): For many contexts one would like to have suitable partitions of the set of objects whose blocks are extents, i.e. they can be described by attributes; first experiences with randomly chosen contexts have shown that there are usually few such partitions if the cardinalities of their blocks are not becoming too small.

In certain situations, the determination problem can be attacked by a structural method which idea is to construct the concept lattice of a context by the concept lattices of some suitable subcontexts. We recall that the order R of the *direct product* of the two ordered sets (P,R_1) and (P,R_2) is defined by $(x_1,x_2)R(y_1,y_2) :\Leftrightarrow x_1R_1y_1$ and $x_2R_2y_2$. The *horizontal sum* of two bounded ordered sets (P_1,R_1) and (P_2,R_2) is obtained from their *cardinal sum* $(P_1\dot{\cup}P_2, R_1\dot{\cup}R_2)$ by identifying the smallest elements and the greatest elements of both ordered sets, respectively. The *vertical sum* of (P_1,R_1) and (P_2,R_2) is obtained from their *ordinal sum* $(P_1\dot{\cup}P_2, R_1\dot{\cup}R_2\dot{\cup}P_1\times P_2)$ by identifying the greatest element of

(P_1,R_1) with the smallest element of (P_2,R_2).

THEOREM. *Let (G_1,M_1,I_1) and (G_2,M_2,I_2) be contexts with $G_1\cap G_2=\emptyset$, $M_1\cap M_2=\emptyset$, $G_i'=\emptyset$ and $M_i'=\emptyset$ for $i=1,2$; let $L_i:=\underline{\mathfrak{B}}(G_i,M_i,I_i)$ for $i=1,2$. Then*

(1) *$\underline{\mathfrak{B}}(G_1\dot\cup G_2,M_1\dot\cup M_2,I_1\dot\cup I_2)$ is isomorphic to the horizontal sum of L_1 and L_2,*

(2) *$\underline{\mathfrak{B}}(G_1\dot\cup G_2,M_1\dot\cup M_2,I_1\dot\cup I_2\dot\cup G_1\times M_2)$ is isomorphic to the vertical sum of L_1 and L_2,*

(3) *$\underline{\mathfrak{B}}(G_1\dot\cup G_2,M_1\dot\cup M_2,I_1\dot\cup I_2\dot\cup G_1\times M_2\dot\cup G_2\times M_1)$ is isomorphic to the direct product of L_1 and L_2.*

PROOF. (1) Obviously, (A_i,B_i) with $\emptyset\neq A_i\subseteq G_i$ and $\emptyset\neq B_i\subseteq M_i$ is a concept of $(G_1\dot\cup G_2,M_1\dot\cup M_2,I_1\dot\cup I_2)$ if and only if it is a concept of (G_i,M_i,I_i) $(i=1,2)$. If $A\subseteq G_1\cup G_2$ and $A\cap G_i\neq\emptyset$ for $i=1,2$ then $A'=\emptyset$ and hence $A''=G_1\cup G_2$; furthermore $\emptyset''=\emptyset$. This proves (1).
(2) $(G_1\dot\cup G_2,M_1\dot\cup M_2,I_1\dot\cup I_2\dot\cup G_1\times M_2)$ has exactly the concepts $(\emptyset,G_1\cup G_2)$, $(A_1,B_1\cup M_2)$ with $A_1\neq\emptyset$ and $(A_1,B_1)\in\underline{\mathfrak{B}}(G_1,M_1,I_1)$, and $(A_2\cup G_1,B_2)$ with $A_2\neq\emptyset$ and $(A_2,B_2)\in\underline{\mathfrak{B}}(G_2,M_2,I_2)$.
(3) $(G_1\dot\cup G_2,M_1\dot\cup M_2,I_1\dot\cup I_2\dot\cup G_1\times M_2\dot\cup G_2\times M_1)$ has exactly the concepts $(A_1\cup A_2,B_1\cup B_2)$ with $(A_1,B_1)\in\underline{\mathfrak{B}}(G_1,M_1,I_1)$ and $(A_2,B_2)\in\underline{\mathfrak{B}}(G_2,M_2,I_2)$.

In cases where the preceding theorem cannot be applied, one may still reduce a given context to smaller contexts to apply the following proposition which can be easily verified using (3).

PROPOSITION. *Let (G,M,I) be a context, let $\{G_j\mid j\in J\}$ be a partition of G, and let $\{M_k\mid k\in K\}$ be a partition of M. Then*

(4) *a $\wedge$-embedding of $\underline{\mathfrak{B}}(G,M,I)$ into the direct product of the $\underline{\mathfrak{B}}(G_j,M,I\cap G_j\times M)$ $(j\in J)$ is given by $(A,B)\mapsto(A\cap G_j,(A\cap G_j)')_{j\in J}$,*

(4') *a $\vee$-embedding of $\underline{\mathfrak{B}}(G,M,I)$ into the direct product of the $\underline{\mathfrak{B}}(G,M_k,I\cap G\times M_k)$ $(k\in K)$ is given by $(A,B)\mapsto((B\cap M_k)',B\cap M_k)_{k\in K}$.*

