CLA 2014

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

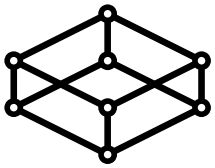
CLA Conference Series

cla.inf.upol.cz

Institute of Computer Science
Pavol Jozef Šafárik University in Košice, Slovakia

ISBN 978–80–8152–159–1

Karell Bertet, Sebastian Rudolph (Eds.)



CLA 2014

Concept Lattices and Their Applications

Volume I

11th International Conference on Concept Lattices and Their Applications
Košice, Slovakia, October 07–10, 2014
Proceedings

P. J. Šafárik University, Košice, Slovakia
2014

Volume editors

Karell Bertet
Université de La Rochelle
La Rochelle, France
E-mail: kbertet@univ-lr.fr

Sebastian Rudolph
Technische Universität Dresden
Dresden, Germany
E-mail: sebastian.rudolph@tu-dresden.de

Technical editor
Sebastian Rudolph, sebastian.rudolph@tu-dresden.de

Cover design
Róbert Novotný, robert.novotny@upjs.sk

ISBN 978–80–8152–159–1

Organization

CLA 2014 was organized by the Institute of Computer Science, Pavol Jozef Šafárik University in Košice.

Steering Committee

Radim Bělohlávek	Palacký University, Olomouc, Czech Republic
Sadok Ben Yahia	Faculté des Sciences de Tunis, Tunisia
Jean Diatta	Université de la Réunion, France
Peter Eklund	University of Wollongong, Australia
Sergei O. Kuznetsov	State University HSE, Moscow, Russia
Engelbert Mephu Nguifo	Université de Clermont Ferrand, France
Amedeo Napoli	LORIA, Nancy, France
Manuel Ojeda-Aciego	Universidad de Málaga, Spain
Jan Outrata	Palacký University, Olomouc, Czech Republic

Program Chairs

Karell Bertet	Université de La Rochelle, France
Sebastian Rudolph	Technische Universität Dresden, Germany

Program Committee

Kira Adaricheva	Nazarbayev University, Astana, Kazakhstan
Cristina Alcalde	Univ del Pais Vasco, San Sebastián, Spain
Jamal Atif	Université Paris Sud, France
Jaume Baixeries	Polytechnical University of Catalonia, Spain
Radim Bělohlávek	Palacký University, Olomouc, Czech Republic
Sadok Ben Yahia	Faculty of Sciences, Tunis, Tunisia
François Brucker	Ecole Centrale Marseille, France
Ana Burusco	Universidad de Navarra, Pamplona, Spain
Claudio Carpineto	Fondazione Ugo Bordoni, Roma, Italy
Pablo Cordero	Universidad de Málaga, Spain
Mathieu D'Aquin	The Open University, Milton Keynes, UK
Christophe Demko	Université de La Rochelle, France
Jean Diatta	Université de la Réunion, France
Florent Domenach	University of Nicosia, Cyprus
Vincent Duquenne	Université Pierre et Marie Curie, Paris, France

Sebastien Ferre	Université de Rennes 1, France
Bernhard Ganter	Technische Universität Dresden, Germany
Alain Gély	Université Paul Verlaine, Metz, France
Cynthia Vera Glodeanu	Technische Universität Dresden, Germany
Robert Godin	Université du Québec à Montréal, Canada
Tarek Hamrouni	Faculty of Sciences, Tunis, Tunisia
Marianne Huchard	LIRMM, Montpellier, France
Céline Hudelot	Ecole Centrale Paris, France
Dmitry Ignatov	State University HSE, Moscow, Russia
Mehdi Kaytoue	LIRIS - INSA de Lyon, France
Jan Konecny	Palacký University, Olomouc, Czech Republic
Marzena Kryszkiewicz	Warsaw University of Technology, Poland
Sergei O. Kuznetsov	State University HSE, Moscow, Russia
Leonard Kwuida	Bern University of Applied Sciences, Switzerland
Florence Le Ber	Strasbourg University, France
Engelbert Mephu Nguifo	Université de Clermont Ferrand, France
Rokia Missaoui	Université du Québec en Outaouais, Gatineau, Canada
Amedeo Napoli	LORIA, Nancy, France
Lhouari Nourine	Université de Clermont Ferrand, France
Sergei Obiedkov	State University HSE, Moscow, Russia
Manuel Ojeda-Aciego	Universidad de Málaga, Spain
Jan Outrata	Palacký University, Olomouc, Czech Republic
Pascal Poncelet	LIRMM, Montpellier, France
Uta Priss	Ostfalia University, Wolfenbüttel, Germany
Olivier Raynaud	LIMOS, Université de Clermont Ferrand, France
Camille Roth	Centre Marc Bloch, Berlin, Germany
Barış Sertkaya	SAP Research Center, Dresden, Germany
Henry Soldano	Laboratoire d'Informatique de Paris Nord, France
Gerd Stumme	University of Kassel, Germany
Laszlo Szathmary	University of Debrecen, Hungary
Petko Valtchev	Université du Québec à Montréal, Canada
Francisco J. Valverde Albacete	Universidad Nacional de Educación a Distancia, Spain

Additional Reviewers

Xavier Dolques	Strasbourg University, France
Philippe Fournier-Viger	University of Moncton, Canada
Michal Krupka	Palacký University, Olomouc, Czech Republic

Organization Committee

Ondrej Krídlo	Pavol Jozef Šafárik University, Košice, Slovakia
Stanislav Krajči	Pavol Jozef Šafárik University, Košice, Slovakia
Ľubomír Antoni	Pavol Jozef Šafárik University, Košice, Slovakia
Lenka Pisková	Pavol Jozef Šafárik University, Košice, Slovakia
Róbert Novotný	Pavol Jozef Šafárik University, Košice, Slovakia

Table of Contents

Preface

Formal Concept Analysis is a mathematical theory formalizing aspects of human conceptual thinking by means of lattice theory. As such, it constitutes a theoretically well-founded, practically proven, human-centered approach to data science and has been continuously contributing valuable insights, methodologies and algorithms to the scientific community.

The International Conference "Concept Lattices and Their Applications (CLA)" is being organized since 2002 with the aim of providing a forum for researchers involved in all aspects of the study of FCA, from theory to implementations and practical applications. Previous years' conferences took place in Horní Bečva, Ostrava, Olomouc (all Czech Republic), Hammamet (Tunisia), Montpellier (France), Olomouc (Czech Republic), Sevilla (Spain), Nancy (France), Fuengirola (Spain), and La Rochelle (France). The eleventh edition of CLA was held in Košice, Slovakia from October 7 to 10, 2014. The event was organized and hosted by the Institute of Computer Science at Pavol Jozef Šafárik University in Košice.

This volume contains the selected papers as well as abstracts of the four invited talks. We received 28 submissions of which 22 were accepted for publication and presentation at the conference. We would like to thank the contributing authors, who submitted high quality works. In addition we were very happy to welcome five distinguished invited speakers: Jaume Baixeries, Hassan Aït-Kassi, Uta Priss, and Ondrej Cepek. All submitted papers underwent a thorough review by members of the Program Committee with the help of additional reviewers. We would like to thank all reviewers for their valuable assistance. A selection of extended versions of the best papers will be published in a renowned journal, pending another reviewing process.

The success of such an event heavily relies on the hard work and dedication of many people. Next to the authors and reviewers, we would also like to acknowledge the help of the CLA Steering Committee, who gave us the opportunity of chairing this edition and provided advice and guidance in the process. Our greatest thanks go to the local Organization Committee from the Institute of Computer Science, Pavol Jozef Šafárik University in Košice, who put a lot of effort into the local arrangements and provided the pleasant atmosphere necessary to attain the goal of providing a balanced event with a high level of scientific exchange. Finally, it is worth noting that we benefited a lot from the EasyChair conference management system, which greatly helped us to cope with all the typical duties of the submission and reviewing process.

October 2014

Karell Bertet
Sebastian Rudolph
Program Chairs of CLA 2014

Relationship between the Relational Database Model and FCA

Jaume Baixeries

Computer Science Department
Universitat Politcnica de Catalunya
Barcelona. Catalonia

The Relational Database Model (RDBM) [3, 4] is one of the most relevant database models that are being currently used to manage data. Although some alternative models are also being used and implemented (namely, object oriented databases and structured datatypes databases or NoSQL databases [1, 2]), the RDBM still maintains its popularity, as some rankings indicate [1].

The RDBM can be formulated from a set-theoretical point of view, such that a tuple is a partial function, and other basic operations in this model such as projections, joins, selections, etc, can be seen as set operations.

Another important feature of this model is the existence of constraints, which are first-order predicates that must hold in a relational database. These constraints mostly describe conditions that must hold in order to keep the consistency of the data in the database, but also help to describe some semantical aspects of the dataset.

In this talk, we consider some aspects of the RDBM that have been characterized with FCA, focusing on different kinds of constraints that appear in the Relational Model. We review some results that formalize different kinds of contraints with FCA [5–8]. We also explain how some concepts of the RDBM such as *key, closure, completion, cover* can be easily be understood with FCA.

References

1. Kai Orend. *Analysis and Classification of NoSQL Databases and Evaluation of their Ability to Replace an Object-relational Persistence Layer*. 2010. doi=10.1.1.184.483
2. A B M Moniruzzaman, Syed Akhter Hossain *NoSQL Database: New Era of Databases for Big data Analytics. Classification, Characteristics and Comparison.* arXiv:1307.0191 [cs.DB]
3. Codd, E. F. *A Relational Model of Data for Large Shared Data Banks.* Commun. ACM, 1970, volume 13, number 6.
4. Date, C. J. *An Introduction to Database Systems (8 ed.)*. Pearson Education. ISBN 0-321-19784-4.
5. Baixeries, Jaume. A Formal Context for Symmetric Dependencies. ICFCA 2008. LNAI 4933.
6. Baixeries, Jaume and Balcázar, José L. Characterization and Armstrong Relations for Degenerate Multivalued Dependencies Using Formal Concept Analysis. Formal Concept Analysis, Third International Conference, ICFCA 2005, Lens, France, February 14-18, 2005, Proceedings. Lecture Notes in Computer Science, 2005

[1] http://db-engines.com/en/ranking

7. Baixeries, Jaume and Balcázar, José L. Unified Characterization of Symmetric Dependencies with Lattices. Contributions to ICFCA 2006. 4th International Conference on Formal Concept Analysis 2005.
8. Baixeries, Jaume. A Formal Concept Analysis framework to model functional dependencies. Mathematical Methods for Learning, 2004.

What Formalism for the Semantic Web?

Hassan Aït-Kaci

hassan.ait-kaci@univ-lyon1.fr

ANR Chair of Excellence
$\mathcal{CEDAR}$ Project
LIRIS
Université Claude Bernard Lyon 1
France

The world is changing. The World Wide Web is changing. It started out as a set of purely notational conventions for interconnecting information over the Internet. The focus of information processing has now shifted from local disconnected disc-bound silos to Internet-wide interconnected clouds. The nature of information has also evolved. From raw uniform data, it has now taken the shape of semi-structured data and meaning-carrying so-called "Knowledge Bases." While it was sufficient to process raw data with structure-aware querying, it has now become necessary to process knowledge with contents-aware reasoning. Computing must therefore adapt from dealing with mere *explicit* data to inferring *implicit* knowledge. How to represent such knowledge and how inference therefrom can be made effective (whether reasoning or learning) is thus a central challenge among the many now facing the world wide web.

So called "ontologies" are being specified and meant to encode formally encyclopedic as well as domain-specific knowledge. One early (still on-going) such effort has been the Cyc[1] system. It is a knowledge-representation system (using LISP syntax) that makes use of a set of varied reasoning methods, altogether dubbed "commonsense." A more recent formalism issued of Description Logic (DL)—viz. the Web Ontology Language (OWL[2])—has been adopted as a W3C recommendation. It encodes knowledge using a specific standardized (XML, RDF) syntax. Its constructs are given a model-theoretic semantics which is usually realized operationally using tableau[3]-based reasoning.[4] The point is that OWL is clearly designed for a specific logic and reasoning method. Saying that OWL is the most adequate interchange formalism for Knowledge Representation (KR) and automated reasoning (AR) is akin to saying that English is the best designed human language for facilitating information interchange among humans—notwithstanding the fact that it was simply imposed by the most recent pervasive ruling power, just as Latin was Europe's *Lingua Franca* for centuries.

Thus, it is fair to ask one's self a simple question: "*Is there, indeed, a single most adequate knowledge representation and reasoning method that can be such a norm?*"

---

[1] http://www.cyc.com/platform/opencyc

[2] http://www.w3.org/TR/owl-features/

[3] http://en.wikipedia.org/wiki/Method_of_analytic_tableaux

[4] Using of tableau methods is the case of the most prominent SW reasoner [6, 5, 7]. Systems using alternative reasoning methods must first translate the DL-based syntax of OWL into their own logic or RDF query processing. This may be costly [9] and/or incomplete [8].

I personally do not think so. In this regard, I share the general philosophy of Doug Lenat[5], Cyc's designer—although not the haphazard approach he has chosen to follow.[6]

If one ponders what characterizes an ontology making up a knowledge base, some specific traits most commonly appear. For example, it is universally acknowledged that, rather than being a general set of arbitrary formal logical statements describing some generic properties of "the world," a formal knowledge base is generally organized as a concept-oriented information structure. This is as important a change of perspective, just as object-oriented programming was with respect to traditional method-oriented programming. Thus, some notion of property "inheritance" among partially-ordered "concepts" (with an *"is-a"* relation) is a characteristic aspect of KR formalisms. In such a system, a concept has a straightforward semantics: its denotes of set of elements (its "instances") and the "is-a" relation denotes set inclusion. Properties attached to a concept denote information pertaining to all instances of this concept. All properties verified by a concept are therefore *inherited* by all its subconcepts.

Sharing this simple characteristic, formal KR formalisms have emerged from symbolic mathematics that offer means to reason with conceptual information, depending on mathematical apparatus formalizing inheritance and the nature of properties attached to concepts. In Description Logic[7], properties are called "roles" and denote binary relations among concepts. On the other hand, Formal Concept Analysis (FCA[8]) uses an algebraic approach whereby an "is-a" ordering is automatical derived from propositional properties encoding the concepts that are attached to as bit vectors. A concept is associated an attribute with a boolean marker (1 or "true") if it possesses it, and with a (0 or "false") otherwise. The bit vectors are simply the rows of the "property matrix" relating concepts to their attributes. This simple and powerful method, originally proposed by Rudolf Wille, has a dual interpretation when matching attributes with concepts possessing them. Thus, dually, it views attributes also as partially ordered (as the columns of the binary matrix). An elegant Galois-connection ensues that enables simple extraction of conceptual taxonomies (and their dual attribute-ordered taxonomies) from simple facts. Variations such as Relational Concept Analysis (RCA[9]) offer more expressive, and thus more sophisticated, knowledge while preserving the essential algebraic properties of FCA. It has also been shown how DL-based reasoning (e.g. OWL) can be enhanced with FCA.[10]

Yet another formalism for taxonomic attributed knowledge, which I will present in more detail in this presentation, is the Order-Sorted Feature ($\mathcal{OSF}$) constraint formalism. This approach proposes to see everything as an order-sorted labelled graph.

[5] `http://en.wikipedia.org/wiki/Douglas_Lenat`

[6] However, I may stand corrected in the future since knowledge is somehow fundamentally haphazard. My own view is that, even for dealing with a heterogenous world, I would rather favor mathematically *formal* representation and reasoning methods dealing with uncertainty and approximate reasoning, whether probabilistic, fuzzy, or dealing with inconsistency (e.g. rough sets, paraconsistency).

[7] `http://en.wikipedia.org/wiki/Description_logic`

[8] `http://en.wikipedia.org/wiki/Formal_concept_analysis`

[9] `http://www.hse.ru/data/2013/07/04/1286082694/ijcai_130803.pdf`

[10] `http://ijcai-11.iiia.csic.es/files/proceedings/T13-ijcai11Tutorial.pdf`

Sorts are set-denoting and partially ordered with an inclusion-denoting "is-a" relation, and so form a conceptual taxonomy. Attributes, called "*features*," are function-denoting symbols labelling directed edges between sort-labelled nodes. Such $\mathcal{OSF}$ graphs are a straightforward generalization of algebraic First-Order Terms (FOTs) as used in Logic Programming (LP) and Functional Programming (FP). Like FOTs, they form a lattice structure with $\mathcal{OSF}$ graph *matching* as the partial ordering, $\mathcal{OSF}$ graph unification as infimum (denoting set intersection), and $\mathcal{OSF}$ graph generalization as supremum.[11] Both operations are very efficient. These lattice-theoretic properties are preserved when one endows a concept in a taxonomy with additional order-sorted relational and functional constraints (using logical conjunction for unification and disjunction for generalization for the attached constraints). These constraints are inherited down the conceptual taxonomy in such a way as to be incrementally enforceable as a concept becomes gradually refined.

The $\mathcal{OSF}$ system has been the basis of Constraint Logic Programming for KR and ontological reasoning (viz. $\mathcal{LIFE}$) [2, 1]. As importantly, $\mathcal{OSF}$ graph-constraint technology has been at work with great success in two essential areas of AI: NLP and Machine Learning:

- it has been a major paradigm in the field of Natural Language Processing (NLP) for a long time; notably, in so-called "Head-driven Phrase Structure Grammar" (HPSG[12]) and Unification Grammar (UG[13]) technology [4]. This is indeed not surprising given the ease with which feature structure unification enables combining both syntactic and semantic information in a clean, declarative, and efficient way.[14]
- Similarly, while most of the attention in the $\mathcal{OSF}$ literature has been devoted to unification, its dual—namely, generalization—is just as simple to use, and computes the most specific $\mathcal{OSF}$ term that subsumes two given terms [3]. This operation is central in Machine Learning and with it, $\mathcal{OSF}$ technology lends itself to be combined with popular Data Mining techniques such as Support Vector Machines using frequency or probabilistic information.

In this presentation, I will give a rapid overview of the essential $\mathcal{OSF}$ formalism for knowledge representation along its reasoning method which is best formalized as order-sorted constraint-driven inference. I will also illustrate its operational efficiency and scalability in comparison with those of prominent DL-based reasoners used for the Semantic Web.

The contribution of this talk to answering the question in its title is that the Semantic Web effort should not impose *a priori* putting all our eggs in one single (untested) basket. Rather, along with DL, other viable alternatives such as the FCA and $\mathcal{OSF}$ formalisms, and surely others, should be combined for realizing a truly *semantic* web.

---

[11] This supremum operation, however, does not (always) denote set union—as for FOT subsumption, it is is *not* modular (and hence neither is it distributive).

[12] `http://en.wikipedia.org/wiki/Head-driven_phrase_structure_grammar`

[13] `http://www.cs.haifa.ac.il/~shuly/malta-slides.pdf`

[14] `http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.51.2021`

References

1. AÏT-KACI, H. Data models as constraint systems—a key to the Semantic Web. *Constraint Processing Letters 1* (November 2007), 33–88. online: `http://cs.brown.edu/people/pvh/CPL/Papers/v1/hak.pdf`.
2. AÏT-KACI, H., AND PODELSKI, A. Towards a meaning of LIFE. *Journal of Logic Programming 16*, 3-4 (1993), 195–234. online: `http://hassan-ait-kaci.net/pdf/meaningoflife.pdf`.
3. AÏT-KACI, H., AND SASAKI, Y. An axiomatic approach to feature term generalization. In *Proceedings of European Cinference on Machine Learning (ECML 2001)* (Freiburg, Germany, 2001), L. D. Raedt and P. Flach, Eds., LNAI 2167, Springer-Verlag, pp. 1–12. online: `http://www.hassan-ait-kaci.net/pdf/ecml01.pdf`.
4. CARPENTER, B. Typed feature structures: A generalization of first-order terms. In *Proceedings of the 1991 International Symposium on Logic Programming* (Cambridge, MA, USA, 1991), V. Saraswat and K. Ueda, Eds., MIT Press, pp. 187–201.
5. MOTIK, B., SHEARER, R., AND HORROCKS, I. Hypertableau reasoning for description logics. *Journal of Artificial Intelligence Research 36*, 1 (September 2009), 165–228. online: `https://www.jair.org/media/2811/live-2811-4689-jair.pdf`.
6. SHEARER, R., MOTIK, B., AND HORROCKS, I. HermiT: A highly-efficient OWL reasoner. In *Proceedings of the 5th International Workshop on OWL Experiences and Directions* (Karlsruhe, Germany, October 2008), U. Sattler and C. Dolbear, Eds., OWLED'08, CEUR Workshop Proceedings. online: `http://www.cs.ox.ac.uk/ian.horrocks/Publications/download/2008/ShMH08b.pdf`.
7. SIRIN, E., PARSIA, B., GRAU, B. C., KALYANPUR, A., AND KATZ, Y. Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics 5*, 2 (June 2007), 51–53. This is a summary; full paper: online: `http://pellet.owldl.com/papers/sirin05pellet.pdf`.
8. STOILOS, G., CUENCA GRAU, B., AND HORROCKS, I. How incomplete is your semantic web reasoner? In *Proceedings of the 24th National Conference on Artificial Intelligence (AAAI 10)* (Atlanta, Georgia, USA, July 11–15, 2010), M. Fox and D. Poole, Eds., AAAI, AAAI Publications, pp. 1431–1436. online: `http://www.cs.ox.ac.uk/ian.horrocks/Publications/download/2010/StCH10a.pdf`.
9. THOMAS, E., PAN, J. Z., AND REN, Y. TrOWL: Tractable OWL 2 reasoning infrastructure. In *Proceedings of the 7th Extended Semantic Web Conference* (Heraklion, Greece, May-June 2010), L. Aroyo, G. Antoniou, E. Hyvnen, A. ten Teije, H. Stuckenschmidt, L. Cabral, and T. Tudorache, Eds., ESWC'10, Springer-Verlag, pp. 431–435. online: `http://homepages.abdn.ac.uk/jeff.z.pan/pages/pub/TPR2010.pdf`.

Linguistic Data Mining with FCA

Uta Priss

ZeLL, Ostfalia University of Applied Sciences
Wolfenbüttel, Germany
www.upriss.org.uk

The use of lattice theory for linguistic data mining applications in the widest sense has been independently suggested by different researchers. For example, Masterman (1956) suggests using a lattice-based thesaurus model for machine translation. Mooers (1958) describes a lattice-based information retrieval model which was included in the first edition of Salton's (1968) influential textbook. Sladek (1975) models word fields with lattices. Dyvik (2004) generates lattices which represent mirrored semantic structures in a bilingual parallel corpus. These approaches were later translated into the language of Formal Concept Analysis (FCA) in order to provide a more unified framework and to generalise them for use with other applications (Priss (2005), Priss & Old (2005 and 2009)).

Linguistic data mining can be subdivided into syntagmatic and paradigmatic approaches. Syntagmatic approaches exploit syntactic relationships. For example, Basili et al. (1997) describe how to learn semantic structures from the exploration of syntactic verb-relationships using FCA. This was subsequently used in similar form by Cimiano (2003) for ontology construction, by Priss (2005) for semantic classification and by Stepanova (2009) for the acquisition of lexico-semantic knowledge from corpora.

Paradigmatic relationships are semantic in nature and can, for example, be extracted from bilingual corpora, dictionaries and thesauri. FCA neighbourhood lattices are a suitable means of mining bilingual data sources (Priss & Old (2005 and 2007)) and monolingual data sources (Priss & Old (2004 and 2006)). Experimental results for neighbourhood lattices have been computed for Roget's Thesaurus, WordNet and Wikipedia data (Priss & Old 2006, 2010a and 2010b).

Previous overviews of linguistic applications of FCA were presented by Priss (2005 and 2009). This presentation summarises previous results and provides an overview of more recent research developments in the area of linguistic data mining with FCA.

References

1. Basili, R.; Pazienza, M.; Vindigni, M. (1997). *Corpus-driven unsupervised learning of verb subcategorization frames.* AI*IA-97.
2. Cimiano, P.; Staab, S.; Tane, J. (2003). *Automatic Acquisition of Taxonomies from Text: FCA meets NLP.* Proceedings of the ECML/PKDD Workshop on Adaptive Text Extraction and Mining, p. 10-17.
3. Dyvik, H. (2004). *Translations as semantic mirrors: from parallel corpus to wordnet.* Language and Computers, 49, 1, Rodopi, p. 311-326.

4. Masterman, Margaret (1956). *Potentialities of a Mechanical Thesaurus.* MIT Conference on Mechanical Translation, CLRU Typescript. [Abstract]. In: Report on research: Cambridge Language Research Unit. Mechanical Translation 3, 2, p. 36. Full paper in: Masterman (2005).
5. Mooers, Calvin N. (1958). *A mathematical theory of language symbols in retrieval.* In: Proc. Int. Conf. Scientific Information, Washington D.C.
6. Priss, Uta; Old, L. John (2004). *Modelling Lexical Databases with Formal Concept Analysis.* Journal of Universal Computer Science, 10, 8, p. 967-984.
7. Priss, Uta (2005). *Linguistic Applications of Formal Concept Analysis.* In: Ganter; Stumme; Wille (eds.), Formal Concept Analysis, Foundations and Applications. Springer Verlag. LNAI 3626, p. 149-160.
8. Priss, Uta; Old, L. John (2005). *Conceptual Exploration of Semantic Mirrors.* In: Ganter; Godin (eds.), Formal Concept Analysis: Third International Conference, ICFCA 2005, Springer Verlag, LNCS 3403, p. 21-32.
9. Priss, Uta; Old, L. John (2006). *An application of relation algebra to lexical databases.* In: Schaerfe, Hitzler, Ohrstrom (eds.), Conceptual Structures: Inspiration and Application, Proceedings of the 14th International Conference on Conceptual Structures, ICCS'06, Springer Verlag, LNAI 4068, p. 388-400.
10. Priss, Uta; Old, L. John (2007). *Bilingual Word Association Networks.* In: Priss, Polovina, Hill (eds.), Proceedings of the 15th International Conference on Conceptual Structures, ICCS'07, Springer Verlag, LNAI 4604, p. 310-320.
11. Priss, Uta (2009). *Formal Concept Analysis as a Tool for Linguistic Data Exploration.* In: Hitzler, Pascal; Scharfe, Henrik (eds.), Conceptual Structures in Practice, Chapman & Hall/CRC studies in informatics series, p. 177-198.
12. Priss, Uta; Old, L. John (2009). *Revisiting the Potentialities of a Mechanical Thesaurus.* In: Ferre; Rudolph (eds.), Proceedings of the 7th International Conference on Formal Concept Analysis, ICFCA'09, Springer Verlag.
13. Priss, Uta; Old, L. John (2010a). *Concept Neighbourhoods in Knowledge Organisation Systems.* In: Gnoli; Mazzocchi (eds.), Paradigms and conceptual systems in knowledge organization. Proceedings of the 11th International ISKO Conference, p. 165-170.
14. Priss, Uta; Old, L. John (2010b). *Concept Neighbourhoods in Lexical Databases.* In: Kwuida; Sertkaya (eds.), Proceedings of the 8th International Conference on Formal Concept Analysis, ICFCA'10, Springer Verlag, LNCS 5986, p. 283-295.
15. Salton, Gerard (1968). *Automatic Information Organization and Retrieval.* McGraw-Hill, New York.
16. Stepanova, Nadezhda A. (2009). *Automatic acquisition of lexico-semantic knowledge from corpora.* SENSE'09 Workshop. Available at http://ceur-ws.org/Vol-476/.
17. Sladek, A. (1975). *Wortfelder in Verbänden.* Gunter Narr Verlag, Tübingen.

Shortest CNF Representations of Pure Horn Functions and their Connection to Implicational Bases

Ondrej Cepek

Charles University, Prague

Pure Horn CNFs, directed hypergraphs, and closure systems are objects studied in different subareas of theoretical computer science. Nevertheless, these three objects are in some sense isomorphic. Thus also properties derived for one of these objects can be usually translated in some way for the other two. In this talk we will concentrate on the problem of finding a shortest CNF representation of a given pure Horn function. This is a problem with many practical applications in artificial intelligence (knowledge compression) and other areas of computer science (e.g. relational data bases). In this talk we survey complexity results known for this problem and then concentrate on the relationships between CNF representations of Horn functions and certain sets of implicates of these functions, called essential sets of implicates. The definition of essential sets is based on the properties of resolution. Essential sets can be shown to fulfill an interesting orthogonality property: every CNF representation and every (nonempty) essential set must intersect. This property leads to non-trivial lower bounds on the CNF size, which are sometimes tight and sometimes have a gap. We will try to derive connections to the known properties of minimal implicational bases.

The talk is based on joint research with Endre Boros, Alex Kogan, Petr Kucera, and Petr Savicky.

Learning Model Transformation Patterns using Graph Generalization

Hajer Saada, Marianne Huchard, Michel Liquière, and Clémentine Nebut

LIRMM, Université de Montpellier 2 et CNRS, Montpellier, France,
first.last@lirmm.fr

Abstract. In Model Driven Engineering (MDE), a Model Transformation is a specialized program, often composed of a set of rules to transform models. The Model Transformation By Example (MTBE) approach aims to assist the developer by learning model transformations from source and target model examples.In a previous work, we proposed an approach which takes as input a fragmented source model and a target model, and produces a set of fragment pairs that presents the many-to-many matching links between the two models. In this paper, we propose to mine model transformation patterns (that can be later transformed in transformation rules) from the obtained matching links. We encode our models into labeled graphs that are then classified using the GRAAL approach to get meaningful common subgraphs. New transformation patterns are then found from the classification of the matching links based on their graph ends. We evaluate the feasibility of our approach on two representative small transformation examples.

1 Introduction

MDE is a subfield of software engineering that relies on models as a central artifact for the software development cycle. Models can be manually or automatically manipulated using model transformations. A model transformation is a program, often composed of a set of transformation rules, that takes as input a model and produces as output another model. The models conform to meta-models, as programs conform to the programming language grammar. If we would like to transform any Java program into any C++ program, we would express this transformation at the level of their grammars. In MDE, model transformations are similarly expressed in terms of the meta-models. Designing a model transformation is thus a delicate issue, because the developer has to master the specialized language in which the transformation is written, the meta-modeling activity, and the subtleties of the source and the target meta-models. In order to assist the developers, the MTBE approach follows the track of the "Programming By Example" approach [6] and proposes to use an initial set of transformation examples from which the model transformation is partly learnt. The first step of the MTBE approach consists in extracting matching links, from which the second step learns transformation rules. Several approaches [1,15,12] are proposed for the second step, but they derive element-to-element (one-to-one) rules

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

that mainly express how a source model element is transformed into a target model element. In this paper, we propose to learn transformation patterns of type fragment-to-fragment (many-to-many) using the output of a previous work [13] that consists in generating matching links between source and target model fragments. We encode our models and model fragments as labeled graphs. These graphs are classified through a lattice using a graph mining approach (GRAAL) to get meaningful common subgraphs. The matching links are then classified using Formal Concept Analysis, the lattice of source graphs and the lattice of target graphs. New transformation patterns are discovered from these classifications that can help the designer of the model transformation. We evaluate the feasibility of our approach on two representative transformation examples.

The next Section 2 gives an overview of our approach. Section 3 presents the transformation pattern mining approach and Section 4 evaluates its feasibility. Section 5 presents the related work, and we conclude in Section 6.

2 Approach Overview

In Model-Driven Engineering, model transformation are programs that transform an input source model into an output target model. A classical model transformation (`UML2REL`) transforms a UML model into a RELational model. Such transformation programs are often written with declarative or semi-declarative languages and composed of a set of transformation rules, defined at the meta-model level. The meta-model defines the concepts and the relations that are used (and instantiated) to compose the models. For example, the UML meta-model

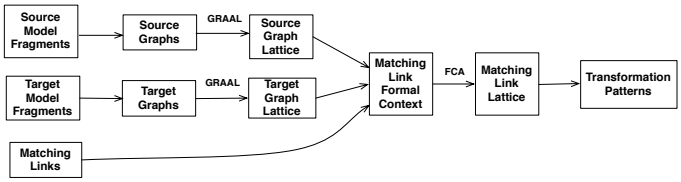


Fig. 1. Process overview

contains the concept of *Class* which owns *Attributes*. This can used to derive a UML model composed of a class *Person* owning the attribute *Name*. In the `UML2REL` example, a very simple transformation pattern would be: *a UML class owning an attribute is transformed into a relational table owning a column*. In this paper, our objective is to learn such transformation patterns that express that a pattern associating entities of the source meta-model (*e.g. a UML class owning an attribute*) is transformed into a pattern associating entities of the target meta-model (*e.g. a relational table owning a column*). Fig. 1 provides an overview of our process. Let us consider that we want to learn rules for

transforming UML models to relational models. Our input data (see Fig. 2) are composed of: fragmented source models (a UML source fragment is given in Fig. 3(a)); fragmented target models (a relational target fragment is given in Fig. 3(b)); and matching links between fragments established by experts or by an automatic matching technique. For example a matching link (`L1`) is established in Fig. 2 between the UML source fragment of Fig. 3(a) and the relational target fragment of Fig. 3(b).