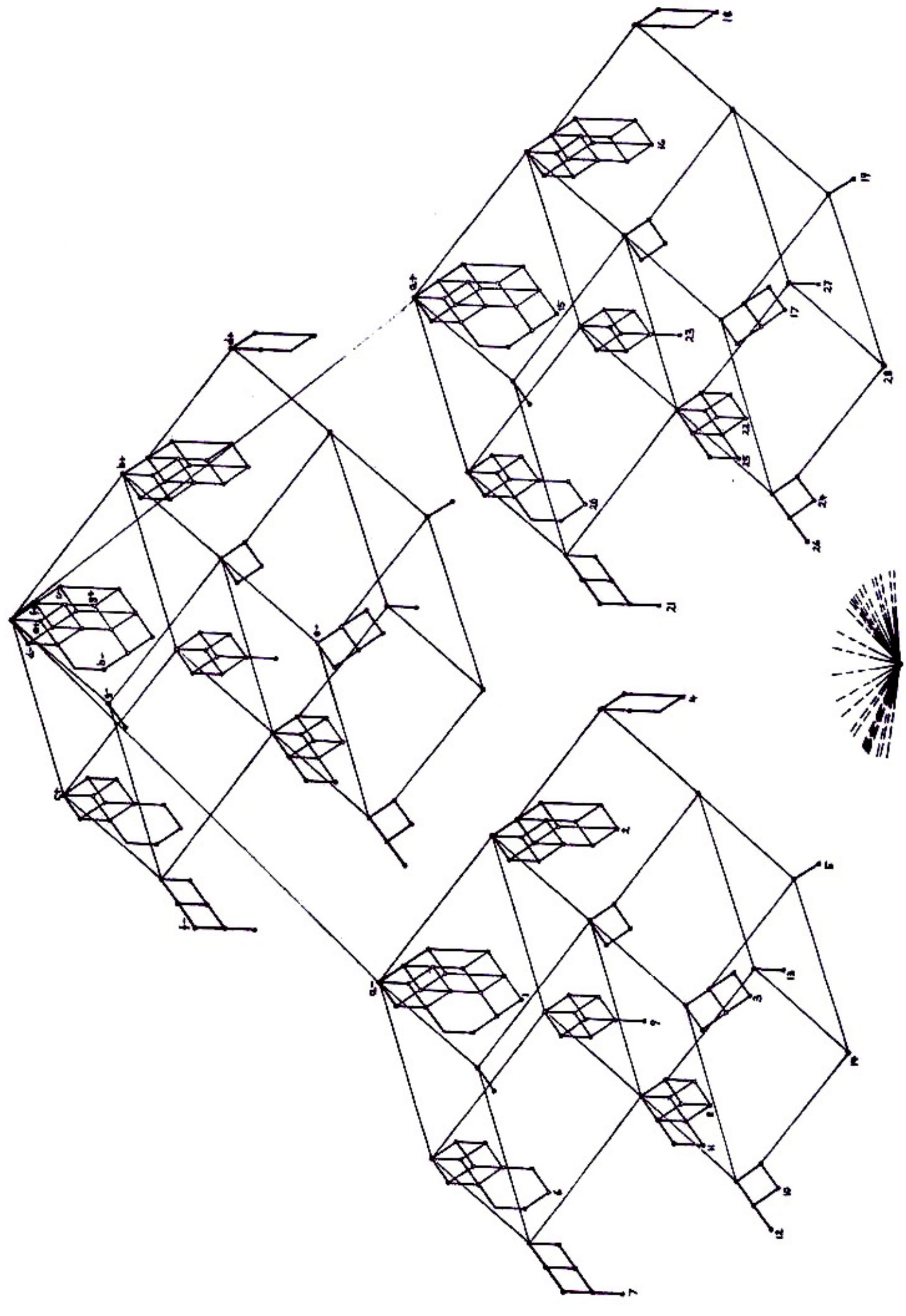
One hint is to apply (4) and (4') several times to reach subcontexts whose concept lattices can be easily determined; then in the direct product of these lattices one may find the images of the concepts of the original context under the order embedding given by (4) and (4'). The concept lattice of a context (G,M,I) is a chain if and only if G and M can be linearly ordered such that I becomes an order filter in $G\times M$. Thus, in the light of the described procedure, it might be a challenging problem to find for a given context (G,M,I) the smallest number of applications of (4) and (4') which yields an order embedding of $\underline{\mathfrak{B}}(G,M,I)$ into a direct product of (two-element) chains.

5. THE DESCRIPTION PROBLEM

Connected with the description problem is the basic problem: *How can one describe the concept lattice of a given context* (G,M,I)? "Good" solutions of this problem are helpful for further analysis and communication, especially with non-mathematicians. The most communicative description is given by Hasse diagrams. If we label in a Hasse diagram of $\underline{\mathfrak{B}}(G,M,I)$ the circle representing $(\{g\}'',\{g\}')$ by a name for g ($g \in G$) and the circle representing $(\{m\}',\{m\}'')$ by a name for m ($m \in M$), as in the examples in Section 3, the diagram retains the full information about (G,M,I) because gIm is equivalent to $(\{g\}'',\{g\}') \leq (\{m\}',\{m\}'')$ by the theorem in Section 2. Although Hasse diagrams are frequently used, it is surprising that there is no "theory of readable Hasse diagrams". Drawing a Hasse diagram is still more an art than a mechanical skill that even a computer can work out. Because of the absence of a theory, we shall only discuss the development of a Hasse diagram by an example.

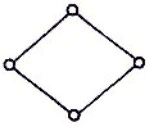
As an example, we consider the context of human Rhesus blood types by Wiener and Wechsler (see Diem, Lentner [13]). Its objects are the 28 Rhesus phenotypes which have been established up to 1956, and its attributes are the following seven test serums: a. Anti-Rh_0, b. Anti-rh', c. Anti-rh'', d. Anti-rh'$^{\mathrm{w}}$, e. Anti-hr', f. Anti-hr'', g. Anti-hr. The symbol + in the table below indicates when a blood type reacts positively with a test serum; no reaction is indicated by the symbol −. We analyse the context as a dichotomic context (cf. Section 3), i.e. we split each attribute x into the two attributes x+ and x−. The concept lattice of the dichotomic context is represented on the next page by a (partial) Hasse diagram in which several line segments are left out to allow easier reading. Besides the circle representing the least element, the diagram consists of three congruent components. If one moves the component above to one of the components below parallel along the line segment joining the top circles, then the traces of the circles will just add the missing line segments between the two components. Analogously, one has to move the subcomponents in each component parallel along the joining line segments; but the trace of a circle is a missing line segment only if the circle reaches another. To get more familiar with the diagram, one may do the exercise to find

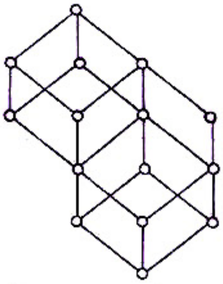
	a	b	c	d	e	f	g
1	−	−	−	−	+	+	+
2	−	+	−	−	+	+	+
3	−	+	−	−	−	+	−
4	−	+	−	+	+	+	+
5	−	+	−	+	−	+	−
6	−	−	+	−	+	+	+
7	−	−	+	−	+	−	−
8	−	+	+	−	+	+	−
9	−	+	+	−	+	+	+
10	−	+	+	−	−	+	−
11	−	+	+	−	+	−	−
12	−	+	+	−	−	−	−
13	−	+	+	+	+	+	−
14	−	+	+	+	−	+	−
15	+	−	−	−	+	+	+
16	+	+	−	−	+	+	+
17	+	+	−	−	−	+	−
18	+	+	−	+	+	+	+
19	+	+	−	+	−	+	−
20	+	−	+	−	+	+	+
21	+	−	+	−	+	−	−
22	+	+	+	−	+	+	−
23	+	+	+	−	+	+	+
24	+	+	+	−	−	+	−
25	+	+	+	−	+	−	−
26	+	+	+	−	−	−	−
27	+	+	+	+	+	+	−
28	+	+	+	+	−	+	−



a+
a-
b+
b-
d+
e-
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28

out which attributes are dependent on $\{a-,c+,g+\}$, i.e. contained in $\{a-,c+,g+\}''$ (take the meet of the elements labelled by $a-$, $c+$, and $g+$, and list all labelled elements above this meet: the result will be $\{a-,c+,g+\}'' = \{a-,c+,d-,e+,f+,g+\}$). The development of the Hasse diagram representing 286 elements has followed the scheme: Search for a suitable chain $L\times L =: R_0 \supset R_1 \supset R_2 \supset \dots \supset R_n := id_L$ of equivalence relations with convex equivalence classes on the lattice L such that $[a]R_{i-1}/R_i$ has dimension ≤ 3 for all $a \in L$ and $i = 1,\dots,n$. In our example we have $n = 3$; L/R_1 and $[1]R_1/R_2$ may be represented by the following Hasse diagrams:





Besides the description by Hasse diagrams, there are other useful methods for describing concept lattices. For instance, a concept lattice might be completely determined by some relative sublattice (e.g. its scaffolding) which can be diagrammed (cf. Wille [39]). A rich multitude of descriptions arises from the general idea of measurement which especially leads to numerical representations; this will be discussed further in Section 7.