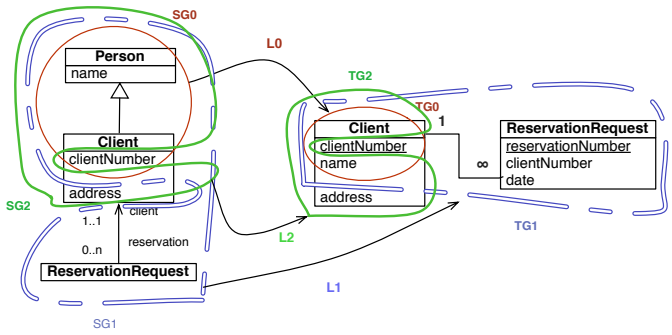
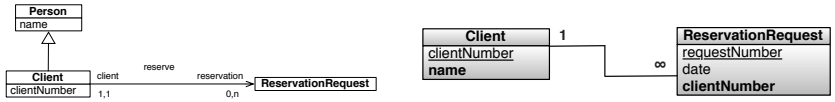


Fig. 2. Three matching links between fragmented UML and relational models (this figure is more readable in a coloured version)

Matching links established by experts or by automatic methods can be used to form a set of model transformation patterns. For example, the `L2` matching link gives rise to a transformation pattern which indicates that *a UML class (with an attribute) with its super-class (with an attribute) is transformed into a unique table with two columns, one being inherited.* Nevertheless, matching links often correspond to patterns that combine several simpler transformations or are triggered from domain knowledge. Besides, they may contain minor errors (such as a few additional or missing elements, for example, column `date` of Table `ReservationRequest` has in fact no equivalent in Class `ReservationRequest`). Moreover, what interests us is beyond the model domain. We do not want to learn that Class `Client` is transformed into Table `Client`, but rather that a UML class is usually transformed into a table.

Our output is composed of a set of model transformation patterns. Some can directly be inferred from initial matching links (as evoked previously), and some will be found thanks to graph generalization and matching link classification. From our simple example, we want to extract the model transformation pattern presented in Figure 4, whose premise and conclusion patterns do not appear as such in the initial set of matching links ($\hookrightarrow$ means "is transformed into").



(a) UML source fragment (b) Relational target fragment

Fig. 3. An example of UML and relational models

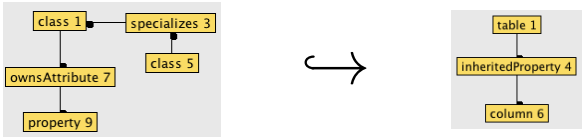


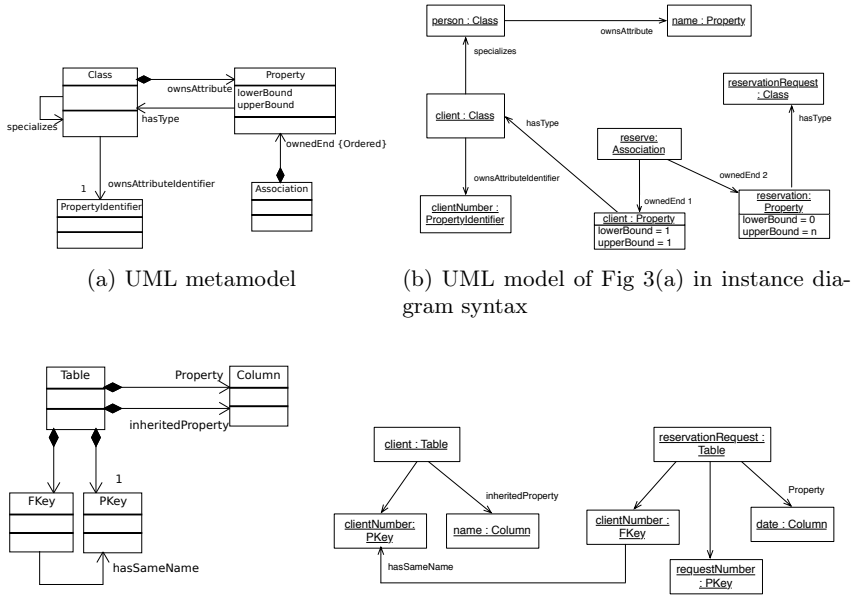
Fig. 4. Transformation pattern: a class specializing a class with an attribute (in UML model) is transformed into a table with an inherited column (in relational model).

3 Model Transformation Pattern Generation

From model fragments to graphs For our example, the source meta-model is inspired by a tiny part of the UML metamodel (see Figure 5(a)), while the target meta-model has its roots in a simplified relational data-base meta-model (see Fig. 5(c)). The models often are represented in a *visual* syntax (as shown in Fig. 3(a) and Fig. 3(b)) for readability reasons. Here we use their representation as instance diagrams of their meta-model (using the UML instance diagram syntax). For example, the UML model of Fig. 3(a) is shown as an instantiation of its meta-model in Fig. 5(b), where each object (in a rectangular box) is described by its meta-class in the meta-model, valued attributes and roles conforming to the attributes and associations from the meta-model: *e.g.* `person` and `client` are explicit instances of `Class`; `client:Class` has a link towards `person` labeled by the role `specializes`; `client:Property` has the attribute `lowerBound` (1).

To extract expressive transformation patterns, we transform our models using their instance diagram syntax, into simpler graphs which have labeled vertices. We limited ourselves to locally injective labelled graphs. A locally injective graph is a labeled graph such that all vertices in the neighbor of a given vertex have different labels. This is not so restrictive in our case, because the fragments identified by the experts rarely include similar neighborhood for an entity. Here are the rules that we use in the transformation from simplified UML instance diagrams to labeled graphs. We associate a labeled node to Objects, Roles, Attributes, Attribute values. Instance diagram of Figure 5(b) and corresponding labeled graph from 6(a) are used to illustrate the transformation: `person:Class` object is transformed into node 1 labeled `class_1` and one of the attribute value `1` is transformed into node 13 labeled `one_13`. Edges come from the following situations: an object has an attribute; an attribute has a value; an object has

a role (is source of a role); an object is the target of a role. For example, for the property which has an attribute lowerBound (equal to `zero`), there is a corresponding edge `(property_17, lowerBound_18)`.



(a) UML metamodel

(b) UML model of Fig 3(a) in instance diagram syntax

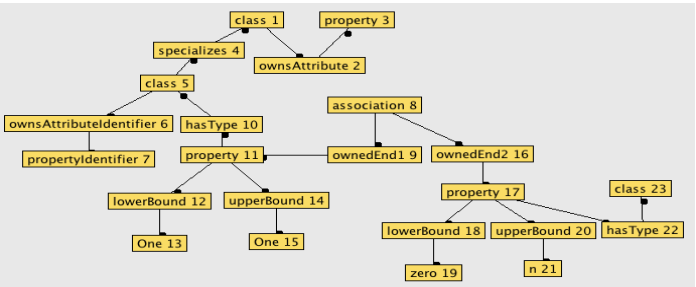
(c) Relational metamodel

(d) Relational model of Fig 3(b) in instance diagram syntax

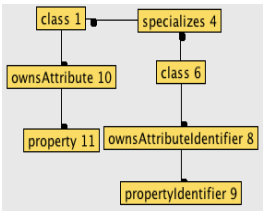
Fig. 5. Source/target metamodel and model, UML (upper par), relational (lower part)

Classification of graphs (GRAAL approach) After the previous step, we obtain a set of source graphs, and a set of target graphs. We illustrate the remainder of this section by using the three source graphs of Fig. 6, the three target graphs of Fig. 7, and the matching links (Source graph i, Target graph i), for i $\in \{0, 1, 2\}$. To get meaningful common subgraphs (on which new transformation patterns will be discovered), we use the graph mining approach proposed in [7] and its derived GRAAL tool. In this approach, examples are described by a description language L provided with two operations: an $\leq$ specialization operation and an $\otimes$ operation which builds the least general generalization of two descriptions. A generalization of the Norris algorithm [11] builds the Galois lattice. Several description languages are implemented in GRAAL, and especially a description based on locally injective graphs. $\otimes$ operation is the reduction of the tensor product of graphs, also called the Kronecker product [14]. We independently classify source graphs and target graphs. Classification of source graphs

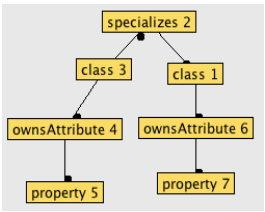
produces the lattice of Fig. 8(a). For example, in this lattice, Concept `sfc012` has for intent a subgraph of source graphs 0, 1 and 2 representing a class which specializes a class which owns an attribute. Classification of target graphs produces the lattice of Fig. 8(b). In this lattice, Concept `tfc012` has for intent a subgraph where a table has an inherited property.



(a) Source graph 1



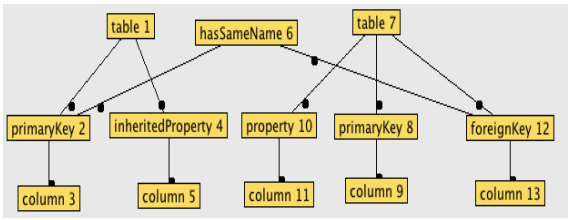
(b) Source graph 0



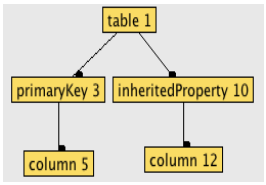
(c) Source graph 2

Fig. 6. Source graphs

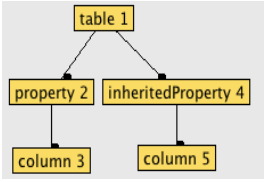
Classification of transformation links In the previous section, we have shown how Galois lattices can be computed on the labeled graphs that represent our model fragments. Now a matching link is described by a pair composed of a source fragment (whose corresponding graph is in the extent of some concepts in the source graph lattice) and a target fragment (whose corresponding graph is in the extent of some concepts in the source graph lattice). This is described in a formal context, where objects are the matching links and attributes are the concepts of the two lattices (source graph lattice and target graph lattice). In this formal context (presented in Table 11(a)), a matching link is associated with the concepts having respectively its source graph and its target graph in their extent. This means that the matching link is described by the graph of its source fragment and by the generalizations of this graph in the lattice. This is the same for the graphs of the target fragments. For example, matching link `L0`, connecting source fragment 0 to target fragment 0, is associated in the formal context to concepts `sfc01`, `sfc012`, `tfc01`, `tfc012`.



(a) Target graph 1

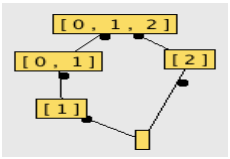


(b) Target graph 0

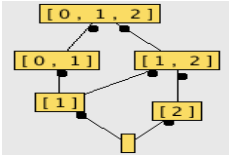


(c) Target graph 2

Fig. 7. Target graphs



(a) Source graph lattice



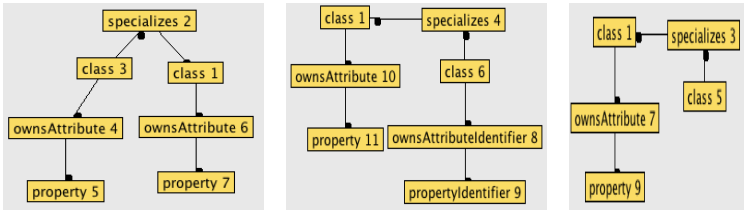
(b) Target graph lattice

Fig. 8. Graph lattices. Only concept extents are represented in the figure. Intents of concepts are shown in Fig. 9 and 10. We denote by $sfcx_1...x_n$ (resp. $tfcx_1...x_n$) the vertex [x_1,...,x_n] of the source (resp. target) graph lattice.

The concept lattice associated with the matching link formal context of Fig. 11(a) is shown in Fig. 11(b). In this representation (obtained with RCAexplore[1]) each box describes a concept: the first compartment informs about the name of the concept, the second shows the simplified intent (here concepts from source fragment lattice and target fragment lattice) and the third one shows the simplified extent (here matching links). `Concept_MatchingLinksFca_4` extent is composed of the links `L0` and `L1`, while the intent is composed of source graph concepts `sfc01, sfc012` and target graph concepts `tfc01, tfc012`.

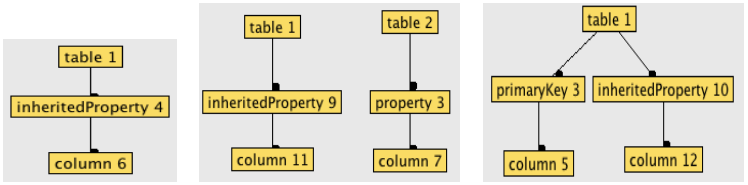
Model transformation pattern mining The last step of the process consists in extracting model transformation patterns from the matching link lattice. This has close connections to the problem of extracting implication rules in a concept lattice, but using only pairs of source and target graph concepts. The more

[1] http://dolques.free.fr/rcaexplore.php



(a) Concept `sfc2` (b) Concept `sfc01` (c) Concept `sfc012`

Fig. 9. Source graph lattice concepts. Concept `sfc1` (not represented) has Source Graph 1 of Fig. 6 as intent



(a) Concept `tfc012` (b) Concept `tfc12` (c) Concept `tfc01`

Fig. 10. Target graph lattice concepts. Concepts `tfc1` and `tfc2` (not represented) have resp. Target Graph 1 and Target Graph 2 from Fig. 7 as their intents.

reliable transformation patterns are given when using a source graph and a target graph in the same simplified intent of a concept, because this corresponds to the fact that the source graph is always present when the target graph is present too (and reversely). For example, from `Concept_MatchingLinksFca_0`, we obtain the following transformation pattern:

$$\textit{graph of } \texttt{sfc012} \textit{ intent} \hookrightarrow \textit{graph of } \texttt{tfc012} \textit{ intent}$$

This pattern expresses a new transformation pattern (*new* in the sense that it does not directly come from a matching link):

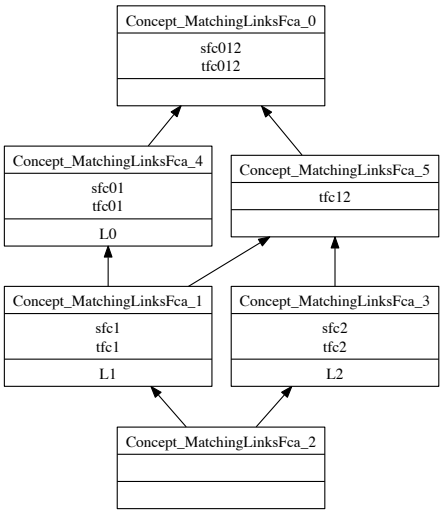
> *A UML model where a class C_d specializes another class C_m which owns an attribute a is transformed into a relational model where a table T owns a (inherited) column c.*

Due to the simplicity of our illustrative example, the other reliable patterns obtained from source and target graphs from the same simplified intent just correspond to matching links.

Obtaining other, less reliable patterns, relies on the fact that if a source graph and a target graph are not in the same simplified intent, but the concept

ML	sfc1	sfc2	sfc01	sfc012	tfc1	tfc2	tfc01	tfc12	tfc012
L0			×	×			×		×
L1	×		×	×	×		×	×	×
L2		×		×		×		×	×

(a) Matching Link Formal Context $MLFC$



(b) Matching Link Lattice

Fig. 11. Matching link formal context and corresponding concept lattice.

C_s which introduces the source graph is below the concept C_t which introduces the target graph, then we infer the following transformation pattern:

$$part\ of\ graph\ of\ C_s\ intent \hookrightarrow graph\ of\ C_t\ intent$$

For example, as `sfc1` appears below `tfc12`, we can deduce that, when the input of the transformation contains the graph of intent of`sfc1`, thus the output contains the graph of intent of `tfc12`. These patterns are less reliable, because the source graph may contain many things that have nothing to do with the target graph (compare `sfc1` and `tfc12` to see this phenomenon). However, experts can have a look on these patterns to find several (concurrent) transformation patterns when several source model fragments are transformed into a same target model fragment. We have a symmetric situation when a source graph and a target graph are not in the same simplified intent, but the concept C_t which introduces the target graph is below the concept C_s which introduces the source graph.

4 Feasibility study

We evaluated the feasibility of the approach on two different realistic transformation examples: (1) UML class diagram to relational schema model that contains

108 model elements, 10 fragments (5 sources, 5 targets) and 5 matching links (U2S) and (2) UML class diagram to entity relationship model that contains 66 model elements, 6 fragments (3 sources, 3 targets) and 3 matching links (U2E). We compute from the obtained graphs for each transformation example several pattern categories (see left-hand side of Table 1).

(1) The transformation patterns coming from simplified intents (which we think are the most relevant patterns): they correspond to graphs pairs (G_S, G_T) such that G_S and G_T are in the simplified intent of a same concept. They can be divided into two sets. The set TP_l groups the patterns that are inferred from the initial matching links (G_S, G_T are the ends of a matching link). The set TP_n contains the patterns that are learned from graph generalization and matching link classification.

(2) The transformation patterns ($TP_{n_{part_s}}$) coming from the graphs G_S and G_T, such that G_S is in simplified intent of a concept C_s which is a subconcept of the concept C_t which has G_T in its simplified intent and all concepts greater than C_s and lower than C_t have an empty simplified intent. In addition, we consider only the case where simplified intent of C_s contains only source graphs or (inclusively) simplified intent of C_t contains only target graphs.

(3) Symmetrically, the transformation patterns ($TP_{n_{part_t}}$) coming from the graphs G_S and G_T, such that G_T is in simplified intent of a concept C_t which is a subconcept of the concept C_s which has G_S in its simplified intent and all concepts greater than C_t and lower than C_s have an empty simplified intent. In addition, we consider only the case where simplified intent of C_s contains only source graphs or (inclusively) simplified intent of C_t contains only target graphs.

Table 1. Results. Left-hand side: sets cardinals. Right-hand side: precision metrics

	$\#TP_l$	$\#TP_n$	$\#TP_{n_{part_s}}$	$\#TP_{n_{part_t}}$
Ill. ex.	2	2	2	1
U2S	1	5	3	0
U2E	2	2	1	1

	P_{TP_l}	P_{TP_n}	$P_{TP_{n_{part_s}}}$	$P_{TP_{n_{part_t}}}$
Ill. ex.	1	1	0.72	0.72
U2S	1	0.75	0.78	-
U2E	1	1	0.73	0.95

We also evaluate each extracted transformation pattern using a precision metric. Precision here is the number of elements in the source and target graphs that participate correctly to the transformation (according to a human expert) divided by the number of elements in the graphs. We then associate a precision measure to a set of transformation patterns, which is the average of the precisions of its elements (See right-hand side of Table 1).

The results show that we learn transformation patterns that correspond to the initial mapping links. These patterns are relevant and efficient (precision = 1). 17 new transformation patterns are also learned from the three used examples. These patterns seems also relevant, with a precision average than 0.83.

5 Related Work

Several approaches have been proposed to mine model transformation. The MTBE approach consists in learning model transformation from examples. An example is composed of a source model, the corresponding transformed model, and matching links between the two models. In [1,15], an alignment between source and target models is manually created to derive transformation rules. The approach of [5] consists in using the analogy to search for each source model its corresponding target model without generating rules. In a previous work [12], we use Relational Concept Analysis (RCA) to derive commonalities between the source and target meta-models, models and transformation links to learn executable transformation rules. The approach based on RCA builds transformation patterns that indicate how a model element, in a specific context, is transformed into a target element with its own specific context. This approach has many advantages for the case when the matching link type is one-to-one, but it is not able to capture the cases where a set of model elements is globally transformed into another set of model elements (matching link type is many-to-many). In this paper, we investigate graph mining approaches, to go beyond the limitations of our previous work. In the current context of MDE, transformation examples are not very large (they are manually designed), thus we do not expect scalability problems. Compared with a solution where we would build a lattice on graphs containing elements from both source and target models coming from matching links, the solution we choose separately classifies source graphs and target graphs. This is because source graphs and target graphs could come from the same meta-model (or from meta-models with common concepts) and it has no meaning in our context to generalize a source graph and a target graph together. We also think that the result is more readable, even in the case of disjoint meta-models.

Our problem has close connections with the pattern structure approach [4] when the pattern structure is given by sets of graphs that have labeled vertices. Graph mining approaches [2,10] aim at extracting repeated subgraphs in a set of graphs. They use a partial order on graphs which usually relies on morphism or on injective morphism, also known as subgraph isomorphism [9]. In the general case, these two morphism operations have an exponential complexity. In this paper, we rely on graph mining to classify independently the origins and the destinations of matching links and to infer from this, a classification of matching links, that is then used to extract transformation patterns.

6 Conclusion

We have proposed an approach to assist a designer in her/his task of writing a declarative model transformation. The approach relies on model transformation examples composed of source and target model fragments and matching links. Models and their fragments are represented by graphs with labelled vertices that are classified. This classification is in turn, used for classifying the matching links.

Finally, the mined model transformation patterns express how a source model fragment is transformed into a target model fragment. Future directions of this work include extending the evaluation to other kinds of source and target metamodels, and define a notion of support for the patterns. We also would like to explore the different kinds of graph mining approaches, in particular to go beyond the limitation of using locally injective graphs. Finally, we plan to apply our approach [12] to transform the obtained patterns into operational rules.

References

1. Balogh, Z., Varro, D.: Model Transformation by Example Using Inductive Logic Programming. Software and Systems Modeling 8(3), 347–364 (2009)
2. Cook, D.J., Holder, L.B.: Mining Graph Data. John Wiley & Sons (2006)
3. Fabro, M.D.D., Valduriez, P.: Towards the efficient development of model transformations using model weaving and matching transformations. Software and System Modeling 8(3), 305–324 (2009)
4. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: Proc. of ICCS'01. pp. 129–142 (2001)
5. Kessentini, M., Sahraoui, H., Boukadoum, M.: Model transformation as an optimization problem. In: MODELS'08, LNCS 5301. pp. 159–173. Springer (2008)
6. Lieberman, H. (ed.): Your Wish Is My Command: Programming by Example. Morgan Kaufmann Publishers (2001)
7. Liquiere, M., Sallantin, J.: Structural machine learning with Galois lattice and Graphs. In: Proc. of ICML'98. pp. 305–313 (1998)
8. Lopes, D., Hammoudi, S., Abdelouahab, Z.: Schema matching in the context of model driven engineering: From theory to practice. In: Advances in Systems, Computing Sciences and Software Engineering. pp. 219–227. Springer (2006)
9. Mugnier, M.L.: On generalization/specialization for conceptual graphs. J. Exp. Theor. Artif. Intell. 7(3), 325–344 (1995)
10. Nijssen, S., Kok, J.N.: The Gaston Tool for Frequent Subgraph Mining. Electr. Notes Theor. Comput. Sci. 127(1), 77–87 (2005)
11. Norris, E.: An algorithm for computing the maximal rectangles in a binary relation. Revue Roumaine Math. Pures et Appl. XXIII(2), 243–250 (1978)
12. Saada, H., Dolques, X., Huchard, M., Nebut, C., Sahraoui, H.A.: Generation of operational transformation rules from examples of model transformations. In: MoDELS. pp. 546–561 (2012)
13. Saada, H., Huchard, M., Nebut, C., Sahraoui, H.A.: Model matching for model transformation - a meta-heuristic approach. In: Proc. of MODELSWARD. pp. 174–181 (2014)
14. Weichsel, P.M.: The Kronecker product of graphs. Proceedings of the American Mathematical Society 13(1), 47–52 (1962)
15. Wimmer, M., Strommer, M., Kargl, H., Kramler, G.: Towards model transformation generation by-example. In: Proc. of HICSS '07. p. 285b (2007)

Interaction Challenges for the Dynamic Construction of Partially-Ordered Sets

Tim Pattison and Aaron Ceglar

{tim.pattison,aaron.ceglar}@defence.gov.au
Defence Science & Technology Organisation
West Ave, Edinburgh
South Australia 5111

Abstract. We describe a technique for user interaction with the interim results of Formal Concept Analysis which we hypothesise will expedite user comprehension of the resultant concept lattice. Given any algorithm which enumerates the concepts of a formal context, this technique incrementally updates the set of formal concepts generated so far, the transitive reduction of the ordering relation between them, and the corresponding labelled Hasse diagram. User interaction with this Hasse diagram should prioritise the generation of missing concepts relevant to the user's selection. We briefly describe a prototype implementation of this technique, including the modification of a concept enumeration algorithm to respond to such prioritisation, and the incremental updating of both the transitive reduction and labelled Hasse diagram.

1 Introduction

Formal Concept Analysis (FCA) takes as input a formal context consisting of a set of attributes, a set of objects, and a binary relation indicating which objects have which attributes. It produces a partially-ordered set, or *poset*, of formal concepts, the size of which is, in the worst case, exponential in the number of objects and attributes in the formal context [1]. The computational tasks of enumerating the set of formal concepts, and of calculating the transitive reduction of the ordering relation amongst them, therefore scale poorly with the size of the formal context. These steps are required to determine the vertices and arcs of the directed acyclic graph whose drawing is known as the Hasse diagram of the partial order. The layout of this layered graph prior to its presentation to the user is also computationally intensive [2]. For contexts of even moderate size, there is therefore considerable delay between user initiation of the process of FCA and presentation of its results to the user.

A number of algorithms exist which efficiently enumerate the formal concepts of a formal context [3–6]. In this paper, we describe an approach which incrementally updates and presents the partial order amongst the formal concepts generated so far. In particular, it: incrementally updates the transitive reduction of the interim partial order as each new concept is generated; incrementally updates the layout of the Hasse diagram; and animates the resultant changes to

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

the Hasse diagram to assist the user in maintaining their mental model. This approach enables user exploration and interrogation of the interim partial order in order to expedite their comprehension of the resultant complete lattice of concepts. It applies equally to any other partial order, the enumeration of whose elements is computationally intensive.

We also describe how this interaction can prioritise the generation and display of those missing concepts which are most relevant to the user's current exploratory focus. By addressing the scalability challenge of visual analytics [7], this user guidance of computationally intensive FCA algorithms [8] facilitates the required "human-information discourse".

1.1 Previous work

Incremental algorithms exist for updating the set of formal concepts and the transitive reduction of the ordering relation following the addition of a new object to the formal context [9–11]. A new object can give rise to multiple additional concepts which must be inserted in the existing complete lattice to produce an updated lattice which is also complete. In contrast, the technique described in this paper involves the addition of a single element at a time to a partially ordered set which is not in general a complete lattice.

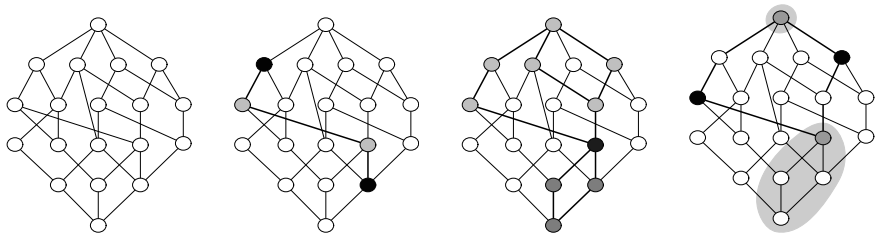
Ceglar and Pattison [8] have argued that user guidance of the FCA process could allow the satisfaction of the user's requirements with a smaller lattice, and consequently in less time, than standard FCA algorithms. They described a prototype tool which facilitates interactive user guidance and implements an efficient FCA algorithm which they have modified to respond to that user guidance. The user interaction challenges identified by that work are described and addressed in this paper.

2 Interacting with a Hasse diagram

2.1 The Hasse diagram

A finite poset $\langle P; < \rangle$ consists of a finite set P and an irreflexive, antisymmetric, transitive binary relation $<$ between its elements. Two elements $a, b \in P$ are said to be *comparable* if $a < b$ or $b < a$, and *incomparable* otherwise. $\langle P; < \rangle$ can be represented as an acyclic directed graph, or digraph, in which each element of the set P is a vertex and an arc connects each pair of comparable elements. The direction of each arc is "upward" in the sense of the relation $<$, from the lesser to the greater element.

The greater the number of comparable pairs, the greater the number of arcs, and hence the harder it is for a user to interpret a drawing of this digraph. The transitive reduction $\prec$ [12] of the ordering relation $<$ results from removing each arc whose source and destination vertices are connected by one or more other directed paths through the digraph. The transitive reduction therefore has fewer arcs than there are pairs of comparable elements in P. In the resultant



(a) Hasse diagram (b) Comparability (c) Upset, downset (d) Bounds

Fig. 1: Determining comparability, upset and downset, and bounds from the Hasse diagram

digraph, $a, b \in P$ are comparable iff there exists a directed path between the corresponding vertices, and b is said to be an *upper neighbour* of a, and a a *lower neighbour* of b, iff there is an arc from a to b.

A layered drawing of the resultant digraph, in which the vertical component of each arc is upwards on the page, is known as a *Hasse diagram* of the partial order. This direction convention reduces clutter by avoiding the need to explicitly represent the direction of the arcs. An example Hasse diagram is shown in Fig. 1a. The black vertices in Fig. 1b are comparable because a monotonically upward path, via the light-grey vertices, exists between them. The more elements there are in P, and the more arcs there are in the transitive reduction of $<$, the harder the visual task of tracing paths in the Hasse diagram to determine comparability.

2.2 Analytical objectives

An upper [respectively lower][1] bound on a given set $S \subseteq P$ is an element $u \in P$ $[l \in P]$ satisfying $\forall s \in S$, $s < u$ $[l < s]$. If S consists of a single element $a \in P$, the set of upper [lower] bounds on S is called an upset [downset], and can be identified by tracing all upward [downward] paths from a in the Hasse diagram. In Fig. 1c, the vertices in the upset [downset] of the black vertex are light [dark] grey. The union of a with its upset and downset gives the set of elements of P which are comparable with a. If S consists of two or more elements of P, then the set $U_S \subset P$ $[D_S \subset P]$ of its upper [lower] bounds is the intersection of the upsets [downsets] of its elements. In Fig. 1d, the sets of upper and lower bounds on the pair of black vertices are encompassed by the shaded areas. The visual task of identifying and intersecting these upsets [downsets] is demanding for small partial orders, and rapidly becomes intractable as the size $|P|$ of P increases.

If there exists an $a^* \in U_S$ such that $a^* \leqslant a$ for all upper bounds $a \in U_S$ on S, then a^* is called the *least upper bound* (LUB); if there exists a $b^* \in D_S$

[1] Square brackets are used throughout this paper to indicate that a sentence is true both when read without the bracketed terms and when read with each bracketed term substituted for the term which precedes it.

such that $b \leqslant b^*$ for all lower bounds $b \in D_S$ on S, then b^* is called the *greatest lower bound* (GLB). In Fig. 1d, the least upper and greatest lower bounds on the pair of black vertices are the upper and lower shaded vertices respectively. By definition, the LUB and GLB are unique whenever they exist, since equality corresponds to identity. In addition to the challenge of identifying the set of upper [lower] bounds on S, the visual task of establishing the existence and identity of the LUB [GLB] from this set also becomes intractable for large $|P|$.

If the LUB and GLB exist for all pairs of elements in P, then $\langle P; < \rangle$ is called a *lattice*. If the LUB and GLB exist for all $S \subseteq P$, then $\langle P; < \rangle$ is called a *complete lattice*, and the LUB [GLB] on P is known as the *supremum* [*infimum*].

2.3 Computer-assisted interaction

In the previous section, we identified a number of operations on posets which a user can perform visually by tracing paths in the Hasse diagram, but which become intractable as $|P|$ increases. To support the user in these tasks, the computer should calculate the results of these operations and display them by highlighting elements of the Hasse diagram. In particular:

1. Comparability between selected elements of P can be determined computationally and all identified paths between them highlighted in the Hasse diagram.
2. The set of elements comparable with a selected element can be determined by calculating and highlighting the members of its upset and downset.
3. The set of upper [lower] bounds on a set S of elements can be determined by calculating the upsets [downsets] of each, and highlighting the members of their intersection.
4. The existence and identity of the LUB [GLB] on a selected subset S can be determined computationally from the result of 3. The identified element of P, if any, should be highlighted.

3 Dynamic presentation of the partial order

So far we have assumed that the elements of $\langle P; < \rangle$ are known *a priori*, so that the transitive reduction can be computed and the Hasse diagram laid out before being presented to the user. Consider now the case where the user's request for the Hasse diagram triggers the on-demand enumeration of the elements of P. If this enumeration is computationally intensive, the user may experience excessive delay before the results are presented. Rather than waiting for the generation of all elements, the user may wish to commence interaction with the Hasse diagram for the elements generated so far, and have this diagram evolve to incorporate each new element as it is added. We hypothesise that user exploration of, and familiarisation with, the evolving $\langle Q \subseteq P; < \rangle$ will facilitate and expedite comprehension of $\langle P; < \rangle$.

In order to examine the feasibility of user interaction with the Hasse diagram of $\langle Q; < \rangle$ as a proxy for that of $\langle P; < \rangle$, the subsequent sections address the following four key questions:

1. What information about $\langle P; < \rangle$ can and cannot be ascertained through inspection of $\langle Q; < \rangle$?
2. Should the response to user interrogation of $\langle Q; < \rangle$ be expressed in terms of $\langle P; < \rangle$?
3. Can $\langle Q; < \rangle$ be incrementally updated to incorporate each new element of P as it is generated?
4. How should these updates be presented so as to minimise disruption to the user's mental model of $\langle Q; < \rangle$?

3.1 Comparing the interim and final partial orders

Two elements which are comparable in $\langle P; < \rangle$ are comparable in any $\langle Q \subset P; < \rangle$ in which they both exist. The upset [downset] of an element in $\langle P; < \rangle$ is a (possibly improper) superset of its counterpart in $\langle Q; < \rangle$. Accordingly, the set of upper [lower] bounds on $S \subseteq Q$ in $\langle P; < \rangle$ is also a superset of its counterpart in $\langle Q; < \rangle$. Importantly, the LUB [GLB] on $S \subseteq Q$ may not be present in $\langle Q; < \rangle$, even if it is in $\langle P; < \rangle$.

3.2 Interacting with the interim partial order

User interaction could be defined to implement the same computer-assisted operations on $\langle Q; < \rangle$ as were defined above on $\langle P; < \rangle$. However, since the user's objective is to find out about $\langle P; < \rangle$, interaction with $\langle Q; < \rangle$ should also prioritise the generation of the requisite elements of P.

The selection of two elements a and b in order to determine their comparability might confirm comparability by not only highlighting the elements $x \in Q$ which lie between a and b, but also prioritise the generation of all such $x \in P$. Selection of an element might not only display its upset and downset in $\langle Q; < \rangle$, but also prioritise the completion of these sets by the process which generates the elements of P. Similarly, the selection of a set $S \subset Q$ of elements in the interim partial order may not only result in the display of the corresponding upper and lower bound sets, but also prioritise the completion of these sets. And finally, if the user requests the LUB [GLB] on a set $S \subset Q$ of elements, then the computer could prioritise the generation of the corresponding result in $\langle P; < \rangle$. If the requested bound exists, the corresponding element of P could be added, if not already present, and highlighted in the Hasse diagram; otherwise a null result should be signalled to the user.

3.3 Updating the interim partial order

As each new element e of P is generated and added to $\langle Q; < \rangle$ to form $\langle Q \cup \{e\}; < \rangle$, both the element set and the transitive reduction $\prec$ of $<$ must be updated. Updating the transitive reduction involves identifying the upper and lower neighbours of e, adding the requisite arcs, and deleting any arcs from the lower to the upper neighbours.

The set $\mathcal{N}^u := \min\{u \in Q : e < u\}$ $\left[\mathcal{N}^l := \max\{l \in Q : l < e\}\right]$ of upper [lower] neighbours can be identified through a top-down [bottom-up] search of $\langle Q; < \rangle$ starting from each of the maximal [minimal] elements of Q. Each downward [upward] path from each maximal [minimal] element can be pruned from the point at which an element is encountered which is not greater [less] than e. An element in $\{u \in Q : e < u\}$ $[\{l \in Q : l < e\}]$ is also in $\mathcal{N}^u$ $[\mathcal{N}^l]$ iff it has no lower [upper] neighbour greater [less] than e.

Once the sets $\mathcal{N}^u$ and $\mathcal{N}^l$ have been identified, the neighbour relation $\prec$ should be updated as follows. In $\langle Q \cup \{e\}; < \rangle$, $e \prec \alpha$ for each $\alpha \in \mathcal{N}^u$ and $\zeta \prec e$ for each $\zeta \in \mathcal{N}^l$. To maintain $\prec$ as the transitive reduction of $<$, any arcs in $\mathcal{N}^l \times \mathcal{N}^u$ must also be removed. Note that either or both of the sets $\mathcal{N}^u$ and $\mathcal{N}^l$ can be empty; the element e is maximal in $\langle Q \cup \{e\}; < \rangle$ iff $\mathcal{N}^u = \emptyset$, and minimal iff $\mathcal{N}^l = \emptyset$.

3.4 Presenting Hasse diagram updates

The layout of the Hasse diagram for $\langle Q \cup \{e\}; < \rangle$ will necessarily differ from that of $\langle Q; < \rangle$. The addition of the new vertex e will require either its accommodation within an existing layer or the creation of a new layer. The deletion of some existing arcs and the creation of new ones may also worsen the aesthetic criteria, such as number of edge crossings, which the chosen layout algorithm seeks to optimise [2].

The addition of the new vertex and its incident arcs, the deletion of superseded arcs, and any changes in layout, must be presented to the user in a way which minimises disruption of the user's mental model of $\langle Q; < \rangle$. The sequential animation of each step in this process is a logical solution to this problem. By also minimising and localising any changes to the layout necessitated by the addition of a new vertex, the number of vertices and edges whose movement the user must visually track can be minimised.

Figure 2 illustrates the process of adding a new element to $\langle Q; < \rangle$ in Fig. 2a. The upper (light grey) and lower (dark grey) neighbours of the new element are identified, and the edges between them (bold) slated for deletion. The vertical separation between the top and bottom layers is increased to make room for a new layer, which is required to accommodate the new vertex. Finally, the new vertex is inserted and attached to its upper and lower neighbours. Placement of the new vertex near its upper and lower neighbours helps localise the resultant changes to the Hasse diagram.

4 Application to Formal Concept Analysis

4.1 Introduction to FCA

FCA takes as input a bigraph consisting of a set G of objects, a set M of attributes, and a binary relation $R \subseteq G \times M$ between them. FCA produces a complete lattice of formal concepts. Each formal concept consists of a set $E \subseteq G$,

(a) Identify neighbours and superceded edges

(b) Make room for, insert and connect new vertex

Fig. 2: Inserting a new element into the poset $\langle Q; < \rangle$.

known as the *extent*, and a set $I \subseteq M$, known as the *intent*, such that $E \times I \subseteq R$. Each object in E has all attributes in I and each attribute in I is possessed by all objects in E. The extent and intent are maximal in the sense that adding elements to either set necessarily entails removing elements from the other. The relation $<$ between concepts is the subset relation $\subset$ between their extents. The upset [downset] of a concept consists of concepts whose intents [extents] are subsets of those of the nominated concept.

4.2 Labelling

The concept whose extent is the set of objects possessing attribute i is referred to as the *attribute concept* for i; the concept whose intent is the set of attributes possessed by object j is known as the *object concept* for j. In the Hasse diagram for the complete lattice of formal concepts, an attribute [object] concept is labelled with the corresponding attribute [object]. The intent of a concept can be inferred from the set of attribute labels on vertices in its upset, while its extent can be inferred from the set of object labels on vertices in its downset.

This reduced labelling scheme works for the interim partial order $\langle Q; < \rangle$ provided that all object and attribute concepts are present *ab initio*. The algorithm which generates the formal concepts should be chosen or modified so as to produce these first, and rendering of the Hasse diagram deferred until they are included in Q. It will be seen in Sect. 4.6 that this deferral also has benefits for the layout of the Hasse diagram.

4.3 Least upper and greatest lower bounds

In FCA, the intent of the LUB on a set of concepts is the intersection of their intents, while the extent of the GLB is the intersection of their extents. Thus, the LUB [GLB] indicates the set of attributes [objects] which they have in common. Since the partial order amongst the full set of formal concepts for a formal context is a complete lattice, the LUB and GLB are guaranteed to exist; they may however be absent from $\langle Q; < \rangle$, and will in general require computation on demand.

Notably, the LUB and GLB on the full set P of formal concepts may not be present in $\langle Q; < \rangle$. A concept enumeration algorithm should be chosen or modified to produce these first. In addition to improving the resemblance between $\langle Q; < \rangle$ and the resultant complete lattice $\langle P; < \rangle$, the existence of the LUB [GLB] *ab initio* provides a single, consistent starting point for the top-down [bottom-up] search of the transitive reduction of $\langle Q; < \rangle$ for the upper [lower] neighbours of each new concept. This process is described in Sect. 4.4.

Selection of a set $\mathcal{A}$ of concepts could trigger the generation, display and highlighting of all concepts $x \in P$ satisfying $\mathcal{A}_* < x < \mathcal{A}^*$. Here $\mathcal{A}_*$ and $\mathcal{A}^*$ denote the GLB and LUB on the set $\mathcal{A}$ in the complete lattice $\langle P; < \rangle$ of formal concepts. $\mathcal{A}_*$ and $\mathcal{A}^*$ constitute the GLB and LUB on the complete lattice of elements whose generation is to be prioritised. Their generation is given the highest priority in order to bound this poset. The elements of $\mathcal{A}$ are all mutually comparable iff at least one directed path from $\mathcal{A}_*$ to $\mathcal{A}^*$ passes through all of these elements. In this case, $\mathcal{A}_* \in \mathcal{A}$ and $\mathcal{A}^* \in \mathcal{A}$; in the special case of $|\mathcal{A}| = 2$, this is a necessary and sufficient condition for comparability.

4.4 Insertion point

As described in Sect. 3.3, the upper [lower] neighbours of the new concept can be identified in $\langle Q; < \rangle$ using a top-down [bottom-up] directed search. The search commences from the LUB [GLB] on P, which by design is present in Q *ab initio*. The current vertex is marked as having been visited, and any lower [upper] neighbour whose intent [extent] is a subset of that of the new concept is queued for subsequent traversal. If the current concept has no such neighbours, then it is an upper [lower] neighbour of the new concept.

4.5 Layer assignment

The poset $\langle Q; < \rangle$ is presented to the user as a layered drawing of the corresponding directed acyclic graph. In order to maintain the direction convention of upward arcs in this Hasse diagram, each new concept must be assigned to a layer which is separated from the infimum [supremum] by at least the maximum length of all directed paths between them in the transitive reduction of $\langle Q; < \rangle$. If no existing layer satisfies both constraints, one must be added. If only one such layer exists, the new concept is assigned to that layer. If more than one existing layer satisfies both constraints, the choice amongst them is arbitrary.

Even if at least one layer satisfies both constraints on path length, it is still possible for the new concept to have upper and lower neighbours in adjacent layers. Adding an intervening layer in this case would create room for the new concept, and would not require the revision of any previous layer assignments. However, given that there are already sufficient layers to accommodate the new concept, a more space-efficient strategy would be to instead promote the lowest upper neighbours to the next highest layer or demote the highest lower neighbours to the next lowest layer. To make room for either change, it may also

be necessary to promote additional members of the upset or demote additional members of the downset of the new concept.

In our preliminary implementation of layer assignment, we: assign a vertex to the uppermost layer, if any, satisfying the dual path length constraints; attempt to demote elements of the downset in order to make room to insert a new concept between upper and lower neighbours which are in adjacent layers; and insert a new layer, if required, which has path length two from the infimum to make room for this demotion. The reason for the choice of path length two will become apparent in Sect. 4.6. This layering strategy is illustrated later in Sect. 4.8.

The chosen strategy requires that concepts be allowed to migrate between layers following initial presentation to the user, which has the potential to disrupt the user's mental model. On the other hand, creating more layers than are required results in inefficient use of the vertical space available for drawing the Hasse diagram, making it harder for the user to maintain simultaneous focus and context as $|Q|$ increases.

4.6 Layout

For the reason given in Sect. 4.3, the concept enumeration algorithm should first generate the LUB and GLB of the poset $\langle P; < \rangle$ of concepts. These should be placed at the centre top and centre bottom of the canvas. For the reason given in Sect. 4.2, all (remaining) attribute and object concepts should be generated next. In this section, we prioritise from amongst these the generation of the lower neighbours of the supremum, which are called *atoms*, and the upper neighbours of the infimum, which are called *co-atoms*. We describe a scheme whereby their relative ordering within their respective layers is chosen so as to improve the aesthetics of the resultant Hasse diagram. The horizontal placement of all subsequent concepts, including the remaining attribute and object concepts, is dependent on this ordering.

The supremum and infimum, along with all atoms and co-atoms, are generated and laid out before the Hasse diagram is first presented to the user. The atoms and co-atoms are ordered within their respective layers so as to minimise edge crossings in the relation R. As subsequent concepts are discovered and added, each edge morphs into one or more directed paths in the transitive reduction of $\langle Q; < \rangle$. Provided care is taken in the placement of the remaining concepts on these paths, the effort invested in minimising edge crossings in R might therefore be repaid with fewer arc crossings, and potentially also shorter paths, in the Hasse diagram.

For each newly-discovered concept, the horizontal barycentre [13] of the atoms and co-atoms with which it is comparable is calculated, and a total order on these barycentres is used to order the concepts within a layer. The barycentre calculation assumes that the co-atom and atom layers are assigned equal width and that atoms and co-atoms are equally spaced within their respective layers. The assigned order, which is designed to place each concept in reasonable horizontal proximity to the corresponding atoms and co-atoms, is only dependent on the fixed horizontal placement of the atoms and co-atoms, rather than the entire

upset and downset. This choice also preserves the order amongst the concepts already placed within the layer, since each new concept is simply inserted into that existing order.

The creation of a new layer may be required to separate vertically, by at least one layer, the upper and lower neighbours of a new concept. That new layer is created immediately above the layer of co-atoms, and elements of the downset of the new concept are demoted as required. Note that demotion beyond this layer would violate the constraint on path length from the infimum. Since each new concept is placed in the uppermost layer satisfying the constraint on path length to the supremum, and new layers are only ever added with path length two from the infimum, concepts will either remain in the layer to which they are originally assigned, or migrate downward.

4.7 Guided concept enumeration

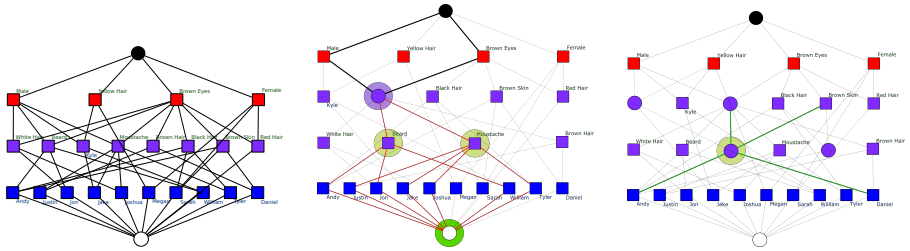
We have so far assumed that algorithms for the enumeration of formal concepts could be modified to prioritise the generation of concepts relevant to the user's current selection. To demonstrate that this is possible, we briefly describe a modification of the algorithm of Choi and Huang [14] to prioritise the generation of the downset of a selected concept.

Choi and Huang enumerate the concepts of a formal context in top-down, breadth-first order, extending the intent of each concept generated so far using attributes not currently in its intent. Ceglar and Pattison [8] have modified this algorithm to: hand over each novel concept, as it is generated, for subsequent processing; respond to user input, if any, after the generation of each novel concept; and switch to depth-first processing at the user's request to enumerate the downset of a nominated concept, resuming breadth-first processing upon completion. The upset of a nominated concept could be generated similarly by switching to bottom-up, depth-first processing, adding objects not in the extent of the nominated concept.

Efficient FCA algorithms employ various strategies for preventing the repeated generation of the same concept, typically by traversing a trie structure [15] superimposed on the concept lattice. Since user guidance of the FCA algorithm can interfere with this strategy, an efficient test for concept novelty, based for example on hash tables [15], is essential. In the proposed interactive approach, the overall efficiency of concept enumeration is less important than responsiveness to user interrogation of the interim partial order.

4.8 Prototype implementation

Figure 3 shows mockups of a prototype interface for the incremental construction of a poset of formal concepts. The example formal context consists of people and their physical attributes. Figure 3a depicts the state of the Hasse diagram first presented to the user. By this stage, all of the attribute and object concepts have been generated by the concept enumeration algorithm, labelled, and laid out to establish the framework for insertion of subsequent concepts.



(a) Initial diagram (b) Multiple selection (c) Insertion

Fig. 3: The result of multiple selection in the initial Hasse diagram, and of the subsequent insertion of multiple concepts.

Figure 3b shows the result of the user selecting in this diagram the attribute concepts for "Beard" and "Moustache", which are highlighted in response with light green halos. This multiple selection triggers the calculation of the extent and intent intersections for the selected concepts. The former corresponds to the infimum, which is accordingly highlighted with a bright green halo; the latter corresponds to a new concept, which is consequently inserted into the Hasse diagram and highlighted with a purple halo. Since this least upper bound has path length 2 from the supremum, a new row has been inserted to accommodate concepts with path length 3, and the selected concepts demoted to it. The least upper bound is inserted into row 2 at ordinal position 2 of 5; this position is based on the horizontal barycentre of its associated atoms and co-atoms, which are predominantly to the left of the centreline. Figure 3c shows the state of the Hasse diagram following the subsequent addition of a number of concepts; the most recently added concept is highlighted and its adjacent arcs shown green.

The prototype interface does not yet implement all of the recommendations in this paper. For example, while multiple selection triggers the calculation and display of the LUB and GLB, it does not prioritise the highlighting and completion of the set of concepts between them. Change animation is also currently lacking. If the context bigraph consists of more than one connected component, these should be dealt with separately (albeit connected to the same supremum and infimum) and the results horizontally juxtaposed. This reduces the computational complexity, and ensures a better result, of the layout of the atoms and co-atoms. It also allows correct handling of the exceptional case where the same concept is both an atom and co-atom.

5 Summary and Future Work

This paper has described a technique for the incremental construction of the Hasse diagram of a poset. We hypothesise that user interaction with this evolving diagram will expedite user comprehension of the partial order. When applied to FCA, this technique incrementally updates the transitive reduction, and the

labelled Hasse diagram, of the partial order amongst the set of concepts generated so far. User interaction with the Hasse diagram prioritises the generation of missing concepts relevant to the user's selection. A prototype implementation, including the modification of a concept enumeration algorithm to respond to downset prioritisation, has also been described through which the hypothesis will be tested in future work. Scalable mechanisms for interactive concept deletion should also be explored to allow the exclusion from the interim partial order of concepts which are no longer of interest to the user.

References

1. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer-Verlag New York, Inc. (1997)
2. Di Battista, G., Eades, P., Tamassia, R., Tollis, I.: Graph Drawing. Prentice Hall, Upper Saddle River, NJ (1999)
3. Kuznetsov, S.O., Obiedkov, S.A.: Algorithms for the construction of concept lattices and their diagram graphs. In: PKDD '01: Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery, Springer (2001) 289–300
4. Kuznetsov, S., Obiedkov, S.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental and Theoretical Artificial Intelligence **14**(2-3) (2002) 189–216
5. Priss, U.: Formal Concept Analysis in Information Science. Annual Review of Information Science and Technology **40** (2006) 521–543
6. Strok, F., Neznanov, A.: Comparing and analyzing the computational complexity of FCA algorithms. In: Proceedings of the 2010 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists. SAICSIT '10, New York, NY, USA, ACM (2010) 417–420
7. Thomas, J.J., Cook, K.A., eds.: Illuminating the Path: The Research and Development Agenda for Visual Analytics. IEEE Press (2005)
8. Ceglar, A., Pattison, T.: Guided Formal Concept Analysis. Technical report, Defence Science and Technology Organisation (2014) To appear.
9. Norris, E.: An algorithm for computing the maximal rectangles in a binary relation. Revue Roumaine de Mathématiques Pures et Appliquées **23**(2) (1978) 243–250
10. Missikoff, M., Scholl, M.: An algorithm for insertion into a lattice: Application to type classification. In Litwin, W., Schek, H.J., eds.: Foundations of Data Organization and Algorithms. Volume 367 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (1989) 64–82
11. Godin, R., Missaoui, R., Alaoui, H.: Incremental concept formation algorithms based on Galois (concept) lattices. Computational Intelligence **11**(2) (1995) 246–267
12. Aho, A., Garey, M., Ullman, J.: The transitive reduction of a directed graph. SIAM Journal on Computing **1**(2) (1972) 131–137
13. Mäkinen, E., Siirtola, H.: The barycenter heuristic and the reorderable matrix. Informatica **29** (2005) 357–363
14. Choi, V., Huang, Y.: Faster algorithms for constructing a Galois lattice, enumerating all maximal bipartite cliques and closed frequent sets. In: SIAM Conference on Discrete Mathematics, University of Victoria, Canada. (2006)
15. Sedgewick, R.: Algorithms. second edn. Computer Science. Addison-Wesley (1988)

The educational tasks and objectives system within a formal context*

Ľubomír Antoni, Ján Guniš, Stanislav Krajči,
Ondrej Krídlo, Ľubomír Šnajder

Institute of Computer Science, Faculty of Science,
Pavol Jozef Šafárik University in Košice, Jesenná 5, 040 01 Košice, Slovakia.
lubomir.antoni@student.upjs.sk, jan.gunis@upjs.sk,
stanislav.krajci@upjs.sk, ondrej.kridlo@upjs.sk, lubomir.snajder@upjs.sk

Abstract. The educational objectives represent the precise statements of what we expect or intend students to learn as a result of education. We have conducted an analysis of the educational tasks and objectives system within a formal context with respect to the collected real data on an array data structure of five teachers in the field of computer science. We submitted a report and the corresponding concept lattice to each individual teacher and explored their additional feedback. In addition, we formulate the general observations and present the feasible set of tasks and objectives of an array data structure. The results are expected to annotate in the future formation of the curricular documents as supplement to the National Education Program in Slovak republic which is formulated concisely.

Keywords: formal context, educational task, system of objectives, concept lattice

1 Introduction

The scope of the computer science education in Slovakia is officially declared in The National Education Program of Slovak republic as the supreme curricular document. The Slovak National Education Program defines the main principles and general objectives on which education and training in computer science is based. The education of computer science at secondary schools in Slovakia includes five areas:

a) Theory of the information (numeral systems, coding, compression, etc.),
b) Information and communication technologies – ICTs (internet, computer networks, safety, etc.),

* This work was partially supported by the grant VEGA 1/0832/12 and by the Slovak Research and Development Agency under contract APVV-0035-10 "Algorithms, Automata, and Discrete Data Structures". We wish to express the special thanks to the teachers for their cooperation in this experiment and to the reviewers for their helpful remarks.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

c) Procedures, problem solving and algorithmic thinking (algorithms, program, programming languages, etc.),
d) Principles of ICTs operations (software, hardware, architectures, etc.),
e) Information society (e-learning, licenses, risks, etc.).

Formal concept analysis [14] as a lattice theory allows us to explore the meaningful groupings of educational tasks (referred to objects) with respect to common objectives (referred to attributes) and it provides the visualization capabilities. The conceptual difficulties in mathematics education [32], or the integrated care pathways [30] are analyzed by formal concept analysis, as well. An extensive overview of the various application domains that include software mining, web analytics, medicine, biology and chemistry data is given by [29], [11]. Recently, the feasible attempts and generalizations are investigated in [1,5,7,22].

In this paper, our aim is to provide the system of objectives and tasks that is expected to fill in the gap of the National Education Program in Slovak republic. In general, the National Education Program is formulated concisely and we put emphasis in a long term to particularize other supplementary curricular documents and express the educational objectives more explicit in various areas. Therefore, we have focused on an algorithmic thinking area and chosen an array data structure as an educational content in which we have fruitfully applied formal concept analysis. Simultaneously in this area, we focus on algorithms including searching, sorting or text processing. In other countries, the national curricular documents and other standards define the educational objectives in the various levels of specification, see [36, 37].

2 Educational objectives of an array data structure

An array data structure, as a collection of indexed elements, plays an important role in the education of programming. An array or its equivalent as a kind of data type is implemented in the most of programming languages. The term is also used in a theoretical computer science as abstract data type.

We aim at specifying the particular and relatively precise objectives of an array data structure education in the algorithmic thinking area. Regarding our long-term cooperation with the teachers in the field, we declare some input set of objectives of an array data structure:

1) to specify an array as the structured homogeneous data type with elements denoted by single identifier,
2) to appoint the real examples of one-dimensional array data structure (e.g. rooms in a hotel, seats in a plane, etc.),
3) to interpret the notions of an array index (an array key) and an array element and to explain the difference between them,
4) to differ an array index type and an array element type,
5) to reason that an array index type is an ordinal type (numbers, characters, other enumerations),
6) to declare a variable of array,

7) to read and to write out the array elements,
8) to manipulate with the array elements, to assign the array element to the other variables, to increment the array elements,
9) to appoint the common errors related to an array data structure (incorrect index type, overflow, incompatibility of the types),
10) to apply an array data structure in the simple issues (e.g. to store an array, to find the maximal value, to modify the elements of array, etc.),
11a) to apply an array data structure in a searching,
11b) to apply an array data structure in a sorting,
11c) to multiple access to the array elements,
11d) to apply an array data structure in a text processing,
11e) to apply an array data structure in a simple game programming,
12) to recognize the issues in which can be applied an array data structure effectively, to appoint the advantages and disadvantages of an array in comparison with other simple data structures (an access to elements, a space complexity).

The specified aims are enumerated by the revised taxonomy of Bloom [18] in order to classify statements of what we expect or intend students to learn as a result of education. The revised taxonomy focuses on four knowledge dimensions including factual knowledge (basic elements), conceptual knowledge (interrelationships among the basic elements), procedural knowledge (how to do something) and metacognitive knowledge (awareness and knowledge of one's own cognition). In general, an educational process consists of a motivation phase, a phase of the first acquisition, a fixation phase and a diagnostic phase. The phase of a systematization, a propedeutics or an application phase can be also involved.

We submitted the previous list of aims to the teachers in the secondary schools in Slovakia. The teachers were instructed to appoint the tasks which they usually apply in an educational process of an array data structure in programming. Teachers were not limited by the number of tasks and moreover, it was possible to add some additional aims (13, 14, ...) if they required. Having such instructions, every teacher was asked to fill in the following table:

Fig. 1. Table of tasks and aims

N.	task / aim	1	2	3	4	5	6	7	8	9	10	11a	11b	11c	11d	11e	12	13	⋯
0	2nd max. element				x		x	x			x	x		x				additional aims	
1																			
2																			
⋮																			

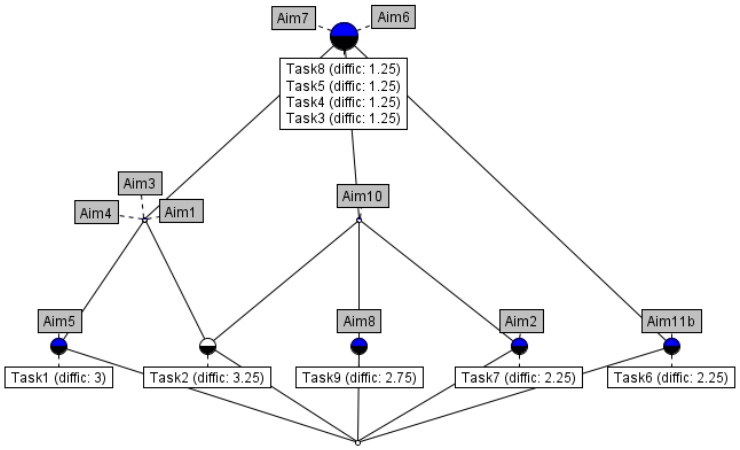
The zero row in the table is an example of task which we have added for an illustration. Particularly, we have appointed the task of finding the second largest element in an array data structure, which fulfills the aims $4, 6, 7, 10, 11a, 11c$.

3 Concept lattice of each individual teacher

We have obtained the data of five teachers who proposed overall 92 tasks, some of them equal with respect to the aims. Four teachers launched the additional

educational aims including the applying of an array data structure as the parameter of the procedure, a dynamic array, two-dimensional arrays, the issue of indexing the first element, reasoning initializing errors (seven aims in overall). For each individual teacher we have constructed the concept lattice from the collected data using ConExp[1] software. We use a concept lattice with reduced labeling (labeled line diagram) regarding own objects and own attributes which is accessible to human reasoning. An example of a concept lattice shown to one of the teachers follows in Fig. 2. In effort to explore the task difficulty, we have assigned the degree to each individual aim from the set $\{0.25, 0.5, 0.75, 1\}$ depending on its dimension in Bloom revised taxonomy. A value of the task difficulty (computed as the sum of the degrees of involved aims) is shown in a concept lattice as a supplement of the particular task label (i.e. own object).

Fig. 2. Concept lattice of the participated teacher



In our feedback given to teachers, we have added the comments and some additional questions. The comments include the aims which were obtained in all tasks (Aim6 and Aim7 in Fig. 2), the tasks which contain the unique aims (for instance Aim11b is included only in Task6) and the following instructions how to read a concept lattice to help teachers analyze the results:

- tasks in the first row are the representatives of the teacher's system of tasks (there is no task with the superset of aims),
- the shaded labels linked to a node in the first row represent the attributes introduced uniquely in the task,
- the aims introduced by a particular task one can obtain by collecting the shaded labels on all paths leading up from the selected task node,

[1] Concept Explorer, version 1.3, website: http://conexp.sourceforge.net. Nevertheless, we have successfully tested some other formal concept analysis software tools, for instance FcaStone, Lattice Miner, ToscanaJ, FCART, as well.

- the tasks that involve a particular aim one can acquire by tracing task labels leading down from the selected aim node,
- the top element of a concept lattice introduces the aims obtained in all tasks,
- a task in a higher row of a concept lattice is appropriate for the first acquisition phase of an educational process (not compulsory),
- a task in a lower row of a concept lattice is appropriate for the fixation or systematization phase of an educational process (not compulsory).

We state that not all of the submitted aims were used by the engaged teachers (for instance Aim9 is not introduced in Fig. 2). In contrary, some of the teachers have added their additional aims. Namely, we have first analyzed the systems of tasks from the viewpoint of each individual teacher. Otherwise, not introduced aims would be figured at the bottom element of a concept lattice.

For each concept lattice, we have calculated the degree of tasks and aims system gradation level as the proportion of the number of task nodes (as own objects) in the longest path and the total number of the tasks in a concept lattice. This indicator shows how gradated are tasks of teacher's system. The smaller number indicates the more diversified system, the higher number expresses the more gradational system. The obtained results and concluding remarks follows:

- the minimal number of tasks was 9, the maximal 27 in teacher's set of tasks,
- the most frequently introduced aims are Aim6, Aim7 in order to declare a variable of array and to read/write out elements of an array,
- two systems contain the aim(s) introduced in every task,
- two systems include the set of equal tasks with respect to the aims (i.e., at least two tasks equal),
- the gradational level in the systems takes the values from 0.13 to 0.22,
- the average task difficulty in teacher's set of tasks takes the values from 1.97 to 4.09; lower value indicates that the set of tasks is appropriate more for beginners, the higher value expresses focusing on advanced students.

We were interested in a feedback of the teachers in relation with the obtained results. One of the teachers confirmed that his/her set of tasks was used for advanced students (the average task difficulty is 3.22). The teachers have explained the reasons to add some new aims, reported the tasks which they used to apply in a diagnostic phase, declared the most problematic aims for students, etc. These issues and some other recommendations will be still discussed with the teachers and other respondents in a formal and an informal way.

4 Attribute exploration of each individual teacher

Beside the concept lattice diagram one can examine the implications between attributes valid in a teacher's tasks and aims tables. For instance, the implication {Aim7, Aim8} $\rightarrow$ {Aim6} shows that the following rule holds in a table of tasks and aims: Aim6 is introduced in every task that includes Aim7 and Aim8 together. This means that the task focused on reading, writing out and

manipulation with the elements of an array will also satisfy the aim of the declaration of an array data type. We have verified that this natural implication holds for the all five collected set of tasks (confidence of this association rule is 100%). On the other hand, the implication $\{\text{Aim8}\} \rightarrow \{\text{Aim7}\}$ does not hold in general, because there are some tasks (the counterexamples) focused on manipulation with the array elements, but do not read and write out the array elements (confidence of this association rule is 66%). Another natural implication $\{\text{Aim4, Aim5}\} \rightarrow \{\text{Aim3}\}$ means that every task focused on recognizing a difference between an array index type and an array element type will satisfy the aim of interpretation of an array index and array element. This association rule holds in the tasks of teachers with confidence 86%.

The implications one can read off from the concept lattice with reduced labeling, it is sufficient to check whether the each attribute's node from an implication's conclusion is above (or equal to) the infimum of all attributes nodes from a premise. In addition, one can compute the Duquenne-Guigues basis of implications, which is optimal in terms of its size and includes a minimum cover of all valid attribute implications, for more information see [14], [6].

The knowledge acquisition method called attribute exploration is described in general by [14]. In our experiment, we generate Duquenne-Guigues basis of implications for each individual teacher. Then, for every implication (one by one) one can make a decision to accept or provide a counterexample. By providing a counterexample, we suggest to add a task with the combination of aims which was still lacking and is fruitful to include in an education. For instance, the attribute exploration process for one of the teachers is shown in Fig. 3. We introduce these implications from basis in which the premise is satisfied by at least one task in the table.