6. FROM STRUCTURES TO CONCEPT LATTICES

Data models are often not contexts as defined in Section 2, i.e. relational structures with only unary relations; they frequently use many-placed relations and operations. Nevertheless, it is also desirable to have hierarchies of concepts for arbitrary relational structures available. For this, the general scheme is to derive (unary) attributes from the given structure and to form the concept lattice of the context consisting of the elements of the structure and these derived attributes. For instance, an n-ary relation R on a set G may be resolved into unary relations defined by $R_{(g_1,\dots,g_{i-1},g_{i+1},\dots,g_n)}(x) :\Leftrightarrow R(g_1,\dots,g_{i-1},x,g_{i+1},\dots,g_n)$ where $(g_1,\dots,g_{i-1},g_{i+1},\dots,g_n) \in G^{n-1}$ and $i \in \{1,\dots,n\}$. The derivation always depends on which concepts will be of interest with respect to the data. In this section we restrict our considerations to structures of "nominal character"; further structures are discussed in Section 7.

In many situations, the term "attribute" has an even more general meaning, i.e. an attribute may allow several values instead of one. A person may have any one of several values for the attribute "profession", for instance, "artist", "bus driver", "carpenter", etc.; there are also people with no profession (no value). We capture the general case by the definition of a *many-valued context* which is a quadruple (G,M,W,I) such that G, M, and W are sets, and I is a binary relation between G and $M \times W$ where $gI(m,w_1)$ and $gI(m,w_2)$ always imply $w_1 = w_2$; the elements of G, M, and W are called *objects*, *(many-valued) attributes*, and *attribute values*, respectively. If $gI(m,w)$ for $g \in G$, $m \in M$, and $w \in W$ we say: the object g *has* value w for the attribute m. If $|W| = n$ then (G,M,W,I) is called an *n-valued context*. A one-valued context can be considered as a context as defined in Section 2. Now, in the case where the attribute values are understood as nominal data, we suggest the derivation to "one-valued" attributes by just taking $(G,M \times W,I)$ as the derived context; let us indicate this case by calling (G,M,W,I) a *nominal context*. Hence, for a nominal context (G,M,W,I), its concept lattice is just $\underline{\mathfrak{L}}(G,M \times W,I)$. In this setting, the context of our planets in Section 3 can be understood as a nominal contexc with $M :=$ {size, distance from sun, moon} and $W :=$ {small, medium, large, near, far, yes, no}. The dichotomic contexts turn out to be always 2-valued nominal contexts.

It is common to assume that an n-valued context (G,M,W,I) is *complete*, i.e. for every $g \in G$ and $m \in M$ there exists a $w \in W$ with $gI(m,w)$; complete n-valued contexts are just a translated version of "relational data models" (cf. Date [9]). For attacking the determination problem and the description problem, the knowledge of special properties satisfied by the concept lattice of a complete nominal context is helpful. For a characterization of the concept lattices of complete n-valued nominal contexts the following definition is used. An element d of a complete lattice L has *valence* n if n is the smallest cardinality of a subset D of $L \setminus \{0\}$ containing d which is maximal with respect to the property that $x \wedge y = 0$ for all $x,y \in D$ with $x \neq y$. Recall that a lattice is *atomistic* if each lattice element is a supremum of atoms.

THEOREM. *A complete lattice L is isomorphic to a concept lattice of a complete n-valued nominal context if and only if L is atomistic and has an infimum-dense subset of elements of valence $\leq n$.*

PROOF. We assume that L has the desired properties. Let A be the set of all atoms of L and let $A(b) := \{a \in A \mid a \leq b\}$ for $b \in L$; furthermore, let M be the set of all elements of valence $\leq n$ in L and let W be a set with $|W| = n$ and $1 \in W$. For each $m \in M$ there must be a subset W_m of W with $1 \in W_m$ and an injective map $\iota_m : W_m \to L$ such that $\iota_m 1 = m$ and $\{A(\iota_m w) \mid w \in W_m\}$ is a partition of A. For $a \in A$, $m \in M$, and $w \in W$, we define $aI(m,w)$ if and only if $w \in W_m$

and $a \leq \iota_m w$. Since $\{\iota_m w \mid m \in M$ and $w \in W_m\}$ is infimum-dense in L and A is supremum-dense in L, L is isomorphic to $\underline{\mathcal{L}}(A,M{\times}W,I)$ by the theorem of Section 2 where (A,M,W,I) is a complete n-valued nominal context. Conversely, let (G,M,W,I) be a complete n-valued nominal context. If $\{g_1\}'' \neq \{g_2\}''$ for $g_1,g_2 \in G$, then w.o.l.g. there exists an $(m,w) \in M{\times}W$ with $g_1I(m,w)$ but not $g_2I(m,w)$. By the completeness, there is a $\bar{w} \in W$ with $g_2I(m,\bar{w})$. Now, $\{(m,w)\}' \cap \{(m,\bar{w})\}' = \emptyset$ implies $\{g_1\}'' \cap \{g_2\}'' = \emptyset$. Hence $\{\{g\}'' \mid g \in G\}$ is a partition of G. Thus, $\underline{\mathcal{L}}(G,M{\times}W,I)$ is atomistic. Obviously, a concept $(\{(m,w)\}',\{(m,w)\}'')$ has valence $\leq n$ in $\underline{\mathcal{L}}(G,M{\times}W,I)$ for all $(m,w) \in M{\times}W$. Since the set of all these concepts is infimum-dense in the concept lattice, the theorem is proved.

COROLLARY. *A finite lattice L is isomorphic to a concept lattice of a finite complete n-valued nominal context if and only if L is atomistic and every $\wedge$-irreducible element of L has valence $\leq n$.*

COROLLARY. *A finite lattice L is isomorphic to a concept lattice of a finite complete 2-valued nominal context if and only if L is atomistic and every $\wedge$-irreducible element of L has a pseudocomplement.*

The concept lattice of a nominal context (G,M,W,I) only clarifies the dependencies between the "one-valued" attributes of $M{\times}W$ but not the dependencies between the many-valued attributes of M which are in general defined as follows (since the elements of M can be understood as partial maps from G into W, we write $mg = w$ instead of $gI(m,w)$): For $m \in M$ and $B \subseteq M$, the attribute m is *dependent* on B if $mg_1 = mg_2$ for all $g_1,g_2 \in G$ whenever $ng_1 = ng_2$ for all $n \in B$ or equivalently, if there exists a mapping $\alpha: W^B \to B$ such that $\alpha(ng)_{n \in B} = mg$ for all objects g for which ng is defined for all $n \in B$ (special dependencies as monotone, linear, quadratic, etc. usually refer to restrictions on the mappings α). The closed subsets of M with respect to the defined dependence are exactly the intents of the context $(G{\times}G,M,\sim)$ where $(g_1,g_2) \sim m :\Leftrightarrow mg_1 = mg_2$. Here lattices of (partial) equivalence relations arise.

7. MEASUREMENT

Numerical descriptions of concept lattices may be viewed as a part of that area concerned with the measurements of objects. Concept lattices may even arise from many-valued contexts whose attribute values are numerical; there the attributes can be understood as some kind of measure of objects. The common theory of measurement defines a scale (measure) to be a (relatively closed) homomorphism from an empirical relational structure into a numerical relational structure (cf. Pflanzagl [26], Krantz, Luce, Suppes, Tversky [20], Roberts [31]). In our approach, it is desirable to have a type-free notion of measure (instead of the type-fixed notion homomorphism) wherefore we separate the notions of scale and measure. A *scale* is defined as a *numerical context* $\Sigma := (G_\Sigma, M_\Sigma, I_\Sigma)$ where G_Σ is a set of numbers or number-valued functions. Then a Σ-*measure* of a context (G,M,I) is a mapping μ from G into G_Σ such that $\mu^{-1}A$ is an extent of (G,M,I) for every extent A of Σ; μ is called *full* if μ^{-1} induces an isomorphism from the concept lattice of $(\mu G, M_\Sigma, I_\Sigma \cap \mu G \times M_\Sigma)$ onto $\underline{\mathfrak{B}}(G,M,I)$. The description problem appears now within the *representation problem* which asks for (full) Σ-measures of a given context into an appropriate scale Σ. Paradigms for scales are the *real ordinal scale* $\Sigma_O := (\mathbb{R}, M_O, \in)$ with $M_O := \{(-\infty, r] \mid r \in \mathbb{R}\}$, the *real interval scale* $\Sigma_I := (\mathbb{R}, M_I, \in)$ with $M_I := M_O \cup \{\{r-s, r, r+s\} \mid r \in \mathbb{R},\ s \in \mathbb{R}^+\}$, the *real ratio scale* $\Sigma_R := (\mathbb{R}, M_R, \in)$ with $M_R := M_I \cup \{\{r:s, r, r \cdot s\} \mid r \in \mathbb{R},\ s \in \mathbb{R}^+\}$, and their multidimensional analogues.

Our setting of measurement provides a general scheme for deriving a context from a many-valued context (G,M,W,I). If an attribute $m \in M$ is understood as a map into a scale Σ, then we propose to resolve m into the set $\{m\} \times M_\Sigma$ where an object $g \in G$ has the attribute (m,n) for $n \in M_\Sigma$ if $g \in m^{-1}\{n\}'$; this translation guarantees that m will be a Σ-measure of the derived context. For instance, if all attributes in M are maps into the real ordinal scale Σ_O, we obtain the context $(G, M \times \mathbb{R}, I_\leq)$ where $gI_\leq(m,r) :\Leftrightarrow mg \leq r$; in such a case, we call (G,M,W,I) an *ordinal context*. The embedding of $\underline{\mathfrak{B}}(G, M \times \mathbb{R}, I_\leq)$ into $\mathbb{R}^M$ will be discussed in a more general frame below.

A main question in the theory of measurement is whether a Σ-measure μ_0 of a context (G,M,I) is *unique* in the sense that for every Σ-measure μ of (G,M,I) which admits a bijection $\beta : \mu_0 G \to \mu G$ inducing an isomorphism between the corresponding concept lattices and satisfying $\beta_0 \mu_0 = \mu$, there exists a bijection $\sigma : G_\Sigma \to G_\Sigma$ which induces an automorphism of $\underline{\mathfrak{B}}(\Sigma)$ and extends β. This indicates that the representation problem also leads to a study of automorphisms of concept lattices. For instance, the automorphisms of the concept lattice of Σ_O, Σ_I, and Σ_R are induced by the order automorphisms, the positive affine mappings, and the positive linear mappings of $\mathbb{R}$, respectively (the set of all permutations of G_Σ inducing automorphisms on $\underline{\mathfrak{B}}(\Sigma)$ is usually considered as the *scale type* of Σ).

How lattice-theoretic developments are connected with a systematic study of the representation problem may be demonstrated in the case of order measurement. Let $(\Omega_t)_{t \in T}$ be a family of complete chains (of numbers) and let Ω be the direct product of the Ω_t $(t \in T)$; then $\Omega_\leq := (\Omega, \Omega, \leq)$ is called an *order scale of dimension* $|T|$ *and of length* $l(\Omega)$. Since Ω is a complete lattice, the mapping given by $x \mapsto ((x],[x))$ is an isomorphism from Ω onto $\underline{\mathfrak{B}}(\Omega_{\leq})$ by the theorem of Section 2 (we recall that $(x] := \{y \mid y \leq x\}$ and $[x) := \{z \mid x \leq z\}$).

PROPOSITION. *For a full $\Omega_{\leq}$-measure μ of a context (G,M,I), let $\overline{\mu}(A,B) := \bigvee \mu A$ for all $(A,B) \in \underline{\mathfrak{B}}(G,M,I)$. Then the mapping given by $\mu \mapsto \overline{\mu}$ is a bijection from the set of all full $\Omega_{\leq}$-measures of (G,M,I) onto the set of all $\bigvee$-embeddings of $\underline{\mathfrak{B}}(G,M,I)$ into Ω; in particular, $\mu g = \overline{\mu}(\{g\}'',\{g\}')$ for all $g \in G$.*