Fig. 3. The educational aim exploration

N.	question	answer	advised counterexample
1.	$\emptyset \rightarrow \{6,7\}$?	yes	
2.	$\{6,7,8\} \rightarrow \{10\}$?	yes	
3.	$\{5,6,7\} \rightarrow \{1,3,4\}$?	yes	
4.	$\{4,6,7\} \rightarrow \{1,3\}$?	yes	
5.	$\{3,6,7\} \rightarrow \{1,4\}$?	no	a task with $\{3,6,7,10\}$
6.	$\{2,6,7\} \rightarrow \{10\}$?	yes	
7.	$\{1,6,7\} \rightarrow \{3,4\}$?	yes	

First question in Fig. 3 indicates that all tasks cover the aims Aim6, Aim7. We have answered this question yes that means that we agree to preserve this implication in a teacher's set of tasks. In contrary, fifth question expresses that if a task has aims Aim3, Aim6, Aim7, then it also has aims Aim1, Aim4. We do not agree to preserve this implication and advise to add a new task having aims Aim3, Aim6, Aim7, Aim10. It is advised to distinguish Aim3 and Aim4. Actually, Aim3 is introduced in a task having, for instance, integers as an array index type and also as an array element type. Nevertheless, Aim4 requires differentiation of an array index type and an array element type to achieve this goal absolutely.

The separation of these two aims in at least one educational task is helpful to encourage students to understand an array data type. Hence, we advise in addition a counterexample of a task with aims Aim3, Aim6, Aim7, Aim10 by the educational aim exploration shown in Fig. 3. This task removes the undesirable implication from the basis of all implications. Remind that a counterexample must not contradict the implications we have confirmed so far (the rows 1,2,3,4 in Fig. 3). However, if the counterexamples are added into the table, the concept lattice is modified.

5 The summary results

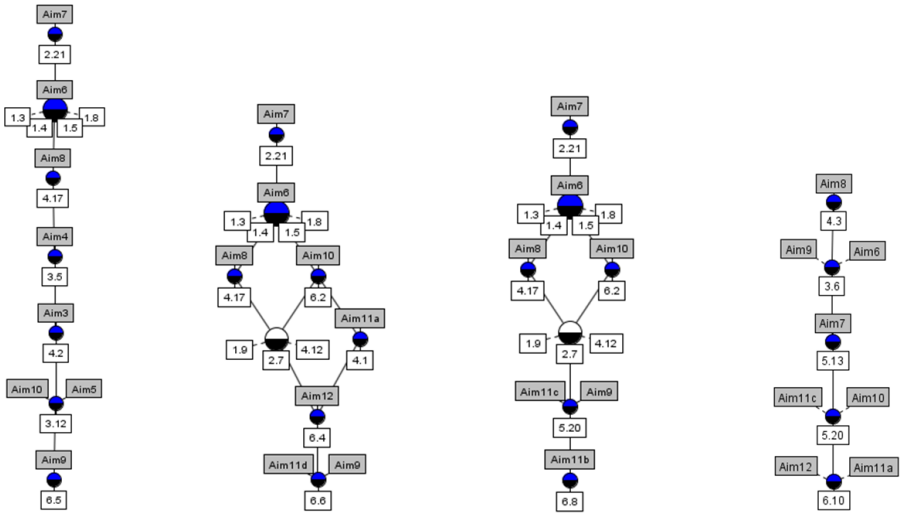
Regarding five teachers data and one additional set of 10 tasks proposed by two of the authors, we have analyzed 102 tasks and 23 educational aims obtained in this research. We have generated the summary concept lattice and found the following observations:

- 45 tasks (the first row in the summary concept lattice) are the representatives; i.e. every task includes the unique set of aims and there is no task that introduces the superset of these aims,
- 5 tasks (from 45 representatives) are such that every task includes the unique set of aims and there is no task that introduces neither superset nor subset of these aims,
- 3 aims (the first row in the summary concept lattice) are unique, i.e. the aim is introduced only by one task,
- the most frequent aims in general are Aim6 (80% of tasks), Aim7 (63%), Aim10 (51%), Aim8 (46%), Aim4 (35%) which represent the basic declaration, read, write out, manipulation and applications of an array data type in the simple tasks,
- the most frequent aims including applications of an array data type in the more difficult situations are Aim11c (22%), Aim11d (22%), Aim12 (20%) and Aim11a (19%); the applications in a sorting and a simple game programming are the least represented from these group of aims.

In effort to prepare the graduated sets of tasks, we have explored the longest paths extracted from the summary concept lattice with reduced labeling of all 102 tasks. Some of the longest paths are shown in Fig. 4. Every path contains the graduated system of tasks depending on the final task we want to achieve in conclusion. The object label, for instance 3.5, corresponds to the fifth task of third teacher. The set of tasks labeled 6.1 – 6.10 comes from the authors.

The paths have different lengths, because there are nodes in the summary concept lattice with reduced labeling, which do not contain neither task or aim label (own object or own attribute). Therefore, these nodes are omitted in the extracted longest paths shown in Fig. 4. Nevertheless, some of the paths can have the same length and can differ only in a small number of tasks. To capture two or more paths (which are similar in this sense) by one figure, we display also not linear cases (b), (c) in Fig. 4.

Fig. 4. The longest paths extracted from the summary concept lattice



(a) (b) (c) (d)

The longest paths are recommended to apply in a diagnostic phase of an educational process. For instance, the longest path (a) illustrates that if a student has a problem with Task6.5, we ask him/her to solve Task3.12. Moreover, if we have found that a student has a problem with Aim6 in Task6.5, we give him/her to fixation one task (or more) from the node which contains the set of equal tasks Task1.3, Task1.4, Task1.5, Task1.8. In contrary, if a student has no problem with Task6.5, we suppose that he/she will pass also the Task3.12. There is only one path including seven task nodes, however paths with six nodes appear in the summary concept lattice several times. The cases (b) and (c) were chosen to cover the most frequent aims by the combination of (a),(b),(c) cases. Moreover, the case (b) shows that if a student has a problem with Task6.4, we can choose either Task1.3 or Task 4.1 in order to cover the aim that was not fulfilled by a student. We can also extract some other (not compulsory the longest) paths starting with other initial aims and different initial tasks extracted from the summary concept lattice. The path (d) presents an example of the five nodes path starting with Task 4.3 in contrary with an initial task of the previously described paths.

As a conclusion, we propose to supplement four of the input objectives and to add two additional objectives (mainly for advanced students) into the input system of objectives as follows:

5) original form supplemented by: the first array index is not necessary 0 or 1,
8) original form supplemented by: find a presence of some value in a an array,
9) original form supplemented by: errors related to a clear of an array,
10) original form supplemented by: an array as a parameter of procedure,

13) to apply a dynamic array in the simple issues,
14) to apply a two-dimensional array in the simple issues.

Moreover, we present some interesting educational tasks which appear in the summary concept lattice mostly in the first row and one can advise them to apply in the educational process related to an array data structure. The formulations are shortened in comparison with the original author's texts.

- Propose the way how to denote the parking places in front of a hotel. How are the train carriages enumerated? How would you denote the overall and final results of six teams in the television knowledge contest?
- We have observed GPS data containing ten altitudes on our tourist route. Write a program to print out the altitudes on a reverse route.
- Imagine that you have received SMS from your friend. Write a program to count the number of words in your text message.
- A musical instrument, like a piano, can be simulated by a computer program. Some of the keys will have assigned a particular tone frequency. Write a program to play a tone when the particular key is pressed.
- Consider the starting sequence of children names and the final shift of Ferris wheel as the input. Write a program to make a list of the children names in the sequence in which they will get out Ferris wheel.
- Write a program to generate twelve random values expressing the number of your website visits in a particular month. Draw a histogram, highlight the maximum and minimum and show an average value as a horizontal line.

6 App Inventor concept lattice

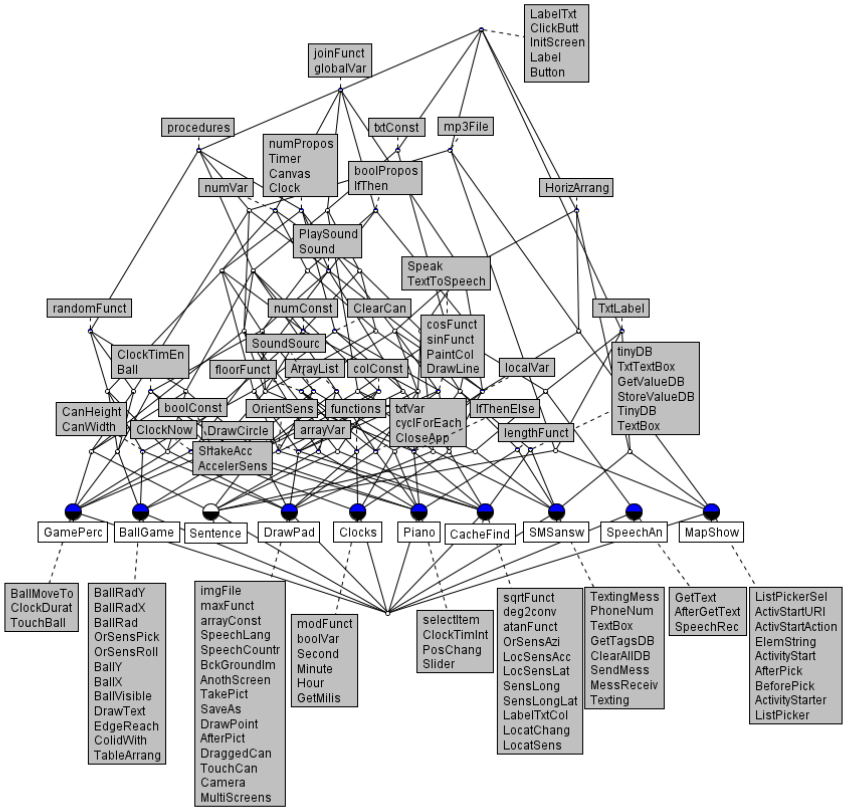
We have fruitfully applied formal concept analysis as a powerful tool in a simultaneous analysis that involves the teaching of programming skills in an open-source web application App Inventor 2. The tutorial website[2] provides materials in the form of learning cards for building the basic applications, but one of the authors of this paper has prepared the set of ten complex educational tasks which in summary cover 129 elements (components and their elements, event handlers, call, set instructions, get instructions, data structures, etc.) available at the present time. The added value includes the proposal of the introductory set of complex tasks and its further modification in effort to teach and learn the different target groups. The talented lower secondary school's pupils participate in our optional university courses and the teachers of secondary schools attend the didactic workshops at our university. Our results are concerned with the inclusion of the programming language elements (available at the App Inventor website at present) in the complex educational tasks and the effort to extract the appropriate tasks for the different types of an educational process. The formal context contains 10 tasks as the set objects and 129 App Inventor programming elements as the set of attributes.

[2] http://www.appinventor.org/

Exploring own attributes, the resulting concept lattice and its attribute labels shown in Fig. 5 give an information about the elements introduced uniquely by a particular task. As conclusion, we recommend the following methodology:

- a task with a high ratio of the own elements and the low total number of elements is advised to use in a first acquisition phase of an education,
- a task with a low ratio of the own elements and the low total number of elements is recommended to use in a fixation phase of an educational process,
- a task with a low ratio of the own elements and the high total number of elements is suggested in a systematization or diagnostic educational phase,
- a task with a high ratio of the own elements and the high total number of elements is the least appropriate for an educational process, because it brings many new elements without their introduction in a more simple task.

Fig. 5. A concept lattice of App Inventor 2 programming in education



7 Conclusion

We have investigated the educational tasks and objectives of five real teachers giving lessons in computer science. Extracted paths from the summary concept lattice seem to be based on the similar idea as in the learning paths from knowledge space theory introduced by Doignon and Falmagne [13]. The peer instruction is a learning method in which the results can be applied, as well.

The learning process of students in computer science is also concerned in the work of Uta Priss [31, 32]. The tools developed for learning and teaching in combination with curricula and teaching practices are aiming at actual project weSPOT at TU Graz with applying the formal concept analysis [33]. The triadic version of formal concept analysis [3, 15] seems to be fruitful for analyzing the concordance of the teachers in our future work.

References

1. C. Alcalde, A. Burusco, R. Fuentes-González, I. Zubia, The use of linguistic variables and fuzzy propositions in the L-fuzzy concept theory, Comput. Math. Appl. 62 (8) (2011) 3111-3122.
2. L. Antoni, S. Krajči, O. Krídlo, B. Macek, L. Pisková, On heterogeneous formal contexts, Fuzzy Sets Syst. 234 (2014) 22–33.
3. R. Bělohlávek, C. Glodeanu, V. Vychodil, Optimal Factorization of Three-Way Binary Data Using Triadic Concepts, Order 30 (2013) 437–454.
4. R. Bělohlávek, Sup-t-norm and inf-residuum are one type of relational product: unifying framework and consequences, Fuzzy Sets Syst. 197 (2012) 45–58.
5. R. Bělohlávek, Ordinally equivalent data: A measurement-theoretic look at formal concept analysis of fuzzy attributes, Int. J. Approx. Reason. 54 (9) (2013) 1496–1506.
6. K. Bazhanov, S. Obiedkov, Optimizations in computing the Duquenne-Guigues basis of implications, Ann. Math. Artif. Intell. 70 (2014) 5–24.
7. P. Butka, J. Pócs, J. Pócsová, On equivalence of conceptual scaling and generalized one-sided concept lattices, Inf. Sci. 259 (2014) 57–70.
8. P. Butka, J. Pócs, J. Pócsová, Representation of fuzzy concept lattices in the framework of classical FCA, J. Appl. Math. (2013) Article ID 236725, 7 pages.
9. P. Butka, J. Pócs, Generalization of one-sided concept lattices, Comput. Inform. 32 (2) (2013) 355–370.
10. I. P. Cabrera, P. Cordero, G. Gutiérez, J. Martinez, M. Ojeda-Aciego, On residuation in multilattices: Filters, congruences, and homorphisms, Fuzzy Sets Syst. 234 (2014) 1–21.
11. C. Carpineto, G. Romano, Concept Data Analysis Theory and Applications, J. Wiley, 2004.
12. M. E. Cornejo, J. Medina, E. Ramírez, A comparative study of adjoint triples, Fuzzy Sets Syst. 211 (2013) 1–14.
13. J. P. Doignon, J. C. Falmagne, Knowledge Spaces, Springer Verlag, 1999.
14. B. Ganter, R. Wille, Formal Concept Analysis Mathematical Foundation, Springer Verlag, 1999.
15. J. Konečný, P. Osička, Triadic concept lattices in the framework of aggregation structures, Inf. Sci. 279 (2014) 512–527.

16. S. Krajči, A generalized concept lattice, Logic J. IGPL, 13 (2005) 543–550.
17. S. Krajči, The basic theorem on generalized concept lattice, in: V. Snášel, R. Bělohlávek (Eds.), Proceed. of the 2nd Intern. Conf. CLA 2004, pp. 25–33.
18. D.R. Krathwohl, A revision of Bloom's taxonomy: an overview, Theor. Pract. 41 (4) (2012) 212-218.
19. O. Krídlo, S. Krajči, M. Ojeda-Aciego, The category of L-Chu correspondences and the structure of L-bonds, Fund. Inform. 115 (4) (2012) 297-325.
20. J. Medina, M. Ojeda-Aciego, Multi-adjoint t-concept lattices, Inf. Sci. 180 (2010) 712–725.
21. J. Medina, M. Ojeda-Aciego, On multi-adjoint concept lattices based on heterogeneous conjunctors, Fuzzy Sets Syst. 208 (2012) 95–110.
22. J. Medina, M. Ojeda-Aciego, Dual multi-adjoint concept lattices, Inf. Sci. 225 (2013) 47–54.
23. J. Medina, M. Ojeda-Aciego, J. Ruiz-Calviño, Formal concept analysis via multi-adjoint concept lattices, Fuzzy Sets Syst. 160 (2009) 130–144.
24. J. Medina, M. Ojeda-Aciego, A. Valverde, P. Vojtáš, Towards biresiduated multi-adjoint logic programming, Lect. Notes Artif. Intell. 3040 (2004) 608–617.
25. J. Medina, M. Ojeda-Aciego, P. Vojtáš, Multi-adjoint logic programming with continuous semantics, Lect. Notes Artif. Intell. 2173 (2001) 351–364.
26. J. Medina, M. Ojeda-Aciego, P. Vojtáš, Similarity-based unification: a multi-adjoint approach, Fuzzy Sets Syst. 146 (2004) 43–62.
27. J. Pócs, Note on generating fuzzy concept lattices via Galois connections, Inf. Sci. 185 (2012) 128–136.
28. J. Pócs, On possible generalization of fuzzy concept lattices using dually isomorphic retracts, Inf. Sci. 210 (2012) 89–98.
29. J. Poelmans, D. I. Ignatov, S. O. Kuznetsov, G. Dedene, Formal concept analysis in knowledge processing: A survey on applications, Expert Syst. Appl. 40(16) (2013) 6538–6560.
30. J. Poelmans, G. Dedene, G. Verheyden, H. Van der Mussele, S. Viaene, E. Peters, Combining business process and data discovery techniques for analyzing and improving integrated care pathways, in: P. Perner (Ed.): ICDM 2010, Springer-Verlag Berlin Heidelberg, 2010, pp. 505–517.
31. U. Priss, Using FCA to Analyse How Students Learn to Program, in: P. Cellier, F. Distel, B. Ganter (Eds.), Proceedings of the 11th International Conference ICFCA 2013, Dresden, Germany, Springer Berlin Heidelberg, 2013, pp. 216–227.
32. U. Priss, P. Riegler, N. Jensen, Using FCA for Modelling Conceptual Difficulties in Learning Processes, in: F. Domenach, D. I. Ignatov, J. Poelmans (Eds.), Contrib. to the 10th Intern. Conf. ICFCA 2012, Leuven, Belgium, 2012, pp. 161–173.
33. A. Protopsaltis, P. Seitlinger, F. Chaimala, O. Firssova, S. Hetzner, K. Kikis-Papadakis, P. Boytchev, Working Environment with Social and Personal Open Tools for inquiry based learning: Pedagogic and Diagnostic Frameworks, Int. J. Sci. Math. Technol. Learn., in press.
34. L. Šnajder, Inquiry approach in learning selected computer science concepts, in: SMEC 2012 : Science and Mathematics Education Conference : Teaching at the heart of learning, Dublin (Ireland), Dublin City University, 2012, pp. 199-204.
35. L. Šnajder, J. Guniš, Inquiry based learning of selected computer sciences concepts and principles, ICTE Journal 1(1) (2012) 28–39.
36. CS Principles: Computation in Action Preview Curriculum, http://csta.acm.org/Curriculum/sub/CurrResources.html
37. The national curriculum in England, framework document, 2013, 224 pages, http://www.education.gov.uk/nationalcurriculum

Pattern Structures for Understanding Episode Patterns

Keisuke Otaki*, Ikeda Madori and Akihiro Yamamoto

Department of Intelligence Science and Technology
Graduate School of Informatics, Kyoto University, Japan.
{ootaki,m.ikeda}@iip.ist.i.kyoto-u.ac.jp, akihiro@i.kyoto-u.ac.jp

Abstract. We investigate an application of pattern structures for understanding episodes, which are labeled directed acyclic graphs representing event transitions. Since typical episode mining algorithms generate a huge number of similar episodes, we need to summarize them or to obtain compact representations of them for applying the outcome of mining to various problems. Though such problems have been well-studied for itemsets, summarization of episodes is still understudied. For a class called diamond episodes, we first provide a pattern structure based on hierarchy of events to obtain small groups of episodes in the form of pattern concepts and lattice structures. To find a summary via pattern concepts, we design an utility function for scoring concepts. After ranking concepts using some function and lattice structures, we try to sample a set of pattern concepts of high scores as a summary of episodes. We report our experimental results of our patten structure, and a ranking result of our simple utility function. Last we discuss pattern concept lattices and their applications for summarization problems.

Keywords: formal concept analysis, pattern structure, episode pattern, pattern summarization

1 Introduction

Knowledge Discovery from binary databases is a fundamental problem setting, where binary databases represent that some objects have some features by their 1 entries. Because such a situation can be seen in many practical problems, both theoretical and practical aspects of the problems have been studied.

On a mathematical viewpoint, Formal Concept Analysis (FCA) [5] has been studied as a model of analyzing such binary databases. We deal with a *context* $K = (O, A, I)$ consisting of 1) a set O of objects, 2) a set A of attributes, and 3) a binary relation $I \subseteq O \times A$ representing that an i-th object has a j-th attribute. FCA adopts two functions f and g for analyzing O and A; f receives a set of objects and returns a set of attributes which are commonly possessed by given objects, and g receives a set of attributes and returns a set of objects which

* Keisuke Otaki is supported as a JSPS research fellow (DC2, 26·4555).

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

have commonly the input attributes. For $X \subseteq O$ and $Y \subseteq A$, a tuple (X, Y) is called a *concept* if $f(X) = Y$ and $X = g(Y)$. Computing the set of all concepts is a fundamental but important task in FCA, which help us to analyze binary databases. On a practical viewpoint, it is well-known that formal concepts are related to closed itemsets studied in frequent itemset mining [13], which are also known as compact representations of itemsets.

To deal with non-binary data in an FCA manner, pattern structures [4] have been studied. A key idea is generalizing both the set intersection $\cap$ and the subset relation $\subseteq$, which are used in two functions f and g in FCA. The set intersection $\cap$ is replaced with a meet operator $\sqcap$ that extracts *common substructures* of two objects. The subset relation $\subseteq$ is also replaced with a partial order $\sqsubseteq$ induced by $\sqcap$, where $\sqsubseteq$ represents some *embedding* from an object into another. We now assume that they would help us to understand complex data.

In this paper, we investigate pattern structures and their applications for understanding patterns, motivated by a requirement of summarization techniques because a large numbers of patterns is always generated by some mining algorithms. As an example in this paper, we deal with some classes of *episode patterns*, which represent event transitions in the form of labeled graphs. From such patterns, we can compute lattice structures based on pattern structures (Section 3). Since such lattices represent mutual relations among patterns and several small clusters as pattern concepts, analyzing them would be helpful to obtain a small set of pattern concepts. We regard a subset of all concepts as a summary of features often used in describing patterns, and develop a way of obtaining a small set of concepts as a summary. When we construct descriptions of objects, we also introduce the wildcard $\star$ as a special symbol representing all events to take into account some hierarchy of labels based on our knowledge. It would be a strong merit of pattern structures for summarization in which similar patterns could be merged into some descriptions with the wildcard $\star$. After providing pattern structures, we provide preliminary experimental results (Section 4) and discuss it on the viewpoint of summarization by giving a utility function for ranking pattern concepts (Section 5).

2 Formal Concept Analysis and Episode Mining

FCA and Pattern Structures We adopt the standard notations of FCA from [5] and pattern structures from [4], respectively. Here we refer the notations of FCA which we have already used in Section 1. For a context $K = (O, A, I)$, $X \subseteq O$ and $Y \subseteq A$, two functions f and g in FCA are formally defined by $f(X) = \{a \in A \mid (o, a) \in I \text{ for all } o \in X\}$ and $g(Y) = \{o \in O \mid (o, a) \in I \text{ for all } a \in Y\}$, respectively. Recall that a pair (X, Y) is a *(formal) concept* if $f(X) = Y$ and $g(Y) = X$. Two operators $f \circ g(\cdot)$ and $g \circ f(\cdot)$ are *closure operators* on 2^O and 2^A, respectively. Note that a concept (X, Y) is in the form either $(g(f(X)), f(X))$ or $(g(Y), f(g(Y)))$. For two concepts (X_1, Y_1) and (X_2, Y_2), the partial order $\leq$ is introduced by $X_1 \subseteq X_2$ ($\Leftrightarrow Y_2 \subseteq Y_1$).

An important aspect of pattern structures is generalization of two operations $\cap$ and $\subseteq$ used in f and g. They are characterized by *meet semi-lattices*: A *meet semi-lattice* $(D, \sqcap)$ of a set D and a meet operator $\sqcap$ is an algebraic structure satisfying: 1) Associativity; $x \sqcap (y \sqcap z) = (x \sqcap y) \sqcap z$ for $x, y, z \in D$, 2) Commutativity; $x \sqcap y = y \sqcap x$ for $x, y \in D$, and 3) Idempotency; $x \sqcap x = x$ for $x \in D$. Elements in D are called *descriptions*. A partial order $\sqsubseteq$ is induced by $\sqcap$ as

$$x \sqsubseteq y \text{ whenever } x \sqcap y = x \text{ for two elements } x, y \in D.$$

Example 1 (A meet semi-lattice with sets). Let $D = 2^{\mathbb{N}}$. For two sets of integers $X = \{1, 2\}$ and $Y = \{1, 2, 3, 4\}$, it holds that $X \sqcap Y = X$, which induces $X \subseteq Y$.

Example 2 (A meet semi-lattice for closed intervals [8]*).* Let D be a set of all closed intervals $[a, b]$ with integers $a, b \in \mathbb{N}$. They define $\sqcap$ of two closed intervals by keeping its convexity, that is, $[a_1, b_1] \sqcap [a_2, b_2] = [\min(a_1, a_2), \max(b_1, b_2)]$.

By generalizing a meet semi-lattice $(2^A, \cap)$ used in FCA, a pattern structure $\mathbb{P}$ is defined by a triple $(O, (D, \sqcap), \delta)$, where O is the set of objects, $(D, \sqcap)$ is a meet semi-lattice of *descriptions*, and $\delta : O \to D$ is a mapping of giving a description for each object. For analyzing pattern structures, we obtain the following Galois connection $\{(\cdot)^{\square}, (\cdot)^{\diamond}\}$ corresponding to f and g in FCA:

$$A^{\square} = \sqcap_{o \in A} \delta(o) \text{ for } A \subseteq O, \tag{1}$$

$$d^{\diamond} = \{o \in O \mid d \sqsubseteq \delta(o)\} \text{ for } d \in D. \tag{2}$$

Pattern concepts based on $\mathbb{P}$ are defined by Equations (1) and (2):

Definition 1 (Pattern concepts) A *pattern concept* of $\mathbb{P}$ is a pair (A, d) of a set $A \subseteq O$ and a pattern $d \in D$ satisfying $A^{\square} = d$ and $d^{\diamond} = A$. Two pattern concepts are partially ordered by $(A_1, d_1) \preceq (A_2, d_2)$ by $A_1 \subseteq A_2 \Leftrightarrow d_2 \sqsubseteq d_1$.

Note that by its partial order, the set of all pattern concepts forms a lattice structure. We denote the set of all pattern concepts by $\mathfrak{P}(\mathbb{P})$. To obtain $\mathfrak{P}(\mathbb{P})$, we need to compute two functions $(\cdot)^{\square}$ and $(\cdot)^{\diamond}$. For example, we can adopt the AddIntent proposed in [12] and used in [8].

Episode Mining We briefly review episode mining based on [7]. Let $\mathcal{E} = \{1 \ldots, m\} \subseteq \mathbb{N}$ be the set of *events*. We call a set $S \subseteq \mathcal{E}$ of events an *event set*. An input of episode mining is a long sequence of event sets; an *input event sequence* $\mathcal{S}$ on $\mathcal{E}$ is a finite sequence $\langle S_1, \ldots, S_n \rangle \in (2^{\mathcal{E}})^*$, where each set $S_i \subseteq \mathcal{E}$ is the i-th event set. For $\mathcal{S}$ of length n, we assume that $S_i = \emptyset$ if $i < 0$ or $i > n$.

Episodes are *labeled directed graphs* (DAGs). An episode G is a triple (V, E, λ), where V is the set of vertices, E is the set of directed edges, and λ is the labeling function from V and E to the set of labels, that is, $\mathcal{E}$. Several classes of episodes have been studied since episode mining is firstly introduced by Mannila *et. al.* [11]. We follow subclasses of episodes studied by Katoh *et al.* [7]. An

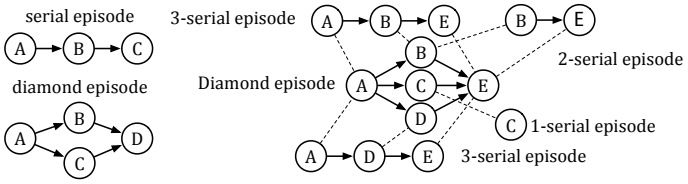


Fig. 1. An example of episode studied in episode mining in [11] and [7].

example of episodes is illustrated in Figure 1. In designing pattern mining algorithms, we need 1) a search space of patterns and a partial order for enumerating patterns, and 2) interestingness measure to evaluate them. For episode mining, we often adopt *occurrences* of episodes defined with *windows*.

Definition 1 (Windows). For a sequence $\mathcal{S} = \langle S_1, \ldots, S_n \rangle$, an *window* W of $\mathcal{S}$ is a contiguous subsequence $\langle S_i, \cdots, S_{i+w-1} \rangle$ of length n, called *width*, for some index i $(-w+1 \leq i \leq n)$ of $\mathcal{S}$ and a positive integer $w \geq 0$.

Definition 2 (Embedding of Episodes). Let $G = (V, E, \lambda)$ be an episode, and $W = \langle S_1, \ldots, S_w \rangle$ be a window of width w. We say that G *occurs* in W if there exists a mapping $h : V \to \{1, \ldots, w\}$ satisfying 1) for all $v \in V$, $h(v) \in S_{h(x)}$, and 2) for all $(u, v) \in E$ with $u \neq v$, it holds that $h(u) < h(v)$. The map h is called an *embedding* of G into W, and it is denoted by $G \preceq W$.

For an input event sequence $\mathcal{S}$ and a episode G, we say that G occurs at position i of $\mathcal{S}$ if $G \preceq W_i$, where $W_i = \langle S_i, \ldots, S_{i+w-1} \rangle$ is the i-th window of width w in $\mathcal{S}$. We then call the index i an *occurrence* of G in $\mathcal{S}$. The *domain* of the occurrences is given by $\mathbf{W}_{\mathcal{S},w} = \{i \mid -w+1 \leq i \leq n\}$. In addition, $\mathbf{W}_{\mathcal{S},w}(G)$ is the *occurrence window list* of an episode G, defined by $\{-w+1 \leq i \leq n \mid G \preceq W_i\}$. Then we can define an interestingness measure *frequency* of episodes.

Definition 3 (Frequency of Episodes). The *frequency* of an episode G in $\mathcal{S}$ and w, denoted by $\mathit{freq}_{\mathcal{S},w}(G)$, is defined by the number of windows of width w containing G. That is, $\mathit{freq}_{\mathcal{S},w}(G) = |\mathbf{W}_{\mathcal{S},w}(G)|$. For a threshold $\theta \geq 1$, a width w and an input event sequence $\mathcal{S}$, if $\mathit{freq}_{\mathcal{S},w}(G) \geq \theta$, G is called *θ-frequent* on $\mathcal{S}$.

The frequent episode mining problem is defined as follows: Let $\mathcal{P}$ be a class of episodes. Given an input event sequence $\mathcal{S}$, a width $w \geq 1$, and a frequency threshold $\theta \geq 1$, the problem is to find all θ-frequent episodes G belonging to the class $\mathcal{P}$. The simplest strategy of finding all θ-frequent episodes is traversing $\mathcal{P}$ by using the anti-monotonicity of the frequency count $\mathit{freq}(\cdot)$. For details, we would like to refer to both [7] and [11].

For our examples of classes, we introduce m-serial episodes and diamond episodes. An *m-serial episode* over $\mathcal{E}$ is a sequence of events in the form of $a_1 \mapsto a_2 \mapsto \cdots \mapsto a_m$. A *diamond episode* over $\mathcal{E}$ is either 1) a 1-serial episode $e \in \mathcal{E}$ or 2) a *proper diamond episode* represented by a triple $Q = \langle a, X, b \rangle \in \mathcal{E} \times 2^{\mathcal{E}} \times \mathcal{E}$, where a, b are events and $X \subseteq \mathcal{E}$ is an event set occurring after a and before

b. For short, we write a diamond episode as $a \mapsto X \mapsto b$. On the one hand definitions of episodes by graphs are much general, on the another hand classes of episode patterns are often restricted.

Example 3 (Episodes). In Figure 1, we show some serial episodes; $A \mapsto B \mapsto E$, $A \mapsto D \mapsto E$, $B \mapsto E$, and C on the set of events $\mathcal{E} = \{A, B, C, D, E\}$. All of them are included in a diamond episode $A \mapsto \{B, C, D\} \mapsto E$.