PROOF. If μ is a full $\Omega_{\leq}$-measure of (G,M,I) then an isomorphism from $\underline{\mathfrak{B}}(G,M,I)$ onto $\underline{\mathfrak{B}}(\mu G,\Omega,\leq)$ is given by $(A,B) \mapsto (\mu A,(\mu A)')$; in particular, μA is an extent of $(\mu G,\Omega,\leq)$ for each extent A of (G,M,I) and therefore $(\bigvee \mu A] \cap \mu G = \mu A$. Thus, $\overline{\mu}$ is injective. Let $(A_j,B_j) \in \underline{\mathfrak{B}}(G,M,I)$ for $j \in J$ and let $a := \bigvee_{j \in J} \bigvee \mu A_j$. There exists a concept (A,B) of (G,M,I) with $\mu A = (a] \cap \mu G$. Since $a \leq \bigvee \mu C$ for every concept (C,D) of (G,M,I) while $\bigvee_{j \in J} (A_j,B_j) \leq (C,D$ it follows $(A,B) = \bigvee_{j \in J} (A_j,B_j)$. Hence $\overline{\mu}$ is a $\bigvee$-embedding of $\underline{\mathfrak{B}}(G,M,I)$ into Ω. Since $(\mu g] \cap \mu G$ is the smallest extent of $(\mu G,\Omega,\leq)$ containing g $(g \in G)$, we have $(\mu g] \cap \mu G = \mu\{g\}''$ and hence $\mu g = \overline{\mu}(\{g\}'',\{g\}')$. Now, let ι be a $\bigvee$-embedding of $\underline{\mathfrak{B}}(G,M,I)$ into Ω. We define $\overline{\iota} g := \iota(\{g\}'',\{g\}')$ for all $g \in G$. Let A be an extent of (G,M,I). Obviously, $(\bigvee \overline{\iota} A] \cap \overline{\iota} G \supseteq \overline{\iota} A$. Let $\overline{\iota} g \leq \bigvee \iota A$. Because of $\bigvee \overline{\iota} A \leq \iota(A,A')$, it follows $\iota(\{g\}'',\{g\}') \leq \iota(A,A')$ and hence $(\{g\}'',\{g\}') \leq (A,A')$ wherefore $g \in A$ and $\iota g \in \overline{\iota} A$. Thus, $(\bigvee \iota A] \cap \iota G = \iota A$. This means that ι induces an injective map from $\underline{\mathfrak{B}}(G,M,I)$ into $\underline{\mathfrak{B}}(\iota G,\Omega,\leq)$. Let $a \in \Omega$ and let $A := \iota^{-1}(a] = \{g \in G \mid \iota(\{g\}'',\{g\}') \leq a\}$. Then $\iota(A'',A') = \iota \bigvee \{(\{g\}'',\{g\}') \mid g \in A\} = \bigvee \{\iota(\{g\}'',\{g\}') \mid g \in A\} \leq a$ which implies $A = A''$. Therefore ι is an $\Omega_{\leq}$-measure of (G,M,I) and induces a bijective map from $\underline{\mathfrak{B}}(G,M,I)$ onto $\underline{\mathfrak{B}}(\iota G,\Omega,\leq)$. Obviously, $(A,B) \leq (C,D)$ implies $\iota A \subseteq \iota C$. Conversely, if $\overline{\iota} A \subseteq \overline{\iota} C$ then $\iota(A,B) \leq \iota(C,D)$ and hence $(A,B) \leq (C,D)$. Now we can summarize that ι induces an isomorphism from $\underline{\mathfrak{B}}(G,M,I)$ onto $\underline{\mathfrak{B}}(\overline{\iota} G,\Omega,\leq)$, i.e. $\overline{\iota}$ is a full $\Omega_{\leq}$-measure of (G,M,I). Finally, $\overline{\overline{\mu}} g = \overline{\mu}(\{g\}'',\{g\}') = \bigvee \mu\{g\}'' = \mu g$ for each full $\Omega_{\leq}$-measure μ of (G,M,I), and $\overline{\overline{\iota}}(A,B) = \bigvee \overline{\iota} A = \bigvee\{\iota(\{g\}'',\{g\}') \mid g \in A\} = \iota \bigvee \{(\{g\}'',\{g\}') \mid g \in A\} = \iota(A,B)$ for each $\bigvee$-embedding ι of $\underline{\mathfrak{B}}(G,M,I)$ into Ω. This finishes the proof of the proposition.

By the preceding proposition, full order scaling can be analysed purely lattice-theoretically. For instance, the smallest dimension of an order scale $\Omega_{\leq}$ which admits a full $\Omega_{\leq}$-measure of (G,M,I) equals the $\bigvee$-*dimension* of $\underline{\mathfrak{B}}(G,M,I)$ which is defined as the smallest number of complete chains whose direct product admits a $\bigvee$-embedding of the lattice. The smallest length of such an order scale is given by the $\bigvee$-*rank* of $\underline{\mathfrak{B}}(G,M,I)$ which is defined as the smallest length of a direct product of complete chains which admits a $\bigvee$-embedding of the lattice. The following theorem gives a basic analysis of $\vee$-embeddings into direct products of complete chains in the case of finite lattices (cf. Lea [22], Ritzert [29]).

THEOREM. *Let L be a finite lattice. If $\chi := \{C_t | t \in T\}$ is a partition of $M(L)$ into chains then $\hat{\chi}: L \to \underset{t\in T}{\times}(C_t \cup \{1_L\})$ defined by $\hat{\chi}a := (a_t)_{t\in T}$ with $a_t := \min\{c \in C_t \cup \{1_L\} | a \leq c\}$ is a $\vee$-embedding. If $\iota: L \to \Omega$ is any $\vee$-embedding in a direct product of complete chains then there exists always a partition $\chi := \{C_t | t \in T\}$ of $M(L)$ into chains and a $\vee$-homomorphism κ from Ω onto $\underset{t\in T}{\times}(C_t \cup \{1_L\})$ such that κ maps $M(\Omega) \cup \{1_\Omega\}$ onto $M(\underset{t\in T}{\times}(C_t \cup \{1_L\})) \cup \{1_{\times C_t}\}$ and $\hat{\chi} = \kappa \circ \iota$.*

PROOF. Since every element of L is a meet of $\wedge$-irreducible elements, $\hat{\chi}$ is injective. Obviously, $a_t \vee b_t = (a \vee b)_t$ for all $a,b \in L$ and $t \in T$. Thus, $\hat{\chi}$ is a $\vee$-embedding of L into $\underset{t\in T}{\times}(C_t \cup \{1_L\})$. Let $\Omega := \underset{s\in S}{\times} \Omega_s$ and let $\pi_s: \Omega \to \Omega_s$ the canonical projection for $s \in S$. If $\pi_s \iota a = \pi_s \iota b^s$ and $a \leq b^s$ for $a \in M(L)$, $s \in S$, and $b^s \in L$, then $\iota a = \bigwedge_{s\in S} \iota b^s$ and hence $a = \bigwedge_{s \in S} b^s$ wherefore $a = b^s$ for some $s \in S$. Thus, there exists an injective map $\sigma: M(L) \to S$ such that $\pi_{\sigma a}\iota a = \pi_{\sigma a}\iota b$ implies $a \geq b$ for all $a \in M(L)$ and $b \in L$. $M(\Omega)$ contains exactly one element $m_{\sigma a}$ with $\pi_{\sigma a} m_{\sigma a} = \pi_{\sigma a}\iota a$ for each $a \in M(L)$. If $\sigma a = \sigma b$ and $m_{\sigma a} < m_{\sigma b}$ for $a,b \in M(L)$ then $a < b$, because otherwise we would have $b < a \vee b$ but $\pi_{\sigma b}\iota b = \pi_{\sigma b}\iota(a \vee b)$. Now, let $C_{\sigma a}$ be the chain consisting of all elements $b \in M(L)$ such that $m_{\sigma a}$ and $m_{\sigma b}$ are comparable in Ω; furthermore, let $T := \sigma M(L)$. Then $\chi := \{C_t | t \in T\}$ is a partition of $M(L)$ into chains. For $x \in \Omega$ we define $\kappa x := (x_t)_{t\in T}$ with $x_t := \min\{b \in C_t \cup \{1_L\} | x \leq m_{\sigma b}\}$ where $m_{\sigma 1_L} = 1_\Omega$. Again, $x_t \vee y_t \vee z_t \vee \ldots = (x \vee y \vee z \vee \ldots)_t$ for all $x,y,z,\ldots \in \Omega$ and $t \in T$; hence κ is a $\vee$-homomorphism. Obviously, $\kappa(M(\Omega) \cup \{1_\Omega\}) = M(\underset{t\in T}{\times}(C_t \cup \{1_L\})) \cup \{1_{\times C_t}\}$ and κ is surjective. If $\iota a \leq m_{\sigma b}$ for $a \in L$ and $b \in M(L)$ then $a \leq b$, because otherwise we would have $b < a \vee b$ but $\pi_{\sigma b}\iota b = \pi_{\sigma b}\iota(a \vee b)$. Therefore $(\iota a)_t = a_t$ for all $a \in L$ and $t \in T$. Hence $\hat{\chi} = \kappa \circ \iota$.