We explain a merit of introducing pattern structures for summarization of structured patterns. As we mentioned above, a common strategy adopted in pattern mining is traversing the space $\mathcal{P}$ in a breadth-first manner with checking some interestingness measure. When generating next candidates of frequent patterns, algorithms always check a *parent-child* relation between two patterns. This order is essential for pattern mining and we thus conjecture that this parent-child relation used in pattern mining can be naturally adopted in constructing a pattern structure for analyzing patterns only by introducing a similarity operation $\sqcap$. After constructing a lattice, it would be helpful to analyze a set of all patterns using it because they represent all patterns compactly.

A crucial problem of pattern structures is the computational complexity concerning both $\sqcap$ and $\sqsubseteq$. Our idea is to adopt trees of height 1 (also called *stars* in Graph Theory). That is, we here assume that trees are expressive enough to represent features of episodes. Our idea is similar that used in designing graph kernels [14][1] and that is inspired by previous studies on pattern structures [2, 4].

3 Diamond Episode Pattern Structures

In the following, we focus on diamond episodes as our objects, and *trees of height* 1 as our descriptions. They have two special vertices; the source and the sink. They can be regarded as important features for representing event transitions. We generate rooted labeled trees from them by putting the node in the root of a tree, and regarding neighbors as children of it. Since heights of all trees here are 1, we can represent them by tuples without using explicit graph notations.

Definition 4 (Rooted Trees of Height 1). Let $(\mathcal{E}, \sqcap_{\mathcal{E}})$ be a meet semi-lattice of event labels. A rooted labeled tree of height 1 is represented by a tuple [2] $(e, C) \in \mathcal{E} \times 2^{\mathcal{E}}$. We represent the set of all rooted labeled trees of height 1 by $\mathfrak{T}$.

Note that in $(\mathcal{E}, \sqcap_{\mathcal{E}})$, we assume that $\sqcap_{\mathcal{E}}$ compares labels based on our background knowledge. We need to take care that this meet semi-lattice $(\mathcal{E}, \sqcap_{\mathcal{E}})$ is independent and different from a meet semi-lattice $\underline{\mathrm{D}}$ of descriptions of a pattern structure $\mathfrak{P}$. This operation $\sqcap_{\mathcal{E}}$ is also adopted when defining an embedding of trees of height 1, that is, a partial order between trees defined as follows.

[1] It intuitively generates a sequence of graphs by relabeling all vertices of a graph. One focus on a label of a vertex $v \in V(G)$ and sees labels $LN_G(v)$ of its neighbors $N_G(v)$. For a tuple $(l_v, LN_G(v))$ for all vertices $v \in V(G)$, we sort all labels lexicographically, and we assign a new label according to its representation. Details are seen in [14].

[2] On the viewpoint of graphs, this tuple (e, C) should represent a graph $G = (V, E, \lambda)$ of $V = \{0, 1, \ldots, |C|\}$, $E = \{(0, i) \mid 1 \leq i \leq |C|\}$, $\lambda(0) = e$, $\{\lambda(i) \mid 1 \leq i \leq |C|\} = C$.

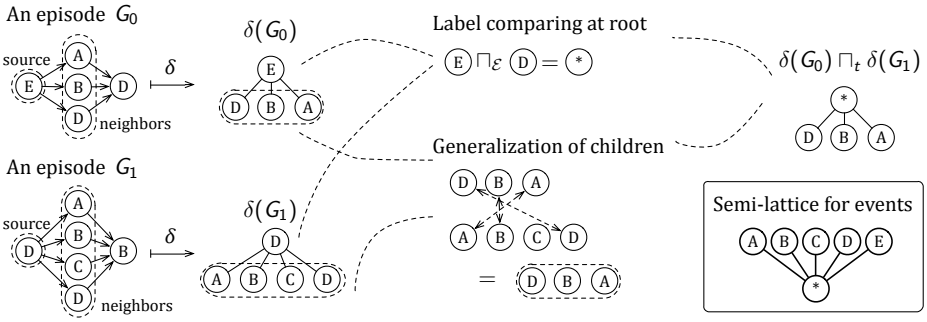


Fig. 2. An example of computations $\sqcap$ of two trees of height 1.

Definition 5 (Partial Order on Trees). A tree $t_1 = (e_1, C_1)$ is a *generalized subtree* of $t_2 = (e_2, C_2)$, denoted by $t_1 \sqsubseteq_T t_2$, iff $e_1 \sqsubseteq_{\mathcal{E}} e_2$ and there exists an injection mapping $\phi : C_1 \to C_2$ satisfying for all $v \in C_1$, there exists $\phi(v) \in C_2$ satisfying $v \sqsubseteq_{\mathcal{E}} \phi(v)$, where $\sqsubseteq_{\mathcal{E}}$ is the induced partial order by $\sqcap_{\mathcal{E}}$.

For defining a similarity operator $\sqcap_T$ between trees, this partial order $\sqsubseteq_T$ is helpful because $\sqcap_T$ is closely related to $\sqsubseteq_T$ in our scenario. Since all trees here are height 1, this computation is easy to describe; For labels of root nodes, a similarity operator is immediately given by using $\sqcap_{\mathcal{E}}$. For their children, it is implemented by using an idea of *least general generalization* (LGG), which is used in Inductive Logic Programming [10], of two sets of labels. A practical implementation of LGG depends on whether or not sets are *multisets*, but it is computationally tractable. An example is seen in Figure 2.

We give formal definitions of δ and D. For a graph $G = (V, E, \lambda)$, we denote the neighbors of $v \in V$ by $N_G(v)$. For some proper diamond episode pattern G, the source vertex $s \in V$ and the sink vertex $t \in V$, computed trees of height 1 corresponding s and t are defined as $T_s = (\{s\} \cup N_G(s), \{(s, u) \mid u \in N_G(s)\}, \lambda)$, and $T_t = (\{t\} \cup N_G(t), \{(u, t) \mid u \in N_G(t)\}, \lambda)$, respectively. By using those trees, $\delta(\cdot)$ can be defined according to vertices s and t: If we see both T_s and T_t, $\delta(G) = (T_s, T_t)$ and then $\sqcap_T$ is adopted element-wise, and D is defined by $\mathfrak{T} \times \mathfrak{T}$. If we focus on either s or t, $\delta(G) = T_s$ or T_v, and we can use $\sqcap_T$ directly by assuming $D = \mathfrak{I}$.

Last we explain relations between our pattern structures and previous studies shortly. This partial order $\sqsubseteq_T$ is inspired from a generalized subgraph isomorphism [4] and a pattern structure for analyzing sequences [2]. We here give another description of similarity operators based on definition used in [4, 9].

Definition 6 (Similarity Operation $\sqcap$ based on [9]). The similarity operation $\sqcap$ is defined by the set of all maximal common subtrees based on the generalized subtree isomorphism $\sqsubseteq_T$; For two trees s_1 and s_2 in $\mathfrak{T}$,

$$s_1 \sqcap s_2 \equiv \{u \mid u \sqsubseteq_T s_1, s_2, \text{ and } \forall u' \sqsubseteq_T s_1, s_2 \text{ satisfying } u \not\sqsubseteq_T u'\}.$$

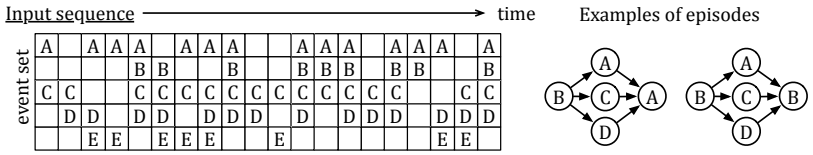


Fig. 3. An input $\mathcal{S}$ and two diamond episodes mined from $\mathcal{S}$ as examples.

Table 1. Numbers of proper diamond episodes and pattern concepts for $w \in \{3, 4, 5\}$ and $M \in \{100, 200, 300, 400, 500, 600, 700\}$. In the table below, DE and PDE means Diamond Episodes and Proper Diamond Episodes, respectively.

			M and # of pattern concepts						
Window width w	# of DE	# of PDE	100	200	300	400	500	600	700
3	729	569	87	137	178	204	247	–	–
4	927	767	74	136	179	225	281	316	336
5	935	775	71	137	187	272	290	313	342

Note that we can regard that our operator $\sqcap_T$ is a special case of the similarity operation $\sqcap$ above. On the viewpoint of pattern structures, our trees of height 1 can be regarded as an example of projections from graphs into trees, studied in [4, 9], such as both k-chains (paths on graphs of length k) and k-cycles.

4 Experiments and Discussion for Diamond Episodes

Data and Experiments We gathered data from MLB baseball logs, where a system records all pitching and plays for all games in a season. We used what types of balls are used in pitching, which can be represented by histograms per batter. For a randomly selected game, we generated an input event sequence of episode mining by transforming each histogram to a set of types of balls used types of balls[3]. In forming $(\mathcal{E}, \sqcap_{\mathcal{E}})$, we let $\mathcal{E}$ be the set of types of balls, and define $\sqcap_{\mathcal{E}}$ naturally (See Example in Fig. 2). For this $\mathcal{S}$, we applied a diamond episode mining algorithm proposed by [7] and obtain a set of diamond episodes. The algorithm have two parameters; the window size w and the frequency threshold θ. We always set $\theta = 1$ and varied $w \in \{3, 4, 5\}$. After generating a set $\mathcal{G}$ of frequent proper diamond episodes, we sampled $M \in \{100, 200, \ldots, 700\}$ episodes from $\mathcal{G}$ as a subset O of $\mathcal{G}$ (that is, satisfying $|O| = M$ and $O \subseteq \mathcal{G}$). We used O as a set of objects in our pattern structure $\mathbb{P}$. From it we computed all pattern concepts $\mathfrak{P}(\mathbb{P})$ based on our discussions in Section 3. In this experiments we set $\delta(G) = T_s$ for a proper diamond episode G and its source vertex s.

[3] In baseball games, pitchers throw many kinds of balls such as fast balls, cut balls, curves, sinkers, etc. They are recorded together with its movements by MLB systems.

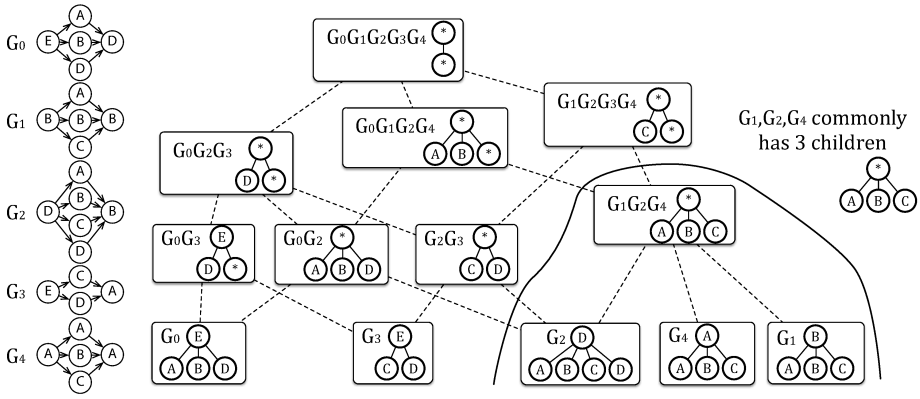


Fig. 4. A small example of a pattern structure lattice.

For experiments, we adopted an implementation by Katoh *et al.* [7] for mining diamond episodes, which is written in C++[4]. We implemented the Galois connection $\{(\cdot)^{\square}, (\cdot)^{\diamond}\}$ by the AddIntent [12] algorithm using Python[5].

Results and Discussion Table 1 shows the results of numbers of pattern concepts and those of proper diamond episodes for each set corresponding $w = 3, 4$, and 5, with varying $M \in \{100, 200, 300, 400, 500, 600, 700\}$. In Figure 4, we show a result of pattern concepts and a pattern concept lattice computed from only first 5 episodes for the case $w = 5$, as an example of our lattice structures.

Because we assume a semi-lattice $(\mathcal{E}, \sqcap_{\mathcal{E}})$ of events in episode patterns, we can obtain pattern concepts in which some vertices are represented by the wildcard $\star$. If we implement $\sqcap_t$ without the wildcard $\star$, we can only obtain a much smaller number of pattern concepts compared with our results including the wildcard $\star$. We thus conjecture that the wildcard $\star$ is useful to represent similar patterns in Figure 4. On such a viewpoint, pattern structures and pattern concepts help us to make some group of patterns, which are basically similar to clustering patterns. Here we do not discuss details of computational complexity of constructing pattern concept lattices, but the complexity basically depends on the number of concepts. Thus it would be interesting to investigate and compare several possible projections and computations of pattern concepts.

5 Pattern Summarization and Pattern Structures

In this section, we discuss pattern summarization based on pattern concept lattices. As we mentioned, closed itemsets [13] have been studied as compact

[4] We adopt gcc4.7 as our compiler including c++11 features (-std=c++11). The code is compiled on a machine of Mac OS X 10.9 with two 2.26 GHz Quad-Core Intel Xeon Processors and 64GB main memory.

[5] We used Python 2.7 without any additional packages (that is, pure python).

representations of itemsets, and they are closely related to the closure operator $g \circ f$ in FCA with (O, A, I), where O is the set of transaction identifiers and A is the set of all items. The difficulty of closed patterns for complex data is there are no common definitions of closure operators, where we usually use the closeness with respect to the frequency. Here we assume that pattern concepts are helpful in the same correspondence between closed itemsets and concepts.

To obtain some compact representations, we need to decide *how to evaluate each pattern*. The problem here is how to deal with the wildcard $\star$ in descriptions. When we obtain a concept (X, Y) for $X \subseteq O, Y \subseteq A$, this concept (X, Y) corresponds to a rectangle on I, and there are no 0 entries in the sub-database $I' = \{(x, y) \in I \mid x \in X, y \in Y\}$ of I induced by (X, Y) because of its definitions. If (X, Y) is not a concept, a rectangle r by (X', Y') contains a few 0 entries in it. We denote the relative ratio of 1 entries in a rectangle r by (X', Y') as

$$\mathbf{r}_1(X', Y', I) = (1 - |\{(x, y) \notin I \mid x \in X', y \in Y'\}|) \left(|X'||Y'|\right)^{-1},$$

where $0 \leq \mathbf{r}_1(X', Y', I) \leq 1$ and $\mathbf{r}_1(X', Y', I) = 1$ if (X', Y') is a concept. These $\mathbf{r}_1(X, Y, I)$, $|X|$, and $|Y|$ are applicable for evaluating itemsets. If we only use the cardinality $|A|$ of a set A of objects, this equals to the support counts computed in *Iceberg concept lattices* [15]. For a concept (X, Y) of a context $K = (O, A, I)$, we compute the support count $supp(X, Y) = |g(Y)|/|O|$ and prune redundant concepts by using some threshold. For formalizing evaluations of patterns, such values are generalized by introducing a *utility function* $u : \mathcal{P} \to \mathbb{R}^+$. A typical and well-studied utility function is, of course, the *frequency count*, or the *area function* $area(\cdot)$ which evaluates the size of a rectangle (X, Y) [6].

Based on discussions above, if we can define a utility function $u(\cdot)$ for evaluating pattern concepts, a similar discussion for pattern concepts are possible; choosing a few number of pattern concepts and constructing summary of patterns with them. Of course, there are no simple way of giving such functions. We try to introduce a simple and straightforward utility function $u_P(\cdot)$ for pattern concepts as a first step of developing pattern summarization via pattern concept lattices. In this paper, we follow the idea used in tiling databases [6], where a key criterion is given by $area(\cdot)$. We consider how to compute the value which corresponds to the area in binary databases. To take into account the wildcard $\star$ used in descriptions, we define the following simple function. For $d \in D$, we let $s(d)$ and $n(d)$ be the numbers of non wildcard and all vertices in a description d, respectively. Note that if $s(d) = n(d)$, d contains no wildcard labels. By using these functions, we compute utility values as follows:

$$u_P(A, d) = |A| \cdot \log\left(1 + s(d)\right).$$

5.1 Experiments and Discussions

We compare results of ranking pattern concepts by 1) using only $|A|$ (similar to the Iceberg concept lattices), and 2) using $u_P(\cdot)$ as a utility function. From the list of pattern concepts generated in experiments of Section 4, we rank all

Table 2. Results of ranking pattern concepts from 750 episodes in $w = 5$.

Utility	Top-5 mutually distinct descriptions of pattern concepts
$\|A\|$	$(\star, \{\star\})$, $(2, \{\star\})$, $(0, \{\star\})$, $(3, \{\star\})$, $(1, \{\star\})$
$u_P(\cdot)$	$(\star, \{0, \star\})$, $(\star, \{0, 2, 3\})$, $(\star, \{0, 1, 2\})$, $(\star, \{0, 1, 3\})$, $(\star, \{1, 2, 3\})$

pattern concepts by using a utility function, and sort the list in an ascending order, and compare two lists. We remove patterns appearing commonly in both lists to highlight differences. We give our results in Table 2.

In the result with $u_P(\cdot)$, larger descriptions appear with higher utility values compared with those by $|A|$. We can see that by modifying terms concerning $\star$, results contain more informative nodes, which are labeled by non-wildcard labels. Here we implicitly assume that descriptions contains less $\star$ would be more useful for understanding data themselves. On this viewpoint, considering two terms $s(d)$ and $n(d)$ for description d would be interesting and useful way to design utility functions for pattern concepts. We conclude that the Iceberg lattice based support counts are less effective if descriptions admit the wildcard $\star$ for pattern summarization problems.

Not only the simple computation in $u_P(A, d)$ used above, also many alternatives could be applicable for ranking. Some probabilistic methods such as the minimum description length (MDL), information-theoretic criteria would be also helpful to analyze our study more clearly. Since pattern structures have no explicit representations of binary cross tables, the difficulty lies on how to deal with a meet semi-lattice $(D, \sqcap)$. For some pattern concept (A, d) and an object $o \in O$, we say that (A, d) *subsumes* o if and only if $d \sqsubseteq \delta(o)$. This subsumption relation would be simple and helpful to evaluate concepts, but they does not adopt any complex information concerning hierarchy of events, or distances between two descriptions. In fact in the experiments, we always assume that all events except $\star$ have the same weight and $\star$ is the minimum of all events. They could be important to take into account similarity measures of events for more developments of ranking methods of pattern concepts.

5.2 Related Work

There are several studies concerning our study. It is well-known that closed itemsets correspond to maximal bipartite cliques on bipartite graphs constructed from $K = (O, A, I)$. Similarly, we sometimes deal with so called *pseudo bipartite cliques* [16], where it holds that $\mathbf{r}_1(X', Y', I) \geq 1 - \varepsilon$ with a user-specified constance ε. Obviously, pseudo bipartite cliques correspond to rectangles containing a few 0. We can regard them as some summarization or approximation of closed itemsets or concepts. Intuitively, if we use some pseudo bipartite cliques as summarization, the value $\mathbf{r}_1(X, Y, I)$ can be considered in evaluating (X, Y). Pseudo bipartite cliques can be regarded as noisy tiles, which is an extension of tiles [6].

Another typical approach for summarization is clustering patterns [18, 1]. A main problem there is how to interpret clusters or centroids, where we need to de-

sign a similarity measure and a space in which we compute the similarity. On the viewpoint of probabilistic models, there is an analysis via the maximum entropy principle [3]. However they assume that entries in a database are independently sampled, and thus we cannot apply those techniques to our setting.

6 Toward Generalizations for Bipartite Episodes

In this paper we assume that our descriptions by trees of height 1 are rich enough to apply many classes of episode patterns. We here show how to apply our pattern structure for other types of episodes, called *bipartite episodes*, as an example. An episode $G = (V, E, \lambda)$ is a *a partial bipartite episode* if 1) $V = V_1 \cup V_2$ for mutually disjoint sets V_1 and V_2, 2) for every directed edge $(x, y) \in E$, $(x, y) \in V_1 \times V_2$. If $E = V_1 \times V_2$, an episode G is called a *proper bipartite episode*. Obviously, vertices in a bipartite episode G are separated into V_1 and V_2, and we could regard them as generalizations of the source vertex and the sink vertex of diamond episodes. This indicates that the same way is applicable for bipartite episodes by defining $\sqcap$ between sets of tress. Fortunately, [9] gives the definition $\sqcap$ for sets of graphs.

$$\{t_1, \ldots, t_k\} \sqcap \{s_1, \ldots, s_m\} \equiv \mathrm{MAX}_{\sqsubseteq_T} \left(\bigcup_{i,j} (\{t_i\} \sqcap \{s_j\}) \right),$$

where $\mathrm{MAX}_{\sqsubseteq_T}(S)$ returns only maximal elements in S with respect to $\sqsubseteq_T$. Since our generalized subtree isomorphism is basically a special case of that for graphs, we can also apply this meet operation. This example suggest that if we have some background knowledge concerning a partition of V, it can be taken into account for δ and $(D, \sqcap)$ in a similar manner of diamond and bipartite episodes.

7 Conclusions and Future Work

In this paper we propose a pattern structure for diamond episodes based on an idea used in graph kernels and projections of pattern structures. Since we do not directly compute graph matching operations we conjecture that our computation could be efficient. With a slight modification of $\sqcap$, our method is also applicable for many classes of episodes, not only for diamond patterns as we mentioned above. Based on our pattern structure, we discussed summarization by using mined pattern concepts and show small examples and experimental results.

Since problems of this type are unsupervised and there is no common way of obtaining good results and of evaluating whether or not the results are good. It would be interesting to study more about this summarization problem based on concept lattices by taking into account theoretical backgrounds such as probabilistic distributions. In our future work, we try to analyze theoretical aspects on summarization via pattern structures including the wildcard $\star$ and its optimization problem to obtain compact and interesting summarization of many patterns based on our important merit of a partial order $\sqsubseteq$ between descriptions.

Acknowledgments

This work was supported by Grant-in-Aid for JSPS Fellows (26·4555) and JSPS KAKENHI Grant Number 26280085.

References

1. Al Hasan, M., Chaoji, V., Salem, S., Besson, J., Zaki, M.: Origami: Mining representative orthogonal graph patterns. In: Proc. of the 7th ICDM. pp. 153–162 (2007)
2. Buzmakov, A., Egho, E., Jay, N., Kuznetsov, S.O., Napoli, A., Raïssi, C.: The representation of sequential patterns and their projections within Formal Concept Analysis. In: Workshop Notes for LML (ECML/PKDD2013) (2013)
3. De Bie, T.: Maximum entropy models and subjective interestingness: an application to tiles in binary databases. Data Mining and Knowledge Discovery 23(3), 407–446 (2011)
4. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: Proc. of the 9th ICCS. pp. 129–142 (2001)
5. Ganter, B., Wille, R.: Formal concept analysis - mathematical foundations. Springer (1999)
6. Geerts, F., Goethals, B., Mielik ainen, T.: Tiling databases. In: Proc. of the 7th DS. pp. 278–289 (2004)
7. Katoh, T., Arimura, H., Hirata, K.: A polynomial-delay polynomial-space algorithm for extracting frequent diamond episodes from event sequences. In: Proc. of the 13th PAKDD. pp. 172–183. Springer Berlin Heidelberg (2009)
8. Kaytoue, M., Kuznetsov, S.O., Napoli, A.: Revisiting Numerical Pattern Mining with Formal Concept Analysis. In: Proc. of the 24th IJCAI (2011)
9. Kuznetsov, S.O., Samokhin, M.V.: Learning closed sets of labeled graphs for chemical applications. In: Proc. of the 15th ILP, pp. 190–208 (2005)
10. Lloyd, J.W.: Foundations of Logic Programming. Springer-Verlag New York, Inc.
11. Mannila, H., Toivonen, H., Inkeri Verkamo, A.: Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery 1(3), 259–289 (1997)
12. Merwe, D., Obiedkov, S., Kourie, D.: AddIntent: A New Incremental Algorithm for Constructing Concept Lattices. In: Proc. of the 2nd ICFCA. pp. 372–385 (2004)
13. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering frequent closed itemsets for association rules. In: Prof. of the 7th ICDT. pp. 398–416 (1999)
14. Shervashidze, N., Schweitzer, P., van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-lehman graph kernels. Journal of Machine Learning Research 12, 2539–2561 (2011)
15. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., Lakhal, L.: Computing iceberg concept lattices with titanic. Data & Knowledge Engineering 42(2), 189–222 (2002)
16. Uno, T.: An efficient algorithm for solving pseudo clique enumeration problem. Algorithmica 56(1), 3–16 (Jan 2010)
17. Vreeken, J., van Leeuwen, M., Siebes, A.: Krimp: mining itemsets that compress. Data Mining and Knowledge Discovery 23(1), 169–214 (2011)
18. Xin, D., Cheng, H., Yan, X., Han, J.: Extracting redundancy-aware top-k patterns. In: Proc. of the 12th KDD. pp. 444–453. ACM (2006)

Formal Concept Analysis for Process Enhancement Based on a Pair of Perspectives

Madori IKEDA, Keisuke OTAKI, and Akihiro YAMAMOTO

Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan
{m.ikeda, ootaki}@iip.ist.i.kyoto-u.ac.jp, akihiro@i.kyoto-u.ac.jp

Abstract. In this paper, we propose to use formal concept analysis for process enhancement, which is applied to enterprise processes, e.g., operations for patients in a hospital, repair of imperfect products in a company. Process enhancement, which is one of main goals of process mining, is to analyze a process recorded in an event log, and to improve its efficiency based on the analysis. Data formats of the logs, which contain events observed from actual processes, depend on perspectives on the observation. For example, events in logs based on a so-called process perspective are represented by their types and time-stamps, and observation based on a so-called organization perspective records events with organizations relating the occurrence of them. The logs recently became large and complex, and events are represented by many features. However, previous techniques of process mining take a single perspective into account. For process enhancement, by formal concept analysis based on a pair of features from different perspectives, we define subsequences of events whose stops are fatal to execution of a process as weak points to be removed. In our method, the extent of every concept is a set of event types, and the intent is a set of resources for events in the extent, and then, for each extent, its weakness is calculated by taking into account event frequency. We also propose some basic ideas to remove the weakest points.

Keywords: formal concept analysis, process mining, business process improvement, event log

1 Introduction

In this paper, we show a new application of formal concept analysis, *process enhancement* (or *business process improvement*), which is one of main goals of *process mining*. We show that formal concepts are useful to discover *weak points* of processes, and that a formal concept lattice works as a good guide to remove the weak points in the process enhancement.

Formal concept analysis (FCA for short) is a data analysis method which focuses on relationship between a set of objects and a set of attributes in data. A concept lattice, which is an important product of FCA, gives us valuable insights from a dual viewpoint based on the objects and the attributes. Moreover, because

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

of its simple and strong definition, various types of data can be translated for FCA, and so FCA attracts attention across various research domains.

Process mining [9,13] is a relatively young research domain, and is researched for treating enterprise processes recorded in event logs, e.g., operations for patients in a hospital, repair of imperfect products in a company. It provides a bridge between business process management (BPM for short) [12] and data mining. BPM has been investigated pragmatically, and data formats, softwares, and management systems are proposed for manipulating processes. Like recent data represented as "big data", the event logs also became huge and complicated. Thus, BPM researchers need theoretically efficient approaches for handling such big data. This is also the recent trend of data mining. Though many results produced in the last decade of process mining, there are still many challenges [11], and we work with FCA on two of them: "combining process mining with other types of analysis" and "dealing with complex event logs having diverse characteristics". We treat business process improvement which is an essential goal of process mining as a application of FCA. In order to achieve it, so many matters should be considered. At first, we have to decide features of a process which are modified for improvement, and there are various types of features to represent the process. In order to categorize the features, six central perspectives have been proposed [4,8]. For improvement in the target features, many modifications can be constructed. According to [8], there are 43 patterns of the modifications. We also have to evaluate the improvement, so an improvement measure is needed for the evaluation. Based on principal aspects of processes, *time*, *quality*, *cost*, and *flexibility*, four types of measures are considered [4,8]. In this paper, for making a process robust and reliable, we focus on two of the perspectives to detecting weak points of the process which are subsequences of events. For the detection, our method calculated a *weakness degree* regarded as one of cost measures for each subsequence which is represented by the extent of a formal concept.

This paper is organized as follows. In the next section, we introduce process mining and give a running example, and then, we show the problem tackled in this paper. In Section 3, we explain our process enhancement method. Conclusions are placed in Section 4.

2 Process Mining

In this section, we outline process mining with an example and show the problem which we try to solve.

2.1 Event Logs Observed from Actual Processes

Process mining has three types: *process discovery*, *process conformance checking*, and *process model enhancement*. Every type strongly focuses on and starts from facts observed from actual processes. It is the main difference from BPM (Business Process Management) [12] and also from WFM (Workflow Management) [6]. They are past fields of process mining and rely on prior knowledge.

The observed facts are recorded in *event logs*, and so the logs are the most important materials in process mining.

Actual event logs are usually represented in a semi-structured format like MXML [15] and XES [17]. Theoretically, every *event log* can be simply formalized as a pair (F, E) of a finite set F of *features* and a finite set E of *events*. Every feature $f \in F$ is a function from E to its *domain* D_f, and every event $e \in E$ is recorded in the form of $(f_1(e), f_2(e), ..., f_{|F|}(e))$ in $\prod_{i=1}^{|F|} D_{f_i}$. Each event corresponds with an occurrence or a task which are found by observation of an actual process. The observation is performed based on *perspectives*, and the set of features is decided by depending on them. Mathematically, a set P of the perspectives satisfies that every perspective $p \in P$ is a non-empty subset of F. Though six central perspectives which are called *process*, *object*, *organization*, *informatics*, *IT application*, or *environment* are proposed [4, 8], there are no standards for deciding P should be adopted in the observation. The set of perspectives P varies from an observation to another based on aims of process mining, kinds of processes executed by organizations, sensor systems installed to organizations, and many other factors. There are however some fundamental perspectives which are currently adopted in construction of event logs. Our approach focuses on two of these. One of them is the process perspective (it is sometimes called a *control-flow perspective*), which is focusing on how process occurs. If a process is observed based on the perspective, the set of features in its event log must include an *event type feature*, a *time stamp feature*, and *a case feature*. The case feature makes clear which case each event occurs in (note that some researches regard the case feature as a feature based on another perspective, *a case perspective*). Based on such a perspective, event logs clarify ordering of events for each case, and the set E of events can be treated as a partially ordered set $(E, \leq)$, so we sometimes use E as the poset $(E, \leq)$ in this paper. A sequence of events occurring in a case which are ordered based on time is called a *trace*. At the same time, the process can be observed based on the organization perspective, which is another fundamental perspective. The perspective focuses on where the occurrence happens or who performs the task, and event logs based on it must have a *place feature*, a *resource feature*, or an *employee feature*. In this paper, we assume that a given event log records statistically enough events.

Example 1 As a running example, we show a process which is handling a request for compensation within an airline. Customers may request the airline to compensate for various reasons, e.g., delay of flight or its cancelation. In such situations, the airline has to examine the validity of the request and needs to pay compensation if it is unquestionable. Table 1 shows an event log recording the compensation process which is partially quoted from [13]. In this example, an event means a task executed by an employee: the first event in the table shows that a task called "register request" is executed as the beginning of Case 1 by Pete at 11:02 on 30 Dec., 2010. In this log, the features Case ID, Event type, and Time are based on the process perspective. Resource feature is based on the organization perspective and represents human resources needed for each of the event. Cost feature comes from another perspective. The log also shows that

Table 1. An event log $L = (F, E)$ recording a compensation process of an airline: each row shows an event which is represented by five features.

Case ID	Event type	Resource	Cost	Time(dd-mm-yyy.hh:mm)
1	register request	Pete	50	30-12-2010.11:02
1	examine thoroughly	Sue	400	31-12-2010.10:06
1	check ticket	Mike	100	05-01-2011.15:12
1	decide	Sara	200	06-01-2011.11:18
1	reject request	Pete	200	07-01-2011.14:24
2	register request	Mike	50	30-12-2010.11:32
2	check ticket	Mike	100	30-12-2010.12:12
2	examine casually	Sean	400	30-12-2010.14:16
2	decide	Sara	200	05-01-2011.11:22
2	pay compensation	Ellen	200	08-01-2011.12:05
3	register request	Pete	50	30-12-2010.14:32
3	examine casually	Mike	400	30-12-2010.15:06
3	check ticket	Ellen	100	30-12-2010.16:34
3	decide	Sara	200	06-01-2011.09:18
3	reinitiate request	Sara	200	06-01-2011.12:18
3	examine thoroughly	Sean	400	06-01-2011.13:06
3	check ticket	Pete	100	08-01-2011.11:43
3	decide	Sara	200	09-01-2011.09:55
3	pay compensation	Ellen	200	15-01-2011.10:45

three cases are observed and recorded as three traces, and that their length are 5, 5, and 9, respectively.