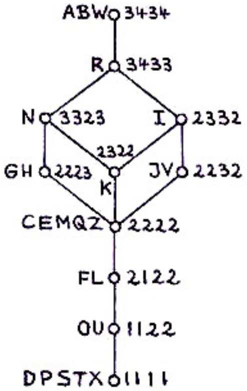
For the first assertion of the following corollary we used, of course, the theorem of R.P. Dilworth on the width of ordered sets [14].

COROLLARY. *For a finite lattice L, the $\vee$-dimension of L is equal to the width of $M(L)$ and the $\vee$-rank of L is equal to the cardinality of $M(L)$.*

COROLLARY. *A finite context (G,M,I) admits a unique full $\Omega_{\leq}$-measure with $\Omega = \underset{t\in T}{\times} \Omega_t$ if and only if there is exactly one* *partition* $\{C_t \mid t \in T\}$ *of* $M(L)$ *into chains such that the length of* Ω_t *is equal to the cardinality of* C_t *for all* $t \in T$*; then, in particular, the dimension and the length of the order scale* $\Omega_\leq$ *will be equal to the width and the cardinality of* $M(\underline{\mathfrak{B}}(G,M,I))$*, respectively.*

To end this section, we apply the results of our lattice-theoretical analysis to a simple example of an ordinal context. The context below is the list of the general grades on conduct (c), diligence (d), attentiveness (a), and orderliness (o) given to a class of the Ludwig-Georgs-Gymnasium (Darmstadt) for the winter term 1980/81 (the best grade is 1). The concept lattice has eight $\wedge$-irreducible elements, namely γR, γN, γI, γG, γJ, γF, γO, and γD; its width is two. Thus, the smallest dimension (length) of an order scale $\Omega_\leq$ allowing a full $\Omega_\leq$-measure is two (eight); for instance, one may take as chains $\{\gamma A, \gamma R, \gamma N, \gamma G, \gamma F, \gamma O, \gamma D\}$ and $\{\gamma A, \gamma I, \gamma J\}$. Since the concept lattice has an automorphism interchanging $\wedge$-irreducible elements, the context admits no unique full $\Omega_\leq$-measure of dimension two. Surprisingly, only the order scale built out of eight chains of length one admits a unique full $\Omega_\leq$-measure. In spite of the missing uniqueness, the conclusion seems reasonable that the given data are two-dimensional with even one main dimension. In general, it may be fruitful to explore whether the described order measurement can be used as basis for a factor analysis of ordinal data (cf. Rummel [32]).

	c	d	a	o
Anna	3	4	3	4
Berend	3	4	3	4
Christa	2	2	2	2
Dieter	1	1	1	1
Ernst	2	2	2	2
Fritz	2	1	2	2
Gerda	2	2	2	3
Horst	2	2	2	3
Ingolf	2	3	3	2
Jürgen	2	2	3	2
Karl	2	3	2	2
Linda	2	1	2	2
Manfred	2	2	2	2
Norbert	3	3	2	3
Olga	1	1	2	2
Paul	1	1	1	1
Quax	2	2	2	2
Rudolf	3	4	3	3
Stefan	1	1	1	1
Till	1	1	1	1
Uta	1	1	2	2
Volker	2	2	3	2
Walter	3	4	3	4
Xaver	1	1	1	1
Zora	2	2	2	2



8. COMPLETIONS OF PARTIAL CONCEPT LATTICES

Up to now, it was always assumed that the concept lattices are derived from known contexts. But in many situations one has only a vague knowledge of a context although many of its concepts are fairly clear. For instance, it seems to be impossible to write down a comprehensive context of musical instruments, but everyone uses concepts as violin, trumpet, guitar, string instrument, etc. and meaningful sentences such as "a violin is a string instrument". In such cases of concepts of a vague context we have a modified determination problem: *How can one extend a partial lattice of concepts to a complete concept lattice of some context?* The following theorem (see MacNeille [23], Schmidt [34]) describes the smallest solution, usually called the *Dedekind-MacNeille completion.*

THEOREM. *For an ordered set* $(P,\leq)$ *an embedding* ι *of* $(P,\leq)$ *into* $\underline{\mathfrak{B}}(P,P,\leq)$ *is defined by* $\iota x := ((x],[x))$ $(x \in P)$; *especially,* $\iota\bigwedge X = \bigwedge\iota X$ *and* $\iota\bigvee X = \bigvee\iota X$ *if* $\bigwedge X$ *and* $\bigvee X$ *exist in* $(P,\leq)$, *respectively. If* λ *is any embedding of* $(P,\leq)$ *into a complete lattice* L *then there exists an order embedding* κ *of* $\underline{\mathfrak{B}}(P,P,\leq)$ *into* L *such that* $\lambda = \kappa\circ\iota$.

PROOF. By definition, the concepts of $(P,P,\leq)$ are exactly the pairs (A,B) such that $A,B \subseteq P$ and $A' = \{x \in P \mid x \leq y \text{ for all } y \in B\}$, $B' = \{y \in P \mid x \leq y \text{ for all } x \in A\}$; in particular, $((x],[x))$ with $x \in P$ are concepts of $(P,P,\leq)$ what confirms ι as an embedding. If $\bigwedge X$ exists in $(P,\leq)$, then $(\bigwedge X] = \bigcap_{x\in X}(x]$ and hence $\iota\bigwedge X = ((\bigwedge X],[\bigwedge X)) = (\bigcap_{x\in X}(x],(\bigcap_{x\in X}(x])') = \bigwedge\iota X$ by the theorem of Section 2; the assertion for $\bigvee X$ is proved dually. Now, let λ be an embedding of $(P,\leq)$ into a complete lattice L. We define $\kappa(A,B) := \bigvee\lambda A$ for $(A,B) \in \underline{\mathfrak{B}}(P,P,\leq)$. Obviously, $\lambda = \kappa\circ\iota$. If $\kappa(A_1,B_1) \leq \kappa(A_2,B_2)$ then $\bigvee\lambda A_1 \leq \bigvee\lambda A_2 \leq \lambda b$ for all $b \in B_2$ and hence $b \in A_1' = B_1$ for all $b \in B_2$ because λ is an embedding. Thus, $B_1 \supseteq B_2$ and therefore $(A_1,B_1) \leq (A_2,B_2)$; in particular, this includes the injectivity of κ. Since κ is order-preserving, κ is the desired order embedding.