2.2 Models of Processes

Models of processes are also important in process mining because they are deeply related with the three types of process mining: models are extracted from event logs by the process discovery, they are used with event logs for the process conformance checking and for the process model enhancement. Note that different types of models can be considered, and have been researched because of various aims of mining. Some models have been proposed for extract procedure of processes, e.g., Petri net [16], Business process modeling notation (BPMN) [3], Event-driven process chain (EPC) [7], and UML activity diagram [2]. These procedure models express workflow of a process clearly as directed graphs. For another aim, expressing how resources are involved in a process or how resources are related with each other, social network models are proposed [10,14]. A working-together social network expresses relations among resources which are used in the same case. A similar-task social network ignores cases but focuses on relations among resources used together for the same event. A handover-of-work social network expresses handovers from resources to resources in cases.

All of these models are developed for expression, and do not provide any analytical function. In other words, they only push event logs into their format,

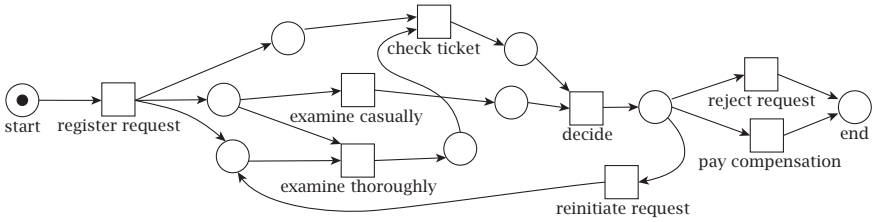


Fig. 1. A Petri net of the compensation process: every square called a transition indicates an event, and every circle called a place represents a state of the process.

and analysis is not their duty. However, for process enhancement, we need some analytical function for evaluating the enhancement. In addition, models focusing on one perspective are apt to neglect other perspectives. For example, the procedure models focusing on the process perspective do not contain information about resources which are observed based on the organization perspective. On the contrary, the social networks focusing on the organization perspective make correlations among resources explicit but make workflows which are observed based on the process perspective unclear. For our goal, detecting weak points of a process, we claim that its weakness should be measured based on at least two perspectives. This work thus relates to process model enhancement which is to extend a process model.

Example 2 Figure 1 shows a procedure model which is expressed in terms of a Petri net [16] extracted from the event log shown in Table 1. This model explicitly expresses the workflow of the compensation process and makes it clear which event happens before/after another event. On the other hand, the model ignores other perspectives: information derived from Resource and Cost features are not expressed at all in the model. Figure 2 shows a similar-task social network [10, 14] generated from the same event log. This model clarifies relations among employees sharing the same tasks, but it does not care about the ordering of events.

2.3 Weak Points Detection for Process Enhancement

Our final goal is process enhancement. For the goal, we propose to detect subsequences of events from a given event log as *weak points* which should be removed. Actually, our method does not decide whether or not subsequences of events are weak points. Instead, the method estimates the *weakness* for each of some subsequences of events and expresses it in a number called a *weakness degree*. Then, some weaker subsequence of events should be removed for the enhancement.

For the definition of the weakness degree, there are various candidates. If the process perspective is focused, sequences of events taking a lot of time in a process must be its weak points. Another type of weak points are looping sequences which

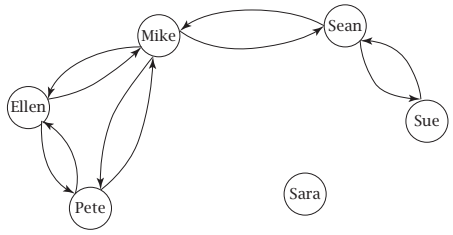


Fig. 2. A similar-task social network of the compensation process: every circle indicates an employee, and an edge is drawn between employees if their tasks are statistically similar.

many cases have to take. In the running example, it is reasonable to take costs of events into account for weakness. In this work, we focus on *importance* of a subsequence of events and *loads* of it. The importance is decided based on the process perspective and on the organization perspective. More precisely, a subsequence of events in an event log is considerable if the events are executed by a small number of resources in the log. Loads of the important sequence increase if the sequence appears many times in the log. In our method, important sequences of events having heavy loads are weak points of a process.

Example 3 In the running example, the subsequence "decide" executed by Sara should be regarded as weaker than the others. Because the subsequence is important due to the fact that it can be executed only by Sara, and because the event, "decide" by Sara, is very frequent. Only from the Petri net shown in Figure 1, it can be induced that the event "decide" is important in the process. It is also induced only from the social network shown in Figure 2 that Sara takes some important role. However, these models do not show explicitly that "decide" by Sara is important and has an impact on the process.

3 Process Enhancement via FCA

We adopt FCA for mining weak points of processes, so we firstly introduce the definitions of formal concepts and formal concept lattices with referring to [1,5]. Then, we explain our method.

3.1 From an Event Log to a Concept Lattice

A *formal context* is a triplet $K = (G, M, I)$ where G and M are mutually disjoint finite sets, and $I \subseteq G \times M$. Each element of G is called an *object*, and each element of M is called an *attribute*. For a subset of objects $A \subseteq G$ and a subset of attributes $B \subseteq M$ of a formal context K, we define $A^I = \{\, m \in M \mid \forall g \in A.\, (g, m) \in I \,\}$, $B^I = \{\, g \in G \mid \forall m \in B.\, (g, m) \in I \,\}$, and a pair (A, B) is a *formal concept* if $A^I = B$ and $A = B^I$. For a formal concept

$c = (A, B)$, A and B are called the *extent* and the *intent*, respectively, and let $\mathrm{Ex}(c) = A$ and $\mathrm{In}(c) = B$. For arbitrary formal concepts c and c', we define an order $c \leq c'$ iff $\mathrm{Ex}(c) \subseteq \mathrm{Ex}(c')$ (or equally $\mathrm{In}(c) \supseteq \mathrm{In}(c')$). The set of all formal concepts of a context $K = (G, M, I)$ with the order $\leq$ is denoted by $\underline{\mathfrak{B}}(G, M, I)$ (for short, $\underline{\mathfrak{B}}(K)$) and is called the *formal concept lattice* (concept lattice for short) of K. For every object $g \in G$ of (G, M, I), the formal concept $(\{ g \}^{II}, \{ g \}^{I})$ is called the *object concept* and denoted by γg. Similarly, for every attribute $m \in M$, the formal concept $(\{ m \}^{I}, \{ m \}^{II})$ is called the *attribute concept* and denoted by μm.

In our method, a formal context is obtained by translation from an event log, and then weak point mining is performed with a concept lattice constructed from the context. Suppose that the event log consists of two types of features, one of them is based on the process perspective, and that the other is based on the organization perspective. In this paper, the first one is called an *event-type feature* and is denoted by f_e, and the second is called a *resource feature* and is denoted by f_r. Note that the event-type feature represents types of events, not cases, and not time. This assumption is not strong because such features are very fundamental and are adopted in XES [17] in fact. From such an event log $L = (F, E)$ that $F \supseteq \{ f_e, f_r \}$, a formal context $K_L = (G, M, I)$ is translated where $G = D_{f_e}$, $M = D_{f_r}$, $I = \{ (g, m) \in G \times M \mid \exists e \in E. f_e(e) = g \wedge f_r(e) = m \}$. In the context $K_L = (G, M, I)$, $(g, m) \in I$ means that events sorted into g need a resource m. For every element $(g, m) \in I$ of the formal context K_L, we additionally define

$$\mathrm{freq}((g, m)) = |\{ e \in E \mid f_e(e) = g \wedge f_r(e) = m \}|.$$

This function outputs frequency of events which are sorted into an event-type g and need resource m in the event log L.

Example 4 In the running example, "Event type" corresponds to the event-type feature, and "Resource" corresponds to the resource feature. Therefore, a formal context $K_L = (G, M, I)$ shown in Table 2 is obtained from the event log shown in Table 1. For example, $\mathrm{freq}((\text{register request}, \text{Pete})) = 2$ shows that an event "register request" by Pete is observed twice in construction of the event log in Table 1.

From a formal context K_L translated from an event log L, a concept lattice $\underline{\mathfrak{B}}(K_L)$ is constructed for process enhancement. Each formal concept $c = (A, B)$ of the concept lattice $\underline{\mathfrak{B}}(K_L)$ represents a pair of a set A of event-types and a set B of resources needed for events in A. For every formal concept $c \in \underline{\mathfrak{B}}(K_L)$, we define

$$\mathrm{Ex}_\gamma(c) = \{ g \in \mathrm{Ex}(c) \mid \gamma g = c \}, \text{and}$$
$$\mathrm{In}_\mu(c) = \{ m \in \mathrm{In}(c) \mid \mu m = c \}.$$

By extending freq for I, we also define

$$\mathrm{freq}(c) = \sum_{g \in \mathrm{Ex}(c)} \sum_{m \in \mathrm{In}(c)} \mathrm{freq}((g, m)).$$

Table 2. A formal context $K_L = (G, M, I)$ constructed from the event log L of the compensation process: elements of G are listed in the left most column, elements of M are listed in the first row, and every cell indicates $\mathrm{freq}(i)$ for $i \in I$ unless $\mathrm{freq}(i) = 0$.

	Pete	Sue	Mike	Sara	Sean	Ellen
register request	2		1			
examine throughly		1			1	
check ticket	1		2			1
decide				4		
reject request	1					
examine casually			1		1	
pay compensation						2
reinitiate request				1		

The value $\mathrm{freq}(c)$ is the sum of frequencies of events which are sorted into an event-type $g \in \mathrm{Ex}(c)$ and need a resource $m \in \mathrm{In}(c)$.

Example 5 Figure 3 shows a concept lattice $\underline{\mathfrak{B}}(K_L)$ of the context $K_L = (G, M, I)$ shown in Table 2. For example, the left most circle in the figure indicates a formal concept $c_2 = (\{\,\mathrm{check\,ticket}, \mathrm{pay\,compensation}\,\}, \{\,\mathrm{Ellen}\,\})$. The sum of frequencies $\mathrm{freq}(c_2) = 3$ means that a task "check ticket" or "pay compensation" executed by Ellen appears three times in the event log L shown in Table 1.

3.2 Calculating Weakness Degrees

As we mentioned in Section 2.3, for every subsequence of events which is the extent of a formal concept, we define the weakness degree, and the weakness is estimated from its importance and its loads.

The importance is estimated based on both of the process perspective and the organization perspective. Every formal concept $(A, B) \in \underline{\mathfrak{B}}(K_L)$ is based on both of the perspectives because A is a set of event-types observed from the process perspective and B is a set of resources observed from the organization perspective. Such a formal concept is considered to represent that accomplishing all the events in A needs at least one of the resources in B and that every resource in B can execute all the events in A. From this consideration, we define the importance $\mathrm{imp}(c)$ of the subsequence $\mathrm{Ex}(c)$ of a formal concept $c \in \underline{\mathfrak{B}}(K_L)$ as

$$\mathrm{imp}(c) = \frac{1 + |\mathrm{Ex}_\gamma(c)|}{1 + |\mathrm{In}(c)|} \times \frac{1 + |\mathrm{Ex}(c)|}{1 + |\mathrm{In}_\mu(c)|}.$$

We call this an *importance factor*. Roughly speaking, this factor becomes large when a small number of resources are needed for a large number of events. The first term means the ratio of the number of events to the number of resources which can accomplish the events. In other words, if some or many events rely on

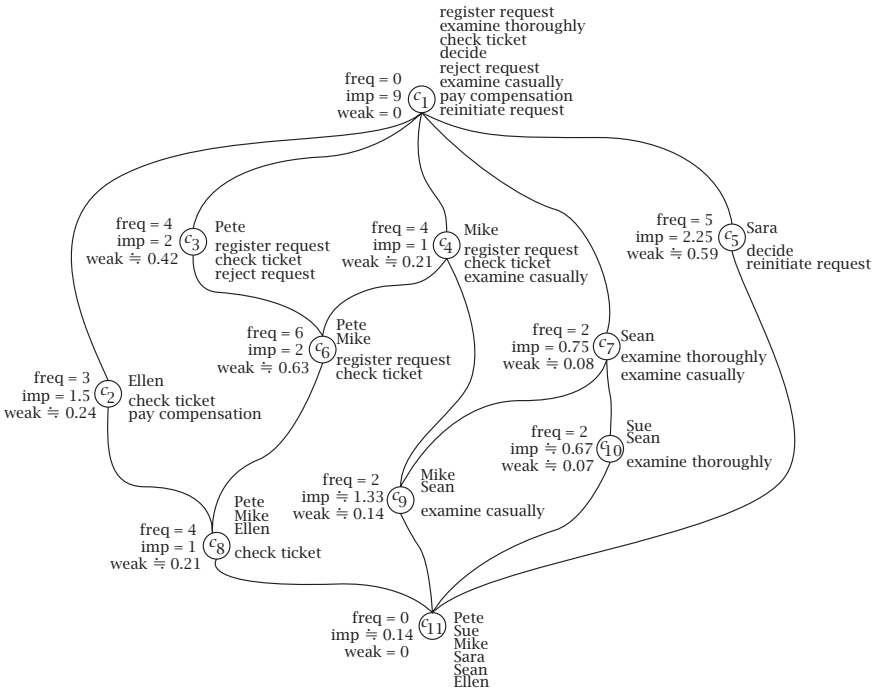


Fig. 3. A formal concept lattice $\underline{\mathfrak{B}}(K_L)$ constructed from the formal context K_L: Each circle represents a formal concept $c \in \underline{\mathfrak{B}}(K_L)$. Each edge represents an order $\leq$ between two concepts, and the greater concept is drawn above, and transitional orders are omitted. Every formal concept c accompanies with $\mathrm{Ex}(c)$ and $\mathrm{In}(c)$ on its right side and with $\mathrm{freq}(c)$, $\mathrm{imp}(c)$, and $\mathrm{weak}(c)$ on its left side.

little resources then the term is large. The second means the ratio of the number of resources to the number of events which are executed by the resources. It becomes large, if some or little resources are exhausted by many events. Also, we define $\mathrm{load}(c)$ of the subsequence $\mathrm{Ex}(c)$ as

$$\mathrm{load}(c) = \frac{\mathrm{freq}(c)}{|E|}$$

and call it a *load factor*. This is a ratio of frequency of events in the sequence $\mathrm{Ex}(c)$ to frequency of the whole events E. Then, for the subsequence $\mathrm{Ex}(c)$, the weakness degree $\mathrm{weak}(c)$ is defined as

$$\mathrm{weak}(c) = \mathrm{imp}(c) \times \mathrm{load}(c).$$

When an important sequence $\mathrm{Ex}(c)$ takes a heavy load, $\mathrm{weak}(c)$ becomes large. In other words, the weakness degree numerically shows liableness of trouble

with $\mathrm{Ex}(c)$ to cause the whole process down. By extending this definition, the weakness of the whole process can be expressed as $\sum_{c \in \underline{\mathfrak{B}}(K_L)} \mathrm{weak}(c)$.

Example 6 In Figure 3, importance factors and weakness degrees of every subsequence of events $\mathrm{Ex}(c)$, $c \in \underline{\mathfrak{B}}(K_L)$ are also drawn. The importance factors show that the sequence of tasks $\mathrm{Ex}(c_5) = \{\,\mathrm{decide}, \mathrm{reinitiate\ request}\,\}$ executed by Sara is the most important. Indeed, there is no employee who can execute the tasks "decide" and "reinitiate request", but Sara. On the other hand, the weakness degrees show that the sequence $\mathrm{Ex}(c_6) = \{\,\mathrm{register\ request}, \mathrm{check\ ticket}\,\}$ of tasks is the weakest, and that the most important sequence $\mathrm{Ex}(c_5)$ is the secondary weakest. This reversal of roles is caused by their load factors. The total weakness of the whole process $\sum_{c \in \underline{\mathfrak{B}}(K_L)} \mathrm{weak}(c)$ is around 2.59.

3.3 Removing Weak Points

A process recorded in an event log L can be enhanced by removing the weakest point or by reducing the total weakness $\sum_{c \in \underline{\mathfrak{B}}(K_L)} \mathrm{weak}(c)$. Though there are many ways for achieving the enhancement, in this paper, we achieve it by operations to an original formal context $K_L = (G, M, I)$ which remove some weakest formal concepts from its concept lattice $\underline{\mathfrak{B}}(K_L)$, or which totally reduce $\sum_{c \in \underline{\mathfrak{B}}(K_L)} \mathrm{weak}(c)$. We here show some basic ideas for such operations.

Observing the definitions about the weakness shows that there are three plans for the reduction: reducing importance factors, reducing load factors, and decreasing the number of formal concepts. Though there are many operations achieving the plans, realizable operations are restricted by considering that we try to manage an actual enterprise process. Reduction of importance factors can be achieved by increasing the number of resources to the number of events requiring the resources. Also, reducing events can decrease importance factors, but we do not adopt this way because it has a risk that the process never works. In other words, we try to enhance processes by investment in equipment not by polishing processes. Besides, reducing load factors is not reasonable for our method, because we do not have control of frequency of events. Thus, our enhancement operations are to increase resources for events requiring them or to decrease formal concepts.

For enhancement of a process recorded in an event log L, we show two kinds of such operations. The first kind is adding $(g, m) \notin I$ such that $g \in \mathrm{Ex}(c)$ and $m \in M$ to I for removing a formal concept c from $\underline{\mathfrak{B}}(K_L) \ni c$. This means to expand flexibility of resources, e.g., updating machines, and expanding applicability of materials by an innovation. We have to note that the total weakness is not always reduced in this case. The second is adding m such that $m \notin M$ and $(g, m) \notin I$ such that $g \in \mathrm{Ex}(c)$ to M and I, respectively. This can reduce the total weakness $\sum_{c \in \underline{\mathfrak{B}}(K_L)} \mathrm{weak}(c)$. This means introducing new resources for sequences of events $\mathrm{Ex}(c)$. For example, purchase of the same machines as existing ones, and using a substitute to make up a shortage of materials. In order to decide properly which kind of operations is executed, we need other factors, e.g., execution time of the process, or costs and easiness of applying the operations.

Example 7 In the running example, there are some choices for removing the weakest sequence $\mathrm{Ex}(c_6) = \{\,\text{register ticket}, \text{check ticket}\,\}$. For example, addition of (register request, Ellen) to I which means that Ellen gets an ability to "register request" can remove the weak point. It removes the concept c_6, changes c_2 into $(\{\,\text{register request}, \text{check ticket}, \text{pay compensation}\,\}, \{\,\text{Ellen}\,\})$, and c_8 into $(\{\,\text{register request}, \text{check ticket}\,\}, \{\,\text{Pete}, \text{Mike}, \text{Ellen}\,\})$, respectively. If we assume that "register request" is shared equally by Pete, Mike, and Ellen, the numbers are changed: $\mathrm{freq}(c_2) = 4$, $\mathrm{imp}(c_2) = 2$, $\mathrm{weak}(c_2) \coloneqq 0.42$, $\mathrm{freq}(c_3) = 3$, $\mathrm{imp}(c_3) = 2$, $\mathrm{weak}(c_3) \coloneqq 0.32$, $\mathrm{freq}(c_8) = 7$, $\mathrm{imp}(c_8) = 2.25$, $\mathrm{weak}(c_8) \coloneqq 0.83$. In this case, the total weakness increases to around 2.66. Employing a new person, Bob, having ability to execute "register request" is an operations of the second type. This is to add $\text{Bob} \notin M$ to M and to add $(\text{register request}, \text{Bob}) \notin I$ to I. In this case, a new concept $c_{12} = (\{\,\text{register request}\,\}, \{\,\text{Bob}\,\})$ is generated, and then, the total weakness decrease to 2.17 by assuming that "register request" is shared equally by Pete, Mike, and Bob. Because $\mathrm{weak}(c_3)$ and $\mathrm{weak}(c_6)$ decrease to around 0.32 and around 0.26, respectively, and $\mathrm{weak}(c_{12}) \coloneqq 0.05$.

4 Conclusions

In this paper, we propose to apply FCA (formal concept analysis) to process enhancement. FCA is to analyze data from a dual viewpoint which is based on objects and attributes. Processes are recorded in event logs which are constructed by observation based on some perspectives. We assign a pair of the process perspective and the organization perspective to the objects and the attributes of FCA in order to investigate weak points of a process. Weakness of a sequence of events executed by resources is calculated by importance and loads of it.

There are many problems to be solved. Our weakness of process is not defined from enough analysis because only two features from two perspectives are considered. For improving a process more efficiently, we need to take into account other features across other perspectives in weak point detection. For example, using a time-stamp feature enables us to detect bottleneck of a process, using a cost feature enables us to find costly sequences. It may be achieved by combining other process models with our concept lattice. We also have to refine the operations for removing weak points. In our method, the number of the choices for enhancement sometimes becomes so large. A plan of the refinement is to estimate in advance the total weakness of a reinforced process for each of the choices. Combining other models is also useful. For example, combining procedure models with our method can suggest some effective operations from the many choices. Because such models sufficiently treat order of events in traces which is ignored by our lattice based approach. On the other hand, there are many constraints on resources in practical processes, e.g., some materials can be substituted few materials but the others can not, and employees are divided into groups in a company. In order to reduce the choices based on such constrains, social network models might be useful.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number 26280085.

References

1. B. A. Davey, and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2002.
2. M. Dumas, and A. ter Hofstede. UML Activity Diagrams as a Workflow Specification Language. In M. Gogolla and C. Kobryn, editors, *UML 2001 The Unified Modeling Language. Modeling Languages, Concepts, and Tools*, *Lecture Notes in Computer Science*, vol. 2185, pp. 76–90, 2001.
3. R. Flowers, and C. Edeki. Business Process Modeling Notation. *International Journal of Computer Science and Mobile Computing*, vol. 2, issue 3, pp. 35–40, 2013.
4. F. Forster. The Idea behind Business Process Improvement : Toward a Business Process Improvement Pattern Framework. *BP Trends*, pp. 1–14, 2006.
5. B. Ganter, and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag New York Inc., 1997.
6. D. Georgakopoulos, M. Hornick, and A. Sheth. An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. *Distributed and Parallel Databases*, vol. 3, issue 2, pp. 119–153, 1995.
7. J. Mendling, and M. Nüttgens. EPC markup language (EPML): an XML-based interchange format for event-driven process chains (EPC). *Information Systems and e-Business Management*, vol. 4, issue 3, pp. 245–263, 2006.
8. H. A. Reijers, and S.L. Mansar. Best Practices in Business Process Redesign: An Overview and Qualitative Evaluation of Successful Redesign Heuristics. *Omega*, vol. 33, issue 4, pp. 283–306, 2005.
9. A. Shtub, and R. Karni. *ERP - The Dynamics of Supply Chain and Process Management*. Springer, 2010.
10. M. Song, and W. van der Aalst. Towards Comprehensive Support for Organizational Mining. *Decis. Support Syst.*, vol. 46, issue 1, pp. 300–317, 2008.
11. W. van der Aalst, et al.. Process Mining Manifesto. In F. Daniel, K. Barkaoui and S. Dustdar, editors, *Business Process Management Workshops*, *Lecture Notes in Business Information Processing*, vol. 99, pp. 169–194, 2012.
12. W. van der Aalst, A. ter Hofstede, and M. Weske. Business Process Management: A Survey. In W. van der Aalst and M. Weske, editors, *Business Process Management*, *Lecture Notes in Computer Science*, vol. 2678, pp. 1–12, 2003.
13. W. van der Aalst. *Process Mining - Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.
14. W. van der Aalst, H. Reijers, and M. Song. Discovering Social Networks from Event Logs. *Comput. Supported Coop. Work*, vol. 14, issue 6, pp. 549–593, 2005.
15. W. van der Aalst, B. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A. Weijters. Workflow Mining: A Survey of Issues and Approaches. *Data Knowl. Eng.*, vol. 47, issue 2, pp. 237–267, 2003.
16. B. van Dongen, A. Alves de Medeiros, and L. Wen. Process Mining: Overview and Outlook of Petri Net Discovery Algorithms. In K. Jensen and W. van der Aalst, editors, *Transactions on Petri Nets and Other Models of Concurrency II*, *Lecture Notes in Computer Science*, vol. 5460, pp. 225–242, 2009.
17. H. Verbeek, J. Buijs, B. van Dongen, and W. van der Aalst. XES, XESame, and ProM 6. In P. Soffer and E. Proper, editors, *Information Systems Evolution*, *Lecture Notes in Business Information Processing*, vol. 72, pp. 60–75, 2011.

Merging Closed Pattern Sets in Distributed Multi-Relational Data

Hirohisa Seki* and Yohei Kamiya

Dept. of Computer Science, Nagoya Inst. of Technology,
Showa-ku, Nagoya 466-8555, Japan
seki@nitech.ac.jp

Abstract. We consider the problem of mining closed patterns from multi-relational databases in a distributed environment. Given two local databases (*horizontal* partitions) and their sets of closed patterns (concepts), we generate the set of closed patterns in the global database by utilizing the *merge* (or *subposition*) operator, studied in the field of Formal Concept Analysis. Since the execution times of the merge operations increase with the increase in the number of local databases, we propose some methods for improving the merge operations. We also present some experimental results using a distributed computation environment based on the MapReduce framework, which shows the effectiveness of the proposed methods.

Key Words: multi-relational data mining, closed patterns, merge (subposition) operator, FCA, distributed databases, MapReduce

1 Introduction

Multi-relational data mining (MRDM) has been extensively studied for more than a decade (e.g., [7, 8] and references therein), and is still attracting increasing interest in the fields of data mining (e.g., [14, 29]) and inductive logic programming (ILP). In the framework of MRDM, data and patterns (or queries) are represented in the form of logical formulae such as datalog (a class of first order logic). This expressive formalism of MRDM allows us to use complex and structured data in a uniform way, including trees and graphs in particular, and multi-relational patterns in general.

On the other hand, Formal Concept Analysis (FCA) has been developed as a field of applied mathematics based on a clear mathematization of the notions of concept and conceptual hierarchy [11]. While it has attracted much interest from various application areas including, among others, data mining, knowledge acquisition and software engineering (e.g., [12]), research on extending the capabilities of FCA for AI (Artificial Intelligence) has recently been attracted much attention [20].

* This work was partially supported by JSPS Grant-in-Aid for Scientific Research (C) 24500171.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

The notion of *iceberg query lattices*, proposed by Stumme [30], combines the notions of MRDM and FCA; frequent datalog queries in MRDM correspond to iceberg concept lattices (or *frequent closed itemsets*) in FCA. Ganter and Kuznetsov [10] have extensively studied the framework of more expressive pattern structures. In MRDM, condensed representations such as closed patterns and free patterns have been also studied in *c-armr* by De Raedt and Ramon [6], and in RelLCM2 by Garriga et al. [13].

We consider in this paper the problem of mining closed patterns (or queries) in multi-relational data, particularly applying the notion of iceberg query lattices to a *distributed* mining setting. The assumption that a given dataset is distributed and stored in different sites will be reasonable for some situations where we might not be able to move local datasets into a centralized site due to too much data size and/or privacy concerns.

Given two local databases (*horizontal* partitions) and their sets of closed patterns (concepts), the set of closed patterns in the global database can be constructed by using *subposition*) operator [11, 33] or the *merge* operator [23]. From our preliminary experiments [28] using a distributed computation environment MapReduce [3], we have found that the execution times of computing the merge operations have increased with the increase in the number of local databases. In this paper, we therefore propose some methods for computing the merge operations so that we can efficiently construct the set of global closed patterns from the sets of local closed patterns. Our methods are based on the properties of the merge operator.

The organization of the rest of this paper is as follows. After summarizing some basic notations and definitions of closed patterns mining in MRDM in Sect. 2, we consider distributed closed pattern mining in MRDB and the merge operator in Sect. 3. We then explain our approach to improving the merge operations in Sect. 4. In Section 5, we show the effectiveness of our methods by some experimental results. Finally, we give a summary of this work in Section 6.

2 Iceberg Query Lattices in Multi-Relational Data Mining

2.1 Multi-Relational Data Mining

In the task of frequent pattern mining in multi-relational databases, we assume that we have a given database $\mathbf{r}$, a language of patterns, and a notion of frequency which measures how often a pattern occurs in the database. We use datalog, or Prolog without function symbols other than constants, to represent data and patterns. We assume some familiarity with the notions of logic programming (e.g., [22, 24]), although we introduce some notions and terminology in the following.

Example 1. Consider a multi-relational database $\mathbf{r}$ in Fig. 1 (above), which consists of five relations, Customer, Parent, Buys, Male and Female. For each relation, we introduce a corresponding predicate, i.e., *customer*, *parent*, *buys*, *male* and *female*, respectively.

Customer
key
allen
carol
diana
fred

Parent	
SR.	*JR.*
allen	bill
allen	jim
carol	bill
diana	eve
fred	eve
fred	hera

Buys	
key	*item*
allen	pizza
carol	pizza
diana	cake
fred	cake

Male
person
bill
jim

Female
person
eve
hera

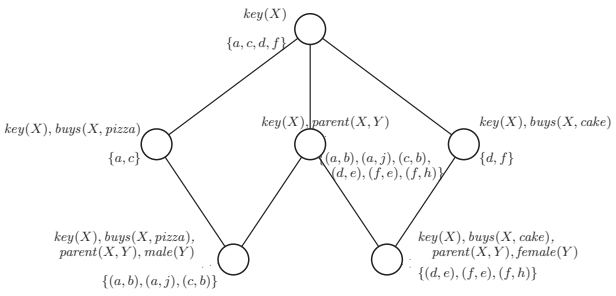


Fig. 1. An Example of Datalog Database **r** with customer relation as a key (above) and the Iceberg Query Lattice Associated to **r** (below), where a substitution $\theta = \{X/t_1, Y/t_2\}$ (resp., $\theta = \{X/t_1\}$) is simply denoted by (t_1, t_2) (resp., t_1), and the name (e.g., *allen*) of each person in the tables is abbreviated to its first character (e.g., *a*).

Consider the following pattern $P = customer(X), parent(X, Y), buys(X, pizza)$. For a substitution θ, $P\theta$ is logically entailed by **r**, denoted by $\mathbf{r} \models P\theta$, if there exists a tuple (a_1, a_2) such that $a_1 \in$ Customer, $(a_1, a_2) \in$ Parent, and tuple $(a_1, pizza) \in$ Buys. Then, $answerset(P, \mathbf{r}) = \{\{X/allen, Y/bill\}, \{X/allen, Y/jim\}, \{X/carol, Y/bill\}\}$. □

An *atom* (or *literal*) is an expression of the form $p(t_1, \ldots . t_n)$, where p is a *predicate* (or *relation*) of arity n, denoted by p/n, and each t_i is a *term*, i.e., a constant or a variable.

A substitution $\theta = \{X_1/t_1, \ldots, X_n/t_n\}$ is an assignment of terms to variables. The result of applying a substitution θ to an expression E is the expression $E\theta$, where all occurrences of variables V_i have been simultaneously replaced by the corresponding terms t_i in θ. The set of variables occurring in E is denoted by $Var(E)$.

A *pattern* is expressed as a conjunction of atoms (literals) $l_1 \wedge \cdots \wedge l_n$, denoted simply by $l_1, \ldots, l_n$. A pattern is sometimes called a *query*. We will represent conjunctions in list notation, i.e., $[l_1, \ldots, l_n]$. For a conjunction C and an atom p, we denote by $[C, p]$ the conjunction that results from adding p after the last element of C.

Let C be a pattern (i.e., a conjunction) and θ a substitution of $Var(C)$. When $C\theta$ is logically entailed by a database $\mathbf{r}$, we write it by $\mathbf{r} \models C\theta$. Let $answerset(C, \mathbf{r})$ be the set of substitutions satisfying $\mathbf{r} \models C\theta$.

In multi-relational data mining, one of the predicates is often specified as a *key* (or *target*) (e.g., [4, 6]), which determines the entities of interest and what is to be counted. The key (target) is thus to be present in all patterns considered. In Example 1, the key is predicate *customer*.

Let $\mathbf{r}$ be a database and Q be a query containing a key atom $key(X)$. Then, the *support* (or *frequency*) of Q, denoted by $supp(Q, \mathbf{r}, key)$, is defined to be the number of different keys that answer Q (called the *support count* or *absolute support*), divided by the total number of keys. Q is said to be *frequent*, if $supp(Q, \mathbf{r}, key)$ is no less than some user defined threshold min_sup.

A pattern containing a key will not be always meaningful; for example, let $C = [customer(X), parent(X, Y), buys(Z, pizza)]$ be a conjunction in Example 1. Variable Z in C is not *linked* to variable X in key atom $customer(X)$; an object represented by Z will have nothing to do with key object X. It will be inappropriate to consider such a conjunction as an intended pattern to be mined. In ILP, the following notion of *linked literals* [16] is used to specify the so-called *language bias*.

Definition 1 (Linked Literal). [16] Let $key(X)$ be a key atom and l a literal. l is said to be *linked* to $key(X)$, if either $X \in Var(l)$ or there exists a literal l_1 such that l is linked to $key(X)$ and $Var(l_1) \cap Var(l) \neq \emptyset$. □

Given a database $\mathbf{r}$ and a key atom $key(X)$, we assume that there are predefined finite sets of predicate (resp. variables; resp. constant symbols), and that, for each literal l in a conjunction C, it is constructed using the predefined sets. Moreover, each pattern C of conjunctions satisfies the following conditions: $key(X) \in C$ and, for each $l \in C, l$ is linked to $key(X)$. In the following, we denote by $\mathcal{Q}$ the set of queries (or patterns) satisfying the above bias condition.