Instead of closing a partial concept lattice to its smallest completion, one is usually more interested to unfold the concepts as far as possible to reach a rich context. Since meet and join are the fundamental operations for concepts which may produce new concepts from known ones, they form a natural tool for the desired extensions. For instance, we may start with some known concepts of musical instruments; then we try to produce new concepts by forming various meets and joins and to separate them by listing more and more objects and attributes. Such a procedure would follow the free generating process in a lattice. Unfortunately, the common set-theoretical language does not allow us to speak of a "free complete lattice"; hence, there is in this sense no largest solution of the modified determination problem available. But if we strictly bound the cardinality of subsets of which meets and joins are taken by some fixed cardinal number α, then there exists a lattice freely α-generated by a given partial concept lattice (see Crawley, Dean [7]); for the final completion one may use the preceding theorem. Now, these large solutions of the modified determination problem give rise to a modified description problem: *How can one describe a lattice freely α-generated by a partial lattice?* The common answer uses lattice terms to describe the lattice elements and gives a procedure how the order of the lattice can be derived from the describing terms (see Whitman [37], [38], Dean [10], Crawley, Dean [7], Jónsson [19], Lakser [21], Grätzer, Lakser, Platt [16]).

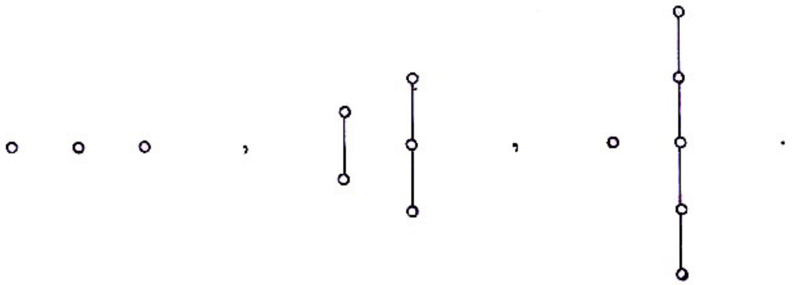
Again, descriptions by Hasse diagrams are desirable. But it seems that already a lattice freely ($\aleph_0$-) generated by three unordered elements cannot be described by a readable Hasse diagram. Nevertheless, many lattices freely generated by a partial lattice can be "drawn". So far, only the case of finite ordered sets has been completely analysed (if one agrees that a lattice having a sublattice freely generated by three unordered elements does not have a readable Hasse diagram). Those finite ordered sets whose free completions can be "drawn" are characterized by the theorem below (see Rival, Wille [30]). It is surprising that all these free completions can be embedded into a lattice freely generated by the six-element ordered set H; the Hasse diagrams of H and of its free completion are on the next page. The Hasse diagrams of all these free completions can be obtained by combining some of the ten diagrams given in [30]. In the case where a free completion of a partial lattice cannot be "drawn", it is still interesting to explore how far, following the generating process, the Hasse diagram can be developed.

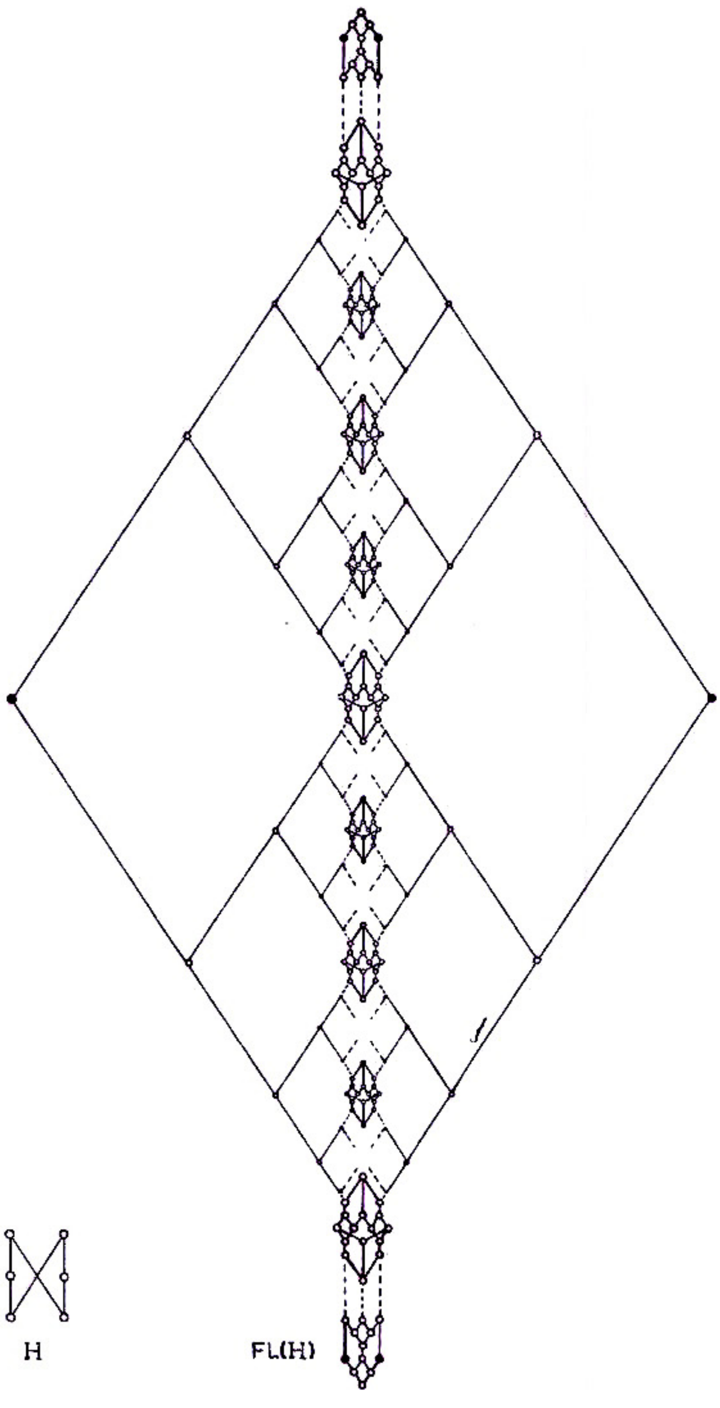
THEOREM. *For a finite ordered set P the following conditions are equivalent*:

(*i*) *A lattice freely generated by P does not contain a sublattice freely generated by three unordered elements*;

(*ii*) *A lattice freely generated by H has a subset isomorphic to P which freely generates a sublattice*;

(*iii*) *P contains no subset having one of the following Hasse diagrams*:




H
FL(H)

9. FURTHER REMARKS

This restructuring of lattice theory does not pretend to be complete in any sense. Many ties of lattice theory to its surroundings have not been mentioned. A comprehensive development of lattice theory should integrate and elaborate much more: for instance, origins as in logic (cf. Rasiowa [28]), connections as in geometry (cf. Birkhoff [5]), interpretations as in computer science (cf. Scott [33]), and applications as in quantum mechanics (cf. Hooker [18]). Especially, its significance to other mathematical disciplines which is still considered to be the main source has to be clarified. Besides the interpretation by hierarchies of concepts, other basic interpretations of lattices should be introduced; an important rôle is already played by the interpretation of lattices as closure systems. Feedback will always be given by the communication with potential users of lattice theory.

Coming back to the initial question: Why develop lattice theory? we may conclude that there is no short answer. The justification of lattice theory arises from its place in the landscape of our culture in general.

ACKNOWLEDGEMENTS. Thoughts and ideas in this paper have been matured to a great extent within the "Arbeitsgemeinschaft Mathematisierung" at the TH Darmstadt in which we have worked on the theme of restructuring mathematics now for more than three years. I would like to thank all members of the Arbeitsgemeinschaft for their stimulation, contributions, and continuous support. I would also like to thank all those who have joined discussions at the several lectures which I gave on the subject during the last two years. I am especially grateful to Ivan Rival for his encouragement and help during the writing of the final manuscript. This work was partially supported by the N.S.E.R.C. of Canada under the International Scientific Exchange Programme.

REFERENCES

[1] B. Banaschewski (1956) Hüllensysteme und Erweiterungen von Quasi-Ordnungen, *Z. Math. Logik Grundlagen Math.* 2, 117-130.

[2] G. Birkhoff (1938) Lattices and their applications, *Bull. Amer. Math. Soc.* 44, 793-800.

[3] G. Birkhoff (1967) *Lattice Theory,* Third edition, Amer. Math. Soc., Providence, R.I.

[4] G. Birkhoff (1970) What can lattices do for you? in: *Trends in Lattice Theory* (J.C. Abbott, ed.) Van Nostrand-Reinhold, New York, 1-40.

[5] G. Birkhoff (1982) Ordered sets in geometry, in: *Symp. Ordered Sets* (I. Rival, ed.) Reidel, Dordrecht-Boston, 107.

[6] H.-H. Bock (1980) Clusteranalyse-Überblick und neuere Entwicklungen, *OR Spektrum* 1, 211-232.

[7] P. Crawley and R.A. Dean (1959) Free lattices with infinite operations, *Trans. Amer. Math. Soc.* 92, 35-47.

[8] P. Crawley and R.P. Dilworth (1973) *Algebraic Theory of Lattices*, Prentice-Hall, Englewood Cliffs, N.J.

[9] C.J. Date (1977) *An Introduction to Data Base Systems*, Second edition, Addison-Wesley, Reading, Mass.

[10] R.A. Dean (1956) Completely free lattices generated by partially ordered sets, *Trans. Amer. Math. Soc.* 83, 238-249.

[11] Deutsches Institut für Normung (1979) DIN 2330, *Begriffe und Benennungen, Allgemeine Grundsätze*, Beuth, Köln.

[12] Deutsches Institut für Normung (1980) DIN 2331, *Bergriffs-systeme und ihre Darstellung*, Beuth, Köln.

[13] K. Diem and C. Lentner (1968) Wissenschaftliche Tabellen, 7. Aufl., *J. R. Geigy AG*, Basel.

[14] R.P. Dilworth (1950) A decomposition theorem for partially ordered sets, *Ann. of Math.* (2) 51, 161-166.

[15] G. Grätzer (1978) *General Lattice Theory*, Birkhäuser, Basel-Stuttgart.

[16] G. Grätzer, H. Lakser, and C.R. Platt (1970) Free products of lattices, *Fund. Math.* 69, 233-240.

[17] H. von Hentig (1972) *Magier oder Magister? Über die Einheit der Wissenschaft im Verständigungsprozess*, Klett, Stuttgart.

[18] C.A. Hooker (ed.) (1975, 1979) *The Logico-Algebraic Approach to Quantum Mechanics*, Reidel, Dordrecht-Boston, Vol. I and Vol. II.

[19] B. Jónsson (1962) Arithmetic properties of freely α-generated lattices, *Canad. J. Math.* 14, 476-481.

[20] D.H. Krantz, R.D. Luce, P. Suppes, and A. Tversky (1971) *Foundations of Measurement*, Vol. I, Academic Press, New York.

[21] H. Lakser (1968) *Free Lattices Generated by Partially Ordered Sets*, Ph. D. Thesis, Univ. of Manitoba, Winnipeg.

[22] J.W. Lea (1972) An embedding theorem for compact semilattices, *Proc. Amer. Math. Soc.* 34, 325-331.

[23] H.M. MacNeille (1937) Partially ordered sets, *Trans. Amer. Math. Soc.* 42, 416-460.

[24] H. Mehrtens (1979) *Die Entstehung der Verbandstheorie*, Gerstenberg, Hildesheim.

[25] *Observer's Handbook 1981* (1980) Royal Astronomical Society Canada, Univ. Toronto Press, Toronto.

[26] J. Pflanzagl (1968) *Theory of Measurement*, Physica-Verlag, Würzburg-Wien.

[27] A. Podlech (1981) *Datenerfassung, Verarbeitung, Dokumentation und Information in den sozialärztlichen Diensten mit Hilfe der elektronischen Datenverarbeitung* (manuscript) TH Darmstadt.

[28] H. Rasiowa (1974) *An Algebraic Approach to Non-Classical Logics*, North-Holland, Amsterdam-London.

[29] W. Ritzert (1977) *Einbettung halbgeordneter Mengen in direkte Produkte von Ketten*, Dissertation, TH Darmstad.

[30] I. Rival and R. Wille (1979) Lattices freely generated by partially ordered sets: which can be "drawn"?, *J. reine angew. Math.* 310, 56-80.

[31] F.S. Roberts (1979) *Measurement Theory*, Addison-Wesley, Reading, Mass.

[32] R.J. Rummel (1970) *Applied Factor Analysis*, Northwestern Univ. Press, Evanston.

[33] D.S. Scott (1976) Data types as lattices, *SIAM J. Comput.* 5, 522-587.

[34] J. Schmidt (1956) Zur Kennzeichnung der Dedekind-MacNeilleschen Hülle einer geordneten Menge, *Arch. Math.* 7, 241-249.

[35] E. Schröder (1890, 1891, 1895) *Algebra der Logik I, II, III*, Leipzig.

[36] H. Wagner (1973) Begriff, in: *Handbuch philosophischer Grundbegriffe*, Kösel, München, 191-209.

[37] Ph. M. Whitman (1941) Free lattices, *Ann. of Math.* (2) 42, 325-330.

[38] Ph. M. Whitman (1942) Free lattices, II, *Ann. of Math.* (2) 43, 104-115.

[39] R. Wille (1977) Aspects of finite lattices, in: *Higher Combinatorics* (M. Aigner, ed.) Reidel, Dordrecht-Boston, 79-100.

[40] R. Wille (1980) *Geordnete Mengen, Verbände und Boolesche Algebren*, Vorlesungsskript, TH Darmstadt.

[41] R. Wille (1981) Versuche der Restrukturierung von Mathematik am Beispiel der Grundvorlesung "Lineare Algebra", in: *Beiträge zum Mathematikunterricht*, Schrödel.

Author Index