2.2 Iceberg Query Lattices with Key

We now consider the notion of a formal context in MRDM, following [30].

Definition 2. [30] Let $\mathbf{r}$ be a datalog database and $\mathcal{Q}$ a set of datalog queries. The *formal context associated to* $\mathbf{r}$ and $\mathcal{Q}$ is defined by $\mathcal{K}_{\mathbf{r},\mathcal{Q}} = (O_{\mathbf{r},\mathcal{Q}}, A_{\mathbf{r},\mathcal{Q}}, I_{\mathbf{r},\mathcal{Q}})$, where $O_{\mathbf{r},\mathcal{Q}} = \{\theta \mid \theta$ is a grounding substitution for all $Q \in \mathcal{Q}\}$, and $A_{\mathbf{r},\mathcal{Q}} = \mathcal{Q}$, and $(\theta, Q) \in I_{\mathbf{r},\mathcal{Q}}$ if and only if $\theta \in answerset(Q, \mathbf{r})$. □

From this formal context, we can define the concept lattice the same way as in [30]. We first introduce an equivalence relation $\sim_{\mathbf{r}}$ on the set of queries: Two queries Q_1 and Q_2 are said to be *equivalent* with respect to database $\mathbf{r}$ if and only if $answerset(Q_1, \mathbf{r}) = answerset(Q_2, \mathbf{r})$. We note that $Var(Q_1) = Var(Q_2)$ when $Q_1 \sim_{\mathbf{r}} Q_2$.

Definition 3 (Closed Query). Let $\mathbf{r}$ be a datalog database and $\sim_{\mathbf{r}}$ the equivalence relation on a set of datalog queries $\mathcal{Q}$. A query (or pattern) Q is said to be *closed* (w.r.t. $\mathbf{r}$ and $\mathcal{Q}$), iff Q is the most specific query among the equivalence class to which it belongs: $\{Q_1 \in \mathcal{Q} \mid Q \sim_{\mathbf{r}} Q_1\}$. □

For any query Q_1, its *closure* is a closed query Q such that Q is the most specific query among $\{Q \in \mathcal{Q} \mid Q \sim_{\mathbf{r}} Q_1\}$. Since it uniquely exists, we denote it by $Clo(Q_1; \mathbf{r})$. We note again that $Var(Q_1) = Var(Clo(Q_1; \mathbf{r}))$ by definition. We refer to this as the *range-restricted* condition here.

Stumme [30] showed that the set of frequent closed queries forms a lattice, called *an iceberg query lattice*. In our framework, it is necessary to take our bias condition into consideration. To do that, we employ the well-known notion of the most specific generalization (or *least generalization*) [26, 24].

For queries Q_1 and Q_2, we denote by $lg(Q_1, Q_2)$ the least generalization of Q_1 and Q_2. Moreover, the *join* of Q_1 and Q_2, denoted by $Q_1 \vee Q_2$, is defined as: $Q_1 \vee Q_2 = lg(Q_1, Q_2)|_{\mathcal{Q}}$, where, for a query Q, $Q|_{\mathcal{Q}}$ is the *restriction* of Q to $\mathcal{Q}$, defined by a conjunction consisting of every literal l in Q which is linked to $key(X)$, i.e., deleting every literal in Q not linked to $key(X)$.

Definition 4. [30] Let $\mathbf{r}$ be a datalog database and $\mathcal{Q}$ a set of datalog queries. The *iceberg query lattice associated to* $\mathbf{r}$ and $\mathcal{Q}$ for *minsupp* $\in [0, 1]$ is defined as: $\mathcal{C}_{\mathbf{r},\,\mathcal{Q}} = (\{Q \in \mathcal{Q} \mid Q \text{ is closed w.r.t. } \mathbf{r} \text{ and } \mathcal{Q}, \text{ and } Q \text{ is frequent}\}, \models)$, where $\models$ is the usual logical implication. □

Example 2. Fig. 1 (below) shows the iceberg query lattice associated to $\mathbf{r}$ in Ex. 1 and $\mathcal{Q}$ with the support count 1, where each query $Q \in \mathcal{Q}$ has $customer(X)$ as a key atom, denoted by $key(X)$ for short, Q is supposed to contain at most two variables (i.e., X, Y), and the 2nd argument of predicate *buys* is a constant. □

Theorem 1. [28] Let $\mathbf{r}$ be a datalog database and $\mathcal{Q}$ a set of datalog queries where all queries contain an atom *key* and they are linked. Then, $\mathcal{C}_{\mathbf{r},\,\mathcal{Q}}$ is a $\vee$-semi-lattice. □

3 Distributed Closed Pattern Mining in MRDB

Horizontal Decomposition of MRDB and Mining Local Concepts

Our purpose in this work is to mine global concepts in a distributed setting, where a global database is supposed to be horizontally partitioned appropriately, and stored possibly in different sites. We first consider the notion of a *horizontal decomposition* of a multi-relational DB. Since a multi-relational DB consists of multiple relations, its horizontal decomposition is not immediately clear.

Definition 5. Let $\mathbf{r}$ be a multi-relational datalog database with a key predicate *key*. We call a pair $\mathbf{r}_1, \mathbf{r}_2$ a *horizontal decomposition* of $\mathbf{r}$, if (i) $\text{key}_{\mathbf{r}} = \text{key}_{\mathbf{r}_1} \cup \text{key}_{\mathbf{r}_2}$, i.e., the key relation $\text{key}_{\mathbf{r}}$ in $\mathbf{r}$ is disjointly decomposed into $\text{key}_{\mathbf{r}_1}$ and $\text{key}_{\mathbf{r}_2}$ in $\mathbf{r}_1$ and $\mathbf{r}_2$, respectively, and (ii) for any query Q, $answerset(Q, \mathbf{r}) = answerset(Q, \mathbf{r}_1) \cup answerset(Q, \mathbf{r}_2)$. □

The second condition in the above states that the relations other than the key relation in $\mathbf{r}$ are decomposed so that any answer substitution in $answerset(Q, \mathbf{r})$ is computed either in partition $\mathbf{r}_1$ or $\mathbf{r}_2$, thereby being preserved in this horizontal decomposition. An example of a horizontal decomposition of $\mathbf{r}$ is shown in Example 3 below.

Given a horizontal decomposition of a multi-relational DB, we can utilize any preferable concept (or closed pattern) mining algorithm for computing local concepts on each partition, as long as the mining algorithm is applicable to MRDM and its resulting patterns satisfy our bias condition. We use here an algorithm called *ffCLM* [27], which is based on the notion of *closure extension* due to Pasquier et al. [25] and Uno et al. [32] in frequent itemset mining.

Computing Global Closed Patterns by Merge Operator in MRDM

To compute the set of global closed patterns from the sets of local closed patterns in MRDM, we need the following *merge* operator $\oplus$. For patterns C_1 and C_2, we denote by $C_1 \cap C_2$ a possibly empty conjunction of the form: $l_1 \wedge \cdots \wedge l_k$ $(k \geq 0)$ such that, for each l_i $(i \leq k)$, $l_i \in C_1$ and $l_i \in C_2$.

Theorem 2. *[28] Let $\mathbf{r}$ be a datalog database, and $\mathbf{r}_1, \mathbf{r}_2$ a horizontal decomposition of $\mathbf{r}$. Let $\mathcal{C}$ ($\mathcal{C}_i$) ($i = 1, 2$) be the set of closed patterns of $\mathbf{r}$ ($\mathbf{r}_i$), respectively. Then, we have the following:*

$$\begin{aligned} \mathcal{C} &= \mathcal{C}_1 \oplus \mathcal{C}_2 \\ &= (\mathcal{C}_1 \cup \mathcal{C}_2) \cup \{C_1 \cap C_2 \mid C_1 \in \mathcal{C}_1, C_2 \in \mathcal{C}_2, \\ &\qquad C_1 \cap C_2 \textit{ is linked with key.}\} \end{aligned} \tag{1}$$

The set of global closed patterns $\mathcal{C}$ is obtained by the union of the local closed patterns $\mathcal{C}_1$ and $\mathcal{C}_2$, and, in addition to that, by intersecting each pattern $C_1 \in \mathcal{C}_1$ and $C_2 \in \mathcal{C}_2$. Furthermore, the pattern obtained by the intersection, $C_1 \cap C_2$, should satisfy the bias condition (Def. 1). We note that $C_1 \cap C_2$ does not necessarily satisfy the linkedness condition; for example, suppose that C_1 (C_2) is a closed pattern of the form: $C_1 = key(X), p(X, Y), m(Y)$ ($C_2 = key(X), q(X, Y), m(Y)$), respectively. Then, $C_1 \cap C_2 = key(X), m(Y)$, which is not linked to $key(X)$, and thus does not satisfy the bias condition.

We note that, in the case of transaction databases, the above theorem coincides with the one by Lucchese et al. [23].

Example 3. We consider a horizontal decomposition $\mathbf{r}_1, \mathbf{r}_2$ of $\mathbf{r}$ in Example 1 such that the key relation $\text{key}_{\mathbf{r}}$ (i.e., Customer) in $\mathbf{r}$ is decomposed into $\text{key}_{\mathbf{r}_1} = \{\text{allen}, \text{carol}\}$ and $\text{key}_{\mathbf{r}_2} = \{\text{dian}, \text{fred}\}$, and the other relations than Customer are decomposed so that they satisfy the second condition of Def. 5.

Consider a globally closed pattern $C = [key(X), parent(X, Y)]$ in Fig. 1. In $\mathbf{r}_1$, there exists a closed pattern C_1 of the form: $[C, buys(X, pizza), male(Y)]$, while, in $\mathbf{r}_2$, there exists a closed pattern C_2 of the form: $[C, buys(X, cake), female(Y)]$. Then, we have that C coincides with $C_1 \cap C_2$. □

We can now formulate our problem as follows:

Mining Globally Closed Patterns from Local DBs:
Input: A set of local databases $\{DB_1, \ldots, DB_n\}$
Output: the set of global closed patterns $\mathcal{C}_{1..n}$.

In order to compute $\mathcal{C}_{1..n}$, our approach consists of two phases: we first compute each set $\mathcal{C}_i$ $(i = 1, \ldots, n)$ of local closed patterns from DB_i, and then we compute $\mathcal{C}_{1..n}$ by applying the merge operators. We call the first phase the *mining phase*, while we call the second phase the *merge phase*.

4 Making Merge Computations Efficient in MRDM

In the merge operation in conventional data mining such as itemsets, computing the intersection of two sets in the merge operation $\oplus$ is straightforward. In MRDM, on the other hand, the computation of $\oplus$ operator becomes somewhat involved due to handling variables occurring in patterns. Namely, two additional tests are required: checking the bias condition (linkedness), and checking equivalence modulo variable renaming for eliminating duplicate patterns.

For closed patterns C_1 and C_2, we must check whether the intersection $C_1 \cap C_2$ satisfies the linkedness condition. Moreover, we must check whether $C_1 \cap C_2$ is equivalent (modulo variable renaming) to the other patterns obtained so far. For example, let C_1 (C_2) be a pattern of the form: $C_1 = key(X), p(X, Y), m(Y)$ ($C_2 = key(X), p(X, Z), m(Z)$), respectively. Then, C_1 is equivalent to C_2 modulo variable renaming.

When implementing a data mining system, such handling variables in patterns will necessarily require string manipulations, and such string operations would lead to undesirable overhead in actual implementation. In the following, we therefore propose two methods for reducing the computational costs in the merge operation.

4.1 Partitioning Pattern Sets

When computing the merge operation, we can use the following property:

Proposition 1. *Let $DB = DB_1 \cup DB_2$, and $\mathcal{C}$ ($\mathcal{C}_i$) the set of closed patterns of DB (DB_i) ($i = 1, 2$), respectively. Then,*

$$\begin{aligned} \mathcal{C} &= \mathcal{C}_1 \oplus \mathcal{C}_2 \\ &= (\mathcal{C}_1 \cup \mathcal{C}_2) \cup \{C_1 \cap C_2 | \, (C_1, C_2) \in (\mathcal{C}_1, \mathcal{C}_2) \, , \\ &\quad C_1 \cap C_2 : \textit{linked with key}, \mathit{Var}(C_1) = \mathit{Var}(C_2)\} \end{aligned} \tag{2}$$

Proof. Let C be a closed pattern in $\mathcal{C}$ such that C is linked with key. From Theorem 2, it suffices to show that there exist patterns $C_i \in \mathcal{C}_i$ $(i = 1, 2)$ such that $C = C_1 \cap C_2$ and $\mathit{Var}(C_1) = \mathit{Var}(C_2)$.

Let $C_i = \mathit{Clo}(C; DB_i)$ $(i = 1, 2)$. Then, we have from the definition of $\mathit{Clo}(\cdot; \cdot)$ that $\mathit{Var}(C) = \mathit{Var}(C_1) = \mathit{Var}(C_2)$. Moreover, we can show that $C = C_1 \cap C_2$, which is to be proved. □

From the above proposition, when computing the intersection of each pair of patterns $C_1 \in \mathcal{C}_1$ and $C_2 \in \mathcal{C}_2$ in (1), we can perform the intersection of only those pairs (C_1, C_2) containing the same set of variables, i.e., $Var(C_1) = Var(C_2)$. When compared with the original definition of the merge operator $\oplus$ (Theorem 2), the above property will be utilized to reduce the cost of the merge operations.

4.2 Merging Diff-Sets

Next, we consider another method for making the merge operation efficient, which is based on the following simple observation:

Observation 1. *Given sets of closed patterns $\mathcal{C}_1$ and $\mathcal{C}_2$, let $\mathcal{D}_1 = \mathcal{C}_1 \setminus \mathcal{C}_2$ and $\mathcal{D}_2 = \mathcal{C}_2 \setminus \mathcal{C}_1$, namely, $\mathcal{D}_i$ is a difference set (*diff-set *for short) $(i = 1, 2)$. Suppose that C is a* new *(or* generator *[33]) pattern in $\mathcal{C}_1 \oplus \mathcal{C}_2$, meaning that $C \in \mathcal{C}_1 \oplus \mathcal{C}_2$, while $C \notin \mathcal{C}_1 \cup \mathcal{C}_2$. Then, C is obtained by intersection operation, i.e., $C = C_1 \cap C_2$ for some patterns $C_1 \in \mathcal{D}_1$ and $C_2 \in \mathcal{D}_2$.*

That is, a new closed pattern C will be generated only when intersecting those patterns in the difference sets in $\mathcal{D}_1$ and $\mathcal{D}_2$. This fact easily follows from the property that the set of closed patterns is a semi-lattice: suppose otherwise that $C_1 \in \mathcal{D}_1$, while $C_2 \notin \mathcal{D}_2$. Then, $C_2 \in \mathcal{C}_1$. Since both C_1 and C_2 are in $\mathcal{C}_1$, we have that $C = C_1 \cap C_2$ is a closed pattern also in $\mathcal{C}_1$, which implies that C is not a new pattern. Algorithm 1 shows the above-mentioned method based on the difference sets. In the algorithm, the computation of supports (or occurrences) is omitted, which is done similarly in [33].

Algorithm 1: Diff-Set_Merge($\mathcal{C}_1, \mathcal{C}_2$)

input : sets of closed patterns $\mathcal{C}_1, \mathcal{C}_2$
output: $\mathcal{C}_{1..2} = \mathcal{C}_1 \oplus \mathcal{C}_2$

```
1 C = C1 ∩ C2; D1 = C1 \ C2; D2 = C2 \ C1;
2 foreach pair (C1, C2) ∈ D1 × D2 do
3     C ← C1 ∩ C2;
4     if C satisfies the bias condition and C ∉ C then
5         C ← C ∪ {C};
6     end
7 end
8 return C
```

5 Experimental Results

Implementation and Test Data

To see the effectiveness of our approach to distributed mining, we have made some experiments. As for the mining phase, we implemented our approach by

using Java 1.6.0_22. Experiments of the phase were performed on 8 PCs with Intel Core i5 processors running at 2.8GHz, 8GB of main memory, and 8MB of L2 cache, working under Ubuntu 11.04. We used Hadoop 0.20.2 using 8 PCs, and 2 mappers working on each PC. On the other hand, experiments of the merging phase were performed on one of the PCs.

We use two datasets, often used in the field of ILP; one is the mutagenesis dataset[1], and the other is an English corpus of the Penn Treebank Project[2].

The mutagenesis dataset, for example, contains 30 chemical compounds. Each compound is represented by a set of facts using predicates such as *atom*, *bond*, for example. The size of the set of predicate symbols is 12. The size of key atom ($active(X)$) is 230, and minimum support $min\_sup = 1/230$. We assume that patterns contain at most 4 variables and they contain no constant symbols. The number of the closed patterns mined is $5,784$.

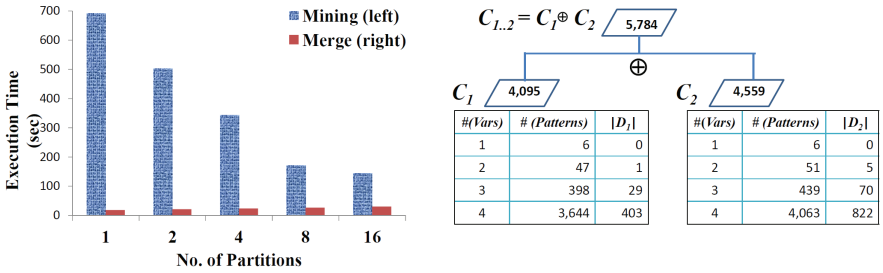
Effect of Partitioning Pattern Sets

Fig. 2 (left) summarizes the results of the execution times for a test data on the mutagenesis dataset. We can see from the figure that the execution times t_1 of the mining phase are reduced almost linearly with the number of partitions. On the other hand, the execution times t_2 of the merging phase for obtaining global closed patterns increase almost linearly with the number p of partitions from 1 (i.e., no partitioning) to 16. This is reasonable; the number of applying the merge operators is $(p-1)$ when we have p partitions. Note that the execution time for the merge phase in the case of a single partition means some start-up overheads such as opening/reading a file of the results of the mining phase, followed by preparing the inputs of the merge operation.

In this particular example, the time spent in the merge phase is relatively small when compared with that for the mining phase. This is because the number of partitions and the number of local closed patterns are rather small. When the number of partitions of a global database becomes larger, however, the execution times for the merging phase will become inevitably larger. Considering efficient merge algorithms is thus an important issue for scalability in MRDM.

To see the effect of using Proposition 1, Fig. 2 (right) shows the numbers of closed patterns in a merge computation $\mathcal{C}_1 \oplus \mathcal{C}_2$ with input sets $\mathcal{C}_1, \mathcal{C}_2$ of closed patterns for the mutagenesis dataset with 16 partitions. Each table shows the number of patterns in $\mathcal{C}_i$ $(i = 1, 2)$ containing k variables for $1 \leq k \leq 4$. The number of computing intersection operations based on Proposition 1 has been reduced to about 80% of that of the original computation. The execution times in Fig. 2 (left) are the results obtained by using this method.

[1] http://www.cs.ox.ac.uk/activities/machlearn/mutagenesis.html
[2] http://www.cis.upenn.edu/ treebank/



#(Vars)	# (Patterns)	$\lvert D_1 \rvert$
1	6	0
2	47	1
3	398	29
4	3,644	403

#(Vars)	# (Patterns)	$\lvert D_2 \rvert$
1	6	0
2	51	5
3	439	70
4	4,063	822

Fig. 2. Execution Times of the Mining Phase and the Merge Phase (left) and No. of Patterns in a Merge Computation (right): An Example in the Mutagenesis Dataset. Each number in a quadrangle is the size of a closed pattern set. $\mathcal{D}_1 = \mathcal{C}_1 \setminus \mathcal{C}_2$ and $\mathcal{D}_2 = \mathcal{C}_2 \setminus \mathcal{C}_1$.

Effect of Merging Diff-Sets

Fig. 3 shows its performance results (the execution times), compared with the naive method, using the same datasets, the mutagenesis (left) and the English corpus (right).

In both datasets, the execution times decrease as the number n of the local DBs increases; in particular, when $n = 16$ in the mutagenesis data set, the execution time is reduced to about 43% of that of the naive method. To see the reason of this results, Fig. 2 (right) shows the sizes of the difference sets $\mathcal{D}_1$ and $\mathcal{D}_2$ used in the merge computation $\mathcal{C}_1 \oplus \mathcal{C}_2$ with input sets $\mathcal{C}_1, \mathcal{C}_2$ of the closed patterns.

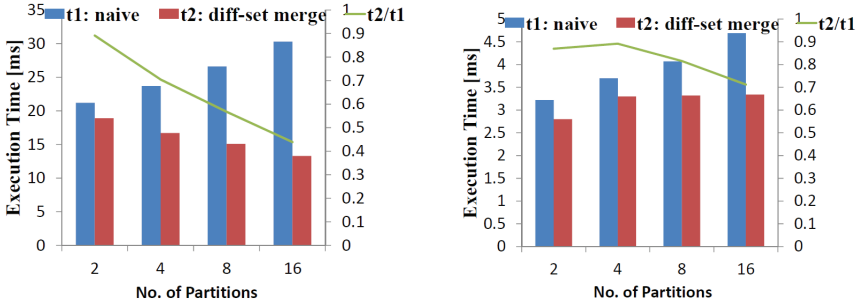


Fig. 3. Results of the Diff-Sets Merge Method: The Mutagenesis Dataset (left) and The English Corpus (right)

6 Concluding Remarks

We have considered the problem of mining closed patterns from multi-relational databases in a distributed environment. For that purpose, we have proposed two methods for making the merge (or subposition) operations efficient, and we have then exemplified the effectiveness of our method by some preliminary experimental results using MapReduce/Hadoop distributed computation framework in the mining process.

In MRDM, efficiency and scalability have been major concerns [2]. Krajca et al. [17, 18] have proposed algorithms to compute search trees for closed patterns simultaneously either in parallel or in a distributed manner. Their approaches are orthogonal to ours; it would be beneficial to employ their algorithms for computing local closed patterns in the mining phase in our framework.

In this work, we have confined ourselves to horizontal partitions of a global MRDB. It will be interesting to study *vertical* partitioning and their mixture in MRDM, where the *apposition* operator studied by Valtchev et al. [34] will play an important role. As future work, our plan is to develop an efficient algorithm dealing with such a general case in MRDM.

Acknowledgement The authors would like to thank anonymous reviewers for their useful comments on the previous version of the paper.

References

1. Agrawal, R., Srikant, R.: Fast Algorithms for Mining Association Rules. in Proc. VLDB Conf., pp. 487–499, 1994.
2. Blockeel, H., Sebag, M.: Scalability and efficiency in multi-relational data mining. SIGKDD Explorations Newsletter 2003, Vol.4, Issue 2, pp.1-14, 2003.
3. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Commun. ACM, Vol. 51, No. 1, pp.107–113, 2008.
4. Dehaspe, L.: Frequent pattern discovery in first-order logic, PhD thesis, Dept. Computer Science, Katholieke Universiteit Leuven, 1998.
5. Dehaspe, L., Toivonen, H.: Discovery of Relational Association Rules. in S. Dzeroski and N Lavrac (eds.) Relational Data Mining, pp. 189–212, Springer, 2001.
6. De Raedt, L., Ramon, J.: Condensed representations for Inductive Logic Programming. in Proc. KR'04, pp. 438-446, 2004.
7. Dzeroski, S.: Multi-Relational Data Mining: An Introduction. SIGKDD Explorations Newsletter 2003, Vol.5, Issue 1, pp.1-16, 2003.
8. Dzeroski, S., Lavrač, N. (eds.): Relational Data Mining. Springer-Verlag, Inc. 2001.
9. Ganter, B.: Two Basic Algorithms in Concept Analysis, Technical Report FB4-Preprint No. 831, TH Darmstadt, 1984. also in Formal Concept Analysis, LNCS 5986, pp. 312-340, Springer, 2010.
10. Ganter, B., Kuznetsov, S.: Pattern structures and Their Projections, ICCS-01, LNCS, 2120, pp. 129-142, 2001.
11. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, 1999.

12. Ganter, B., Stumme, G., Wille, R.: Formal Concept Analysis, Foundations and Applications. LNCS 3626, Springer, 2005.
13. Garriga,G. C., Khardon, R., De Raedt, L.: On Mining Closed Sets in Multi-Relational Data. in Proc. IJCAI 2007, pp.804-809, 2007.
14. Goethals, B., Page, W. L., Mampaey, M.: Mining Interesting Sets and Rules in Relational Databases. in Proc. 2010 ACM Sympo. on Applied Computing (SAC '10), pp. 997-1001, 2010.
15. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edition, Morgan Kaufmann Publishers Inc., 2005.
16. Helft, N.: Induction as nonmonotonic inference. in Proc. KR'89, pp. 149–156, 1989.
17. Krajca, P., Vychodil, V.: Distributed Algorithm for Computing Formal Concepts Using Map-Reduce Framework, in Proc. IDA '09, Springer-Verlag, pp. 333–344, 2009.
18. Krajca, P., Outrata, J., Vychodil, V.: Parallel algorithm for computing fixpoints of Galois connections, Annals of Mathematics and Artificial Intelligence, Vol. 59, No. 2, pp. 257–272, Kluwer Academic Publishers, 2010.
19. Kuznetsov, S. O.: A Fast Algorithm for Computing All Intersections of Objects in a Finite Semi-lattice, Automatic Documentation and Mathematical Linguistics, Vol. 27, No. 5, pp. 11-21, 1993.
20. Kuznetsov, S. O., Napoli, A., Rudolph, S., eds: FCA4AI: "What can FCA do for Artificial Intelligence?" IJCAI 2013 Workshop, Beijing, China, 2013.
21. 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.
22. Lloyd, J. W.: Foundations of Logic Programming, Springer, Second edition, 1987.
23. Lucchese, C., Orlando, S., Rergo, R.: Distributed Mining of Frequent Closed Itemsets: Some Preliminary Results. International Workshop on High Performance and Distributed Mining, 2005.
24. Nienhuys-Cheng, S-H., de Wolf, R.: Foundations of Inductive Logic Programming, LNAI 1228, Springer, 1997.
25. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering Frequent Closed Itemsets for Association Rules. in Proc. ICDT'99, LNAI 3245, pp. 398-416, 1999.
26. Plotkin, G.D.: A Note on Inductive Generalization. Machine Intelligence, Vol. 5, pp. 153-163, 1970.
27. Seki, H., Honda, Y., Nagano, S.: On Enumerating Frequent Closed Patterns with Key in Muti-relational Data. LNAI 6332, pp. 72-86, 2010.
28. Seki, H., Tanimoto, S.: Distributed Closed Pattern Mining in Multi-Relational Data based on Iceberg Query Lattices: Some Preliminary Results. in Proc. CLA'12, pp.115-126, 2012
29. Spyropoulou, E., De Bie. T., Boley, M.: Interesting Pattern Mining in Multi-Relational Data. Data Min. Knowl. Discov. 42(2), pp. 808-849, 2014.
30. Stumme, G.: Iceberg Query Lattices for Datalog. In Conceptual Structures at Work, LNCS 3127, Springer-Verlag, pp. 109-125, 2004.
31. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., Lakhal, L.: Computing Iceberg Concept Lattices with Titanic. J. on Knowledge and Data Engineering (KDE) 42(2), pp. 189-222, 2002.
32. Uno, T., Asai, T. Uchida, Y., Arimura, H.: An Efficient Algorithm for Enumerating Closed Patterns in Transaction Databases. DS'04, LNAI 3245, pp. 16-31, 2004.
33. Valtchev, P., Missaoui, R.: Building Concept (Galois) Lattices from Parts: Generalizing the Incremental Methods. In Proc. 9th Int'l. Conf. on Conceptual Structures: Broadening the Base (ICCS '01), Springer-Verlag, London, UK, pp. 290-303, 2001.
34. Valtchev, P., Missaoui, R., Pierre Lebrun, P.: A Partition-based Approach towards Constructing Galois (Concept) Lattices. Discrete Mathematics 256(3): 801-829, 2002.

Looking for bonds between nonhomogeneous formal contexts

Ondrej Krídlo, Lubomir Antoni, Stanislav Krajči

University of Pavol Jozef Šafárik, Košice, Slovakia*

Abstract. Recently, the concept lattices working with the heterogeneous structures have been fruitfully applied in a fuzzy formal concept analysis. We present a situation under nonhomogeneous formal contexts and explore the bonds in a such nonhomogeneous case. This issue requires to formulate the alternative definition of a bond and to investigate the relationships between bonds and the particular formal contexts.

Keywords: bond, heterogeneous formal context, second order formal context

1 Introduction

Formal concept analysis (FCA) [16] as an applied lattice theory allows us to explore the meaningful groupings of objects with respect to common attributes. In general, FCA is an interesting research area that provides theoretical foundations, fruitful methods, algorithms and underlying applications in many areas and has been investigated in relation to various disciplines and integrated approaches [13,15]. The feasible attempts and generalizations are investigated, one can see dual multi-adjoint concept lattices working with adjoint triples [27–29], interval-valued L-fuzzy concept lattices [1], heterogeneous concept lattices [2,3], connectional concept lattices [12, 32, 33]. Classical bonds and their generalizations acting on residuated lattices were analyzed from a broader perspective in [17,21,24].

In this paper, we deal with an alternative notion of the bonds and with a problem of looking for bonds in a nonhomogeneous formal contexts. In particular, Section 2 recalls the basic notions of a concept lattice, notion of a bond, its equivalent definition and preliminaries of a second order formal context and a heterogeneous formal context. Section 3 describes the idea of a looking for bonds in a nonhomogeneous case. Sections 4 and 5 provide the solution of this issue in terms of a second order formal context and heterogeneous formal context.

* This work was partly supported by grant VEGA 1/0832/12 and by the Slovak Research and Development Agency under contract APVV-0035-10 "Algorithms, Automata, and Discrete Data Structures".

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

2 Preliminaries

Definition 1. *Let B and A be the nonempty sets, $R \subseteq B \times A$ be an arbitrary binary relation. Triple $\langle B, A, R\rangle$ is said to be a formal context with a set of objects B and a set of their attributes A. Relationships between objects and their attributes are saved in the relation R. Let us define a pair of derivation operators $(\uparrow, \downarrow)$ as the mappings between powersets of B and A such that*

- $\uparrow\colon \mathcal{P}(B) \to \mathcal{P}(A)$ *and* $\downarrow\colon \mathcal{P}(A) \to \mathcal{P}(B)$ *where for any* $X \subseteq B$ *and* $Y \subseteq A$ *is*
- $\uparrow(X) = \{a \in A | (\forall b \in X)(b, a) \in R\}$
- $\downarrow(Y) = \{b \in B | (\forall a \in Y)(b, a) \in R\}$.

Such derivation operators can be defined as the mappings between 2-sets (borrowed from fuzzy generalization of FCA that is sometimes easier to use)

- $\uparrow\colon 2^B \to 2^A$ *and* $\downarrow\colon 2^A \to 2^B$ *where for any* $X \in 2^B$ *and* $Y \in 2^A$
- $\uparrow(X)(a) = \bigwedge_{b\in B}((b \in X) \Rightarrow ((b, a) \in R)) = \bigwedge_{b\in B}(X(b) \Rightarrow R(b, a))$
- $\downarrow(Y)(b) = \bigwedge_{a\in A}((a \in Y) \Rightarrow ((b, a) \in R)) = \bigwedge_{a\in A}(Y(a) \Rightarrow R(b, a))$.

Pair of such derivation operators forms an antitone Galois connection between complete lattices of all subsets of B and A. Hence, the compositions of the mappings form closure operators on such complete lattices.

Definition 2. *Let $\mathcal{C} = \langle B, A, R\rangle$ be a formal context. Any pair of sets $(X, Y) \in 2^B \times 2^A$ is said to be a* formal concept *iff $X = \downarrow(Y)$ and $Y = \uparrow(X)$. Object part of any concept is called* extent *and attribute part is called* intent. *Set of all extents of formal context $\mathcal{C}$ will be denoted by* $\mathrm{Ext}(\mathcal{C})$. *The notation* $\mathrm{Int}(\mathcal{C})$ *stands for the set of all intents of $\mathcal{C}$.*

All concepts ordered by set inclusion of extents (or equivalently by dual of intent inclusion) form a complete lattice structure.

2.1 Notion of bond and its equivalent definition

Definition 3. *Let $\mathcal{C}_i = \langle B_i, A_i, R_i\rangle$ for $i \in \{1, 2\}$ be two formal contexts. Relation $\beta \subseteq B_1 \times A_2$ is said to be a* bond *iff any row of the table is an intent of $\mathcal{C}_2$ and any of its column is an extent of $\mathcal{C}_1$. Set of all bonds between $\mathcal{C}_1$ and $\mathcal{C}_2$ will be denoted by* $2\text{-Bonds}(\mathcal{C}_1, \mathcal{C}_2)$.

Lemma 1. *Let $\mathcal{C}_i = \langle B_i, A_i, R_i\rangle$ for $i \in \{1, 2\}$ be two formal contexts. Then $\beta \subseteq B_1 \times A_2$ is a bond between $\mathcal{C}_1$ and $\mathcal{C}_2$ if and only if* $\mathrm{Ext}(\langle B_1, A_2, \beta\rangle) \subseteq \mathrm{Ext}(\mathcal{C}_1)$ *and* $\mathrm{Int}(\langle B_1, A_2, \beta\rangle) \subseteq \mathrm{Int}(\mathcal{C}_2)$.

Proof. $\Rightarrow$: Let $X \in \mathrm{Ext}(\langle B_1, A_2, \beta\rangle)$ be an arbitrary extent of any bond between formal contexts $\mathcal{C}_1$ and $\mathcal{C}_2$. Derivation operators of $\mathcal{C}_i$ will be denoted by $(\uparrow_i, \downarrow_i)$

for $i \in \{1,2\}$. Derivation operators of the bond will be denoted by $(\uparrow_\beta, \downarrow_\beta)$. Then there exists a set of attributes $Y \subseteq A_2$ such that

$$\begin{aligned}
\downarrow_\beta (Y)(b_1) &= \bigwedge_{a_2 \in A_2} (Y(a_2) \Rightarrow \beta(b_1, a_2)) \\
&\quad \beta(-, a_2) \text{ is an extent of } \mathrm{Ext}(\mathcal{C}_1) \text{ hence there exists } Z \subseteq A_1 \\
&= \bigwedge_{a_2 \in A_2} (Y(a_2) \Rightarrow \downarrow_1 (Z)(b_1)) \\
&= \bigwedge_{a_2 \in A_2} \Big(Y(a_2) \Rightarrow \bigwedge_{a_1 \in A_1} \big(Z(a_1) \Rightarrow R_1(b_1, a_1) \big) \Big) \\
&= \bigwedge_{a_2 \in A_2} \bigwedge_{a_1 \in A_1} (Y(a_2) \Rightarrow (Z(a_1) \Rightarrow R_1(b_1, a_1))) \\
&= \bigwedge_{a_2 \in A_2} \bigwedge_{a_1 \in A_1} ((Y(a_2) \wedge Z(a_1)) \Rightarrow R_1(b_1, a_1)) \\
&= \bigwedge_{a_1 \in A_1} \Big(\bigvee_{a_2 \in A_2} \big(Y(a_2) \wedge Z(a_1) \big) \Rightarrow R_1(b_1, a_1) \Big) \\
&= \bigwedge_{a_1 \in A_1} (Z_Y(a_1) \Rightarrow R_1(b_1, a_1)) \\
&= \downarrow_1 (Z_Y)(b_1) \text{ where } Z_Y(a_1) = \textstyle\bigvee_{a_2 \in A_2} (Y(a_2) \wedge Z(a_1))
\end{aligned}$$

Hence, $\mathrm{Ext}(\langle B_1, A_2, \beta \rangle) \subseteq \mathrm{Ext}(\mathcal{C}_1)$. Similarly for intents.

$\Leftarrow$: Assume a formal context $\langle B_1, A_2, \beta \rangle$ such that it holds $\mathrm{Ext}(\langle B_1, A_2, \beta \rangle) \subseteq \mathrm{Ext}(\mathcal{C}_1)$ and $\mathrm{Int}(\langle B_1, A_2, \beta \rangle) \subseteq \mathrm{Int}(\mathcal{C}_2)$. From the simple fact that any row of any context is its intent and any column is its extent and from the previous inclusions, we obtain that β is a bond between $\mathcal{C}_1$ and $\mathcal{C}_2$. □

Hence, the notion of bond can be defined equivalently as follows.

Definition 4. *Let $\mathcal{C}_i = \langle B_i, A_i, R_i \rangle$ for $i \in \{1,2\}$ be two formal contexts. Formal context $\mathcal{B} = \langle B_1, A_2, \beta \rangle$ is said to be a* bond *between $\mathcal{C}_1$ and $\mathcal{C}_2$ if $\mathrm{Ext}(\mathcal{B}) \subseteq \mathrm{Ext}(\mathcal{C}_1)$ and $\mathrm{Int}(\mathcal{B}) \subseteq \mathrm{Int}(\mathcal{C}_2)$.*

More about the equivalent definition of bond could be found in [17–19].

2.2 Direct product of two formal contexts and bonds

Let us recall the definition and important property of direct product of two formal contexts. More details about such topic can be found in [21, 26].

Definition 5. *Let $\mathcal{C}_i = \langle B_i, A_i, R_i \rangle$ be two formal contexts. Formal context $\mathcal{C}_1 \Delta \mathcal{C}_2 = \langle B_1 \times A_2, B_2 \times A_1, R_1 \Delta R_2 \rangle$ where*

$$\begin{aligned}
(R_1 \Delta R_2)((b_1, a_2), (b_2, a_1)) &= R_1(b_1, a_1) \vee R_2(b_2, a_2) \\
&= \neg R_1(b_1, a_1) \Rightarrow R_2(b_2, a_2) \\
&= \neg R_2(b_2, a_2) \Rightarrow R_1(b_1, a_1)
\end{aligned}$$

for any $(b_i, a_i) \in B_i \times A_i$ *for all* $i \in \{1,2\}$ *is said to be a* direct product *of formal contexts* $\mathcal{C}_1$ *and* $\mathcal{C}_2$.

Lemma 2. *Let* $\mathcal{C}_i = \langle B_i, A_i, R_i \rangle$ *be two formal contexts. Every extent of* $\mathcal{C}_1 \Delta \mathcal{C}_2$ *is a bond between* $\mathcal{C}_1$ *and* $\mathcal{C}_2$.

2.3 Second order formal contexts

In this subsection, we remind a notion of a second order formal concept [24].

Definition 6. *Consider two non-empty index sets* I *and* J *and a formal context* $\langle \bigcup_{i\in I} B_i, \bigcup_{j\in J} A_j, r \rangle$, *whereby*

- $B_{i_1} \cap B_{i_2} = \emptyset$ *for any* $i_1, i_2 \in I$, $i_1 \neq i_2$,
- $A_{j_1} \cap A_{j_2} = \emptyset$ *for any* $j_1, j_2 \in J$, $j_1 \neq j_2$,
- $r : \bigcup_{i\in I} B_i \times \bigcup_{j\in J} A_j \to 2$.

Moreover, consider two non-empty sets of 2*-contexts notated*

- $\{\mathcal{C}_i = \langle B_i, T_i, p_i \rangle : i \in I\}$
- $\{\mathcal{D}_j = \langle O_j, A_j, q_j \rangle : j \in J\}$.

Formal context of second order is a tuple

$$\Big\langle \bigcup_{i\in I} B_i,\ \{\mathcal{C}_i; i \in I\},\ \bigcup_{j\in J} A_j,\ \{\mathcal{D}_j; j \in J\},\ \bigcup_{(i,j)\in I\times J} r_{i,j} \Big\rangle,$$

where $r_{i,j} : B_i \times A_j \to 2$ *defined as* $r_{i,j}(b,a) = r(b,a)$ *for any* $b \in B_i$ *and* $a \in A_j$.

In what follows, consider the below described notation. Let us have an L-set $f : X \to 2$ for a non-empty universe set $X = \bigcup_{i\in I} X_i$, where $X_{i_1} \cap X_{i_2} = \emptyset$ for any $i_1, i_2 \in I$. Then $f^i : X_i \to 2$ is defined as $f^i(x) = f(x)$ for an arbitrary $x \in X_i$ and $i \in I$.

We define the mappings between direct products of two sets of concept lattices (that correspond to the two sets of 2-contexts given above) in the following form:

Definition 7. *Let us define the mappings* $\langle \Uparrow, \Downarrow \rangle$ *as follows*

$$\Uparrow\colon \prod_{i\in I} \mathrm{Ext}(\mathcal{C}_i) \to \prod_{j\in J} \mathrm{Int}(\mathcal{D}_j) \text{ and } \Downarrow\colon \prod_{j\in J} \mathrm{Int}(\mathcal{D}_j) \to \prod_{i\in I} \mathrm{Ext}(\mathcal{C}_i)$$

$$\Uparrow (\Phi)^j = \bigwedge_{i\in I} \uparrow_{ij} (\Phi^i), \text{ for any } \Phi \in \prod_{i\in I} \mathrm{Ext}(\mathcal{C}_i)$$

$$\Downarrow (\Psi)^i = \bigwedge_{j\in J} \downarrow_{ij} (\Psi^j), \text{ for any } \Psi \in \prod_{j\in J} \mathrm{Int}(\mathcal{D}_j)$$

such that $(\uparrow_{ij}, \downarrow_{ij})$ *is a pair of derivation operators defined on* $\langle B_i, A_j, \rho_{ij} \rangle$ *where*

$$\rho_{ij} = \bigwedge \{\beta \in 2\text{-Bonds}(\mathcal{C}_i, \mathcal{D}_j) : (\forall (b_i, a_j) \in B_i \times A_j) \beta(b_i, a_j) \geq r_{ij}(b_i, a_j)\}.$$

2.4 Heterogeneous formal contexts

A heterogeneous extension in FCA based on the totally diversification of objects, attributes and table fields has been introduced in [3]. In the following, we remind the definition of a heterogeneous formal context and its derivation operators.

Definition 8. *Heterogeneous formal context is a tuple* $\mathcal{C} = \langle B, A, \mathcal{P}, R, \mathcal{U}, \mathcal{V}, \odot \rangle$, *where*

- B *and* A *are non-empty sets,*
- $\mathcal{P} = \{\langle P_{b,a}, \leq_{P_{b,a}} \rangle : (b,a) \in B \times A\}$ *is a system of posets,*
- R *is a mapping from* $B \times A$ *such that* $R(b,a) \in P_{b,a}$ *for any* $b \in B$ *and* $a \in A$,
- $\mathcal{U} = \{\langle U_b, \leq_{U_b} \rangle : b \in B\}$ *and* $\mathcal{V} = \{\langle V_a, \leq_{V_a} \rangle : a \in A\}$ *are systems of complete latices,*
- $\odot = \{\circ_{b,a} : (b,a) \in B \times A\}$ *is a system of isotone and left-continuous mappings* $\circ_{b,a} : U_b \times V_a \longrightarrow P_{b,a}$.

Let us define the derivation operators of a heterogeneous formal context as a pair of mappings $(\nearrow, \swarrow)$, *whereby* $\nearrow$: $\prod_{b \in B} U_b \to \prod_{a \in A} V_a$ *and* $\swarrow$: $\prod_{a \in A} V_a \to \prod_{b \in B} U_b$ *such that*

- $\swarrow (f)(a) = \bigvee \{v \in V_a | f(b) \circ_{b,a} v \leq R(b,a)\}$ *for any* $f \in \prod_{b \in B} U_b$
- $\nearrow (g)(b) = \bigvee \{u \in U_b | u \circ_{b,a} g(a) \leq R(b,a)\}$ *for any* $g \in \prod_{a \in A} V_a$.

3 Problem description and sketch of solution

In this section we discussed why we have proposed an equivalent definition of bond. First, consider the classical definition of bond. It is a binary relation (table) between objects and attributes from different contexts such that its rows are intents and columns are extents of different input contexts. The issue of looking for bonds in a classical or homogeneous fuzzy case can be solved successfully [17, 21].

The solution of this issue requires the alternative definition of a bond. Hence, new definition of a bond focuses not only on a relation with some special properties, but also on a bond as a formal context, whereby its concept lattice is connected to concept lattices of input contexts in some sense. As a consequence, a generalization for heterogeneous bonds is possible. One can find the methods in effort to equivalently modify the input heterogeneous formal contexts and to extract bonds as the extents of a direct product.

The proposed modification runs as follows. Each individual pair that includes a "conjunction" $\circ_{b,a}$ and a value of the poset $P_{b,a}$ is replaced by a bond from 2-Bonds$(\langle U_b, U_b, \leq \rangle, \langle V_a, V_a, \geq \rangle)$. This completely covers the Galois connection between the complete lattices of any object–attribute pair from $B \times A$.

At the beginning, we will show how this modification looks in terms of second order formal contexts. Then we define new modified heterogeneous formal context such that its concept lattice is identical to the original.

4 Second order form of scaled heterogeneous formal context

In effort to formalize the second order form of scaled heterogeneous formal context and its derivation operators, the definition of the following mappings is required:

Definition 9. *Let $(L, \leq)$ be a complete lattice. Let us define mappings $\overline{(-)}^L$ and $\underline{(-)}_L$ where*

- $\overline{(-)}^L : L \to 2^L$ *such that* $\overline{k}^L(m) = (m \leq k)$ *for any* $k, m \in L$
- $\underline{(-)}_L : 2^L \to L$ *such that* $\underline{X}_L = \bigvee X$ *for any* $X \subseteq L$.

Let us have an arbitrary $f \in \prod_{b\in B} U_b$. Let us denote $\overline{f}$ as a subset of $\bigcup_{b\in B} U_b$ defined as $\overline{f} = \bigcup_{b\in B}\{u \in U_b | u \leq f(b)\}$. Similarly for any $g \in \prod_{a\in A} V_a$.

More information about Cartesian representation of fuzzy sets could be found in [10].

Now, consider a heterogeneous formal context $\mathcal{C} = \langle B, A, \mathcal{P}, R, \mathcal{U}, \mathcal{V}, \odot\rangle$. A second order form of scaled heterogeneous formal context is defined as

$$\overline{\mathcal{C}} = \left\langle \bigcup_{b\in B} U_b, \{\langle U_b, U_b, \leq\rangle | b \in B\}, \bigcup_{a\in A} V_a, \{\langle V_a, V_a, \geq\rangle | a \in A\}, \overline{R} \right\rangle,$$

whereby all external contexts are classical crisp contexts and $\overline{R}$ is a classical crisp binary relation defined as $\overline{R}(u, v) = ((u \circ_{b,a} v) \leq R(b, a))$ for any $(u, v) \in U_b \times V_a$ and any $(b, a) \in B \times A$.

In the following, we define the derivation operators of such special second order formal context. First, we state some appropriate remarks and facts. Note that a relation $\overline{R}$ constrained to $U_b \times V_a$ for any pair $(b, a) \in B \times A$ is monotone in both arguments due to its definition. Similarly, consider the fact that any extent of $\langle U_b, U_b, \leq\rangle$ and any intent of $\langle V_a, V_a, \geq\rangle$ is a principal down-set of a corresponding complete lattice (i.e. there exists an element in this complete lattice such that all lower or equal elements are in the extent or in the intent). Hence, a relation $\overline{R}$ constrained to $U_b \times V_a$ for some $(b, a) \in B \times A$ is a 2-bond between $\langle U_b, U_b, \leq\rangle$ and $\langle V_a, V_a, \geq\rangle$ which will be denoted by $\rho_{b,a}$. Note that any $\Phi \in \prod_{b\in B} \mathrm{Ext}(\langle U_b, U_b, \leq\rangle)$ has the form $\overline{f}$ for some $f \in \prod_{b\in B} U_b$. Consider an arbitrary $f \in \prod_{b\in B} U_b$ and $g \in \prod_{a\in A} V_a$. Hence, the derivation operators are defined as follows:

- $\overline{\nearrow}(\overline{f})(v) = \bigwedge_{b\in B} \uparrow_{b,a} (\overline{f(b)}^b)(v)$ for any $v \in V_a$ and $a \in A$
- $\overline{\swarrow}(\overline{g})(u) = \bigwedge_{a\in A} \downarrow_{b,a} (\overline{g(a)}^a)(u)$ for any $u \in U_b$ and $b \in B$.

In a previous definition, the pair of mappings $(\uparrow_{b,a}, \downarrow_{b,a})$ are derivation operators of a formal context $\langle U_b, V_a, \rho_{b,a}\rangle$ for any $(b, a) \in B \times A$. For the sake of brevity, we use the shortened notation $\overline{(-)}^b$ instead of $\overline{(-)}^{U_b}$ and similarly $\overline{(-)}^a$ instead of $\overline{(-)}^{V_a}$.

Lemma 3. *The concept lattices of $\mathcal{C}$ and $\overline{\mathcal{C}}$ are isomorphic.*

Proof. Consider an arbitrary $f \in \prod_{b\in B} U_b$. We will show that $\overline{\nearrow}(\overline{f}) = \overline{\nearrow(f)}$.

Firstly consider the fact of left-continuity of both arguments of $\circ_{b,a}$ for any $(b,a) \in B\times A$. Due to this property, one can define two residuums in the following way. Let $(b,a) \in B \times A$ be an arbitrary object-attribute pair and consider the arbitrary values $u \in U_b$, $v \in V_a$ and $p \in P_{b,a}$. Then define

- $\to_{b,a}\colon U_b \times P_{b,a} \to V_a$, such that $u \to_{b,a} p = \bigvee\{v \in V_a | u \circ_{b,a} v \le p\}$
- $\to_{a,b}\colon V_a \times P_{b,a} \to U_b$, such that $v \to_{a,b} p = \bigvee\{u \in U_b | u \circ_{b,a} v \le p\}$.

$$
\begin{aligned}
\overline{\nearrow}\left(\overline{f}\right)(v) &= \bigwedge_{b\in B} \uparrow_{b,a} \left(\overline{f(b)}^{b}\right)(v) \\
&= \bigwedge_{b\in B} \bigwedge_{u\in U_b} \left(\overline{f(b)}^{b}(u) \Rightarrow \rho_{b,a}(u,v)\right) \\
&= \bigwedge_{b\in B} \bigwedge_{u\in U_b} ((u \le f(b)) \Rightarrow (u \circ_{b,a} v \le R(b,a))) \\
&= \bigwedge_{b\in B} \left(\bigwedge_{u\in U_b; u\not\le f(b)} 1 \wedge \bigwedge_{u\in U_b; u\le f(b)} ((u \le f(b)) \Rightarrow (u \circ_{b,a} v \le R(b,a))) \right) \\
&= \bigwedge_{b\in B} \bigwedge_{u\in U_b; u\le f(b)} (u \circ_{b,a} v \le R(b,a)) \\
&= \bigwedge_{b\in B} (f(b) \circ_{b,a} v \le R(b,a)) \\
&= \bigwedge_{b\in B} (v \le f(b) \to_{b,a} R(b,a)) \\
&= \left(v \le \bigwedge_{b\in B} (f(b) \to_{b,a} R(b,a)) \right) \\
&= \left(v \le \bigwedge_{b\in B} \bigvee\{w \in V_a | (f(b) \circ_{b,a} w \le R(b,a))\} \right) \\
&= \left(v \le \bigvee\{w \in V_a | (\forall b \in B)(f(b) \circ_{b,a} w \le R(b,a))\} \right) \\
&= (v \le \overline{\nearrow}(f)(a)) = \overline{\nearrow(f)(a)}^{a}(v).
\end{aligned}
$$

Analogously one can obtain $\overline{\swarrow}(\overline{g})(u) = \overline{\swarrow(g)(b)}^{b}(u)$. □

4.1 Back to heterogeneous formal contexts

Now, we look at heterogeneous formal context introduced in Subsection 2.3. A second order formal context $\overline{\mathcal{C}}$ can be seen as a special heterogeneous formal context $\widehat{\mathcal{C}}$, whereby the family of posets $\{\langle P_{b,a}, \le\rangle | (b,a) \in B \times A\}$ is replaced by

a set of 2-bonds $\{\rho_{b,a} \in \text{2-Bonds}(\langle U_b, U_b, \leq\rangle, \langle V_a, V_a, \leq\rangle) | (b,a) \in B \times A\}$. Hence, the final form of such heterogeneous formal context is

$$\widehat{\mathcal{C}} = \left\langle B, A, \rho, \widehat{R}, \mathcal{U}, \mathcal{V}, \{\overline{\times}_{b,a} | (b,a) \in B \times A\} \right\rangle$$

where

- $\rho = \{\rho_{b,a} \in \text{2-Bonds}(\langle U_b, U_b, \leq\rangle, \langle V_a, V_a, \leq\rangle) | (b,a) \in B \times A\}$
- $\rho_{b,a}(u,v) = (u \circ_{b,a} v \leq R(b,a))$
- $\widehat{R}(b,a) = \rho_{b,a} \in \text{2-Bonds}(\langle U_b, U_b, \leq\rangle, \langle V_a, V_a, \leq\rangle)$ for any $(b,a) \in B \times A$
- $\overline{\times}_{b,a} : U_b \times V_a \to 2^{U_b \times V_a}$ defined as a Cartesian product $u \overline{\times} v = \overline{u} \times \overline{v}$.

The derivation operators of $\widehat{\mathcal{C}}$ are defined as

- $\uparrow (f)(a) = \bigvee \{v \in V_a | (\forall b \in B) f(b) \overline{\times}_{b,a} v \subseteq \rho_{b,a}\}$ for any $f \in \prod_{b \in B} U_b$
- $\downarrow (g)(b) = \bigvee \{u \in U_b | (\forall a \in A) u \overline{\times}_{b,a} g(a) \subseteq \rho_{b,a}\}$ for any $g \in \prod_{a \in A} V_a$.

Lemma 4. *The concept lattices of $\mathcal{C}$ and $\widehat{\mathcal{C}}$ are identical.*

Proof. Firstly consider that for any $(u,v) \in U_b \times V_a$ for any $(b,a) \in B \times A$ the following holds:

$$\begin{aligned} u \overline{\times} v \subseteq \rho_{b,a} &= \overline{u} \times \overline{v} \subseteq \rho_{b,a} \\ &= \rho_{b,a}(u,v) \\ &= (u \circ_{b,a} v \leq R(b,a)). \end{aligned}$$

Let $f \in \prod_{b \in B} U_b$ be arbitrary. Then

$$\begin{aligned} \uparrow (f)(a) &= \bigvee \{v \in V_a | (\forall b \in B) f(b) \overline{\times}_{b,a} v \subseteq \rho_{b,a}\} \\ &= \bigvee \{v \in V_a | (\forall b \in B) f(b) \circ_{b,a} v \leq R(b,a)\} \\ &= \nearrow (f)(a). \end{aligned}$$

Analogously for $\downarrow (g)(b) = \swarrow (g)(b)$ for any $g \in \prod_{a \in A} V_a$. □

5 Bonds between heterogeneous formal contexts

We present a definition of a bond between two heterogeneous formal contexts which can be formulated as follows.

Definition 10. *Let $\mathcal{C}_i = \langle B_i, A_i, \mathcal{P}_i, R_i, \mathcal{U}_i, \mathcal{V}_i, \odot_i \rangle$ for $i \in \{1,2\}$ be two heterogeneous formal contexts. The heterogeneous formal context $\mathcal{B} = \langle B_1, A_2, \mathcal{P}, R, \mathcal{U}_1, \mathcal{V}_2, \odot \rangle$ such that* $\text{Ext}(\mathcal{B}) \subseteq \text{Ext}(\mathcal{C}_1)$ *and* $\text{Int}(\mathcal{B}) \subseteq \text{Int}(\mathcal{C}_2)$ *is said to be a* bond *between two heterogeneous formal contexts $\mathcal{C}_1$ and $\mathcal{C}_2$.*

5.1 Direct product of two heterogeneous formal contexts

In this subsection, we define a direct product of two heterogeneous formal contexts. Further, we give an answer on how to find a bond between two heterogeneous formal contexts.

Definition 11. *Let* $\mathcal{C}_i = \langle B_i, A_i, \mathcal{P}_i, R_i, \mathcal{U}_i, \mathcal{V}_i, \odot_i \rangle$ *for* $i \in \{1,2\}$ *be two heterogeneous formal contexts. The heterogeneous formal context*

$$\mathcal{C}_1 \Delta \mathcal{C}_2 = \langle B_1 \times A_2, B_2 \times A_1, \mathcal{P}_\Delta, R_\Delta, \mathcal{U}_\Delta, \mathcal{V}_\Delta, \times \rangle$$

such that

- $\mathcal{P}_\Delta = \{\rho_{b_1,a_1} \Delta \rho_{b_2,a_2} | ((b_1,a_2),(b_2,a_1)) \in (B_1 \times A_2) \times (B_2 \times A_1)\}$
- *where* $\rho_{b_i,a_i}(u,v) = (u \circ_{b_i,a_i} v \leq R_i(b_i,a_i))$ *for any* $(u,v) \in U_{b_i} \times V_{a_i}$ *for any* $(b_i,a_i) \in B_i \times A_i$ *for any* $i \in \{1,2\}$
- $R_\Delta((b_1,a_2),(b_2,a_1)) = \rho_{b_1,a_1} \Delta \rho_{b_2,a_2}$ *for any* $b_i \in B_i$ *and* $a_i \in A_i$ *for all* $i \in \{1,2\}$
- $\mathcal{U}_\Delta = \{\gamma_{1,2} \in 2\text{-Bonds}(\langle U_{b_1}, U_{b_1}, \leq \rangle, \langle V_{a_2}, V_{a_2}, \geq \rangle) | (b_1,a_2) \in B_1 \times A_2\}$
- $\mathcal{V}_\Delta = \{\gamma_{2,1} \in 2\text{-Bonds}(\langle U_{b_2}, U_{b_2}, \leq \rangle, \langle V_{a_1}, V_{a_1}, \geq \rangle) | (b_2,a_1) \in B_2 \times A_1\}$

is said to be a direct product of two heterogeneous formal contexts.

Lemma 5. *Let* $\mathcal{C}_i = \langle B_i, A_i, \mathcal{P}_i, R_i, \mathcal{U}_i, \mathcal{V}_i, \odot_i \rangle$ *for* $i \in \{1,2\}$ *be two heterogeneous formal contexts. Let*

$$R \in \prod_{(b_1,a_2) \in B_1 \times A_2} 2\text{-Bonds}(\langle U_{b_1}, U_{b_1}, \leq \rangle, \langle V_{a_2}, V_{a_2}, \geq \rangle)$$

be an extent of the direct product $\mathcal{C}_1 \Delta \mathcal{C}_2$*. Then a heterogeneous formal context* $\mathcal{B} = \langle B_1, A_2, \rho, R, \mathcal{U}_1, \mathcal{V}_2, \overline{\times} \rangle$ *where*

$$\rho = \{2\text{-Bonds}(\langle U_{b_1}, U_{b_1}, \leq \rangle, \langle V_{a_2}, V_{a_2}, \geq \rangle) | (b_1,a_2) \in B_1 \times A_2\}$$

is a bond between $\mathcal{C}_1$ *and* $\mathcal{C}_2$*.*

Proof. Let us have any intent of $\mathcal{B}$. Then there exists $f \in \prod_{b_1 \in B_1} U_{b_1}$ such that

$$\overline{\nearrow_{\mathcal{B}}(f)(a_2)}^{a_2}(v_2) = \overline{\nearrow_{\overline{\mathcal{B}}}}(\overline{f})(v_2)$$

$$= \bigwedge_{b_1 \in B_1} \uparrow_{R(b_1,a_2)} (\overline{f(b_1)}^{b_1})(v_2)$$

$$= \bigwedge_{b_1 \in B_1} \bigwedge_{u_1 \in U_{b_1}} (\overline{f(b_1)}^{b_1}(u_1) \Rightarrow R(b_1,a_2)(u_1,v_2))$$

$$R = \swarrow_\Delta (Q) \text{ for some } Q \in \prod_{(b_2,a_1) \in B_2 \times A_1} 2\text{-Bonds}(\langle U_{b_2}, U_{b_2}, \leq \rangle, \langle V_{a_1}, V_{a_1}, \geq \rangle)$$

$$= \bigwedge_{b_1 \in B_1} \bigwedge_{u_1 \in U_{b_1}} (\overline{f(b_1)}^{b_1}(u_1) \Rightarrow \swarrow_\Delta (Q)(b_1,a_2)(u_1,v_2))$$

$$= \bigwedge_{b_1 \in B_1} \bigwedge_{u_1 \in U_{b_1}} \Big(\overline{f(b_1)}^{b_1}(u_1) \Rightarrow \bigwedge_{(b_2,a_1) \in B_2 \times A_1} \downarrow_{\rho_{b_1,a_1} \Delta \rho_{b_2,a_2}} (Q(b_2,a_1))(u_1,v_2) \Big)$$

$$= \bigwedge_{b_1 \in B_1} \bigwedge_{u_1 \in U_{b_1}} \Big(\overline{f(b_1)}^{b_1}(u_1) \Rightarrow$$

$$\bigwedge_{b_2 \in B_2} \bigwedge_{a_1 \in A_1} \bigwedge_{(u_2,v_1) \in U_{b_2} \times V_{a_1}} \big(Q(b_2,a_1)(u_2,v_1) \Rightarrow (\rho_{b_1,a_1} \Delta \rho_{b_2,a_2})((u_1,v_2),(u_2,v_1)) \big) \Big)$$

$$= \bigwedge_{b_1 \in B_1} \bigwedge_{u_1 \in U_{b_1}} \Big(\overline{f(b_1)}^{b_1}(u_1) \Rightarrow$$

$$\bigwedge_{b_2 \in B_2} \bigwedge_{a_1 \in A_1} \bigwedge_{u_2 \in U_{b_2}} \bigwedge_{v_1 \in V_{a_1}} \big(Q(b_2,a_1)(u_2,v_1) \Rightarrow (\neg \rho_{b_1,a_1}(u_1,v_1) \Rightarrow \rho_{b_2,a_2}(u_2,v_2)) \big) \Big)$$

$$= \bigwedge_{b_1 \in B_1} \bigwedge_{b_2 \in B_2} \bigwedge_{a_1 \in A_1} \bigwedge_{u_2 \in U_{b_2}} \bigwedge_{v_1 \in V_{a_1}} \bigwedge_{u_1 \in U_{b_1}} \Big(\overline{f(b_1)}^{b_1}(u_1) \Rightarrow$$

$$\big(Q(b_2,a_1)(u_2,v_1) \Rightarrow (\neg \rho_{b_1,a_1}(u_1,v_1) \Rightarrow \rho_{b_2,a_2}(u_2,v_2)) \big) \Big)$$

$$= \bigwedge_{b_2 \in B_2} \bigwedge_{u_2 \in U_{b_2}}$$

$$\Big(\bigvee_{b_1 \in B_1} \bigvee_{u_1 \in U_{b_1}} \bigvee_{a_1 \in A_1} \bigvee_{v_1 \in V_{a_1}} \overline{(f(b_1)}^{b_1}(u_1) \wedge Q(b_2,a_1)(u_2,v_1) \wedge \neg \rho_{b_1,a_1}(u_1,v_1) \Big)$$

$$\Rightarrow \rho_{b_2,a_2}(u_2,v_2) \Big)$$

$$= \bigwedge_{b_2 \in B_2} \bigwedge_{u_2 \in U_{b_2}} \overline{(q(b_2)}^{b_2}(u_2) \Rightarrow \rho_{b_2,a_2}(u_2,v_2))$$

$$= \overline{\nearrow_{\mathcal{C}_2}}(\overline{q})(v_2) = \overline{\nearrow_{\mathcal{C}_2}(q)(a_2)}(v_2)$$

where

$$q(b_2)(u_2) = \bigvee_{b_1 \in B_1} \bigvee_{u_1 \in U_{b_1}} \bigvee_{a_1 \in A_1} \bigvee_{v_1 \in V_{a_1}} \overline{f(b_1)}(u_1) \wedge Q(b_2,a_1)(u_2,v_1) \wedge \neg \rho_{b_1,a_1}(u_1,v_1)$$

Hence, $\nearrow_{\mathcal{B}}(f) = \nearrow_{\mathcal{C}_2}(q)$. So any intent of $\mathcal{B}$ is an intent of $\mathcal{C}_2$.

By using the following equality

$$(\neg \rho_{b_1,a_1}(u_1,v_1) \Rightarrow \rho_{b_2,a_2}(u_2,v_2)) = (\neg \rho_{b_2,a_2}(u_2,v_2) \Rightarrow \rho_{b_1,a_1}(u_1,v_1))$$

analogously we obtain that any extent of $\mathcal{B}$ is an extent of $\mathcal{C}_1$. Hence, $\mathcal{B}$ is a bond between $\mathcal{C}_1$ and $\mathcal{C}_2$. □

6 Conclusion

Bonds and their L-fuzzy generalizations represent a feasible way to explore the relationships between formal contexts. In this paper we have investigated the notion of a bond with respect to the heterogeneous formal contexts. In conclusion, an alternative definition of a bond provides an efficient tool to work with

the nonhomogeneous data and one can further explore this uncharted territory in formal concept analysis.

Categorical properties of heterogeneous formal contexts and bonds as morphisms between such objects and categorical relationship to homogeneous FCA categorical description will be studied in the near future.

References

1. C. Alcalde, A. Burusco, R. Fuentes-González and I. Zubia. The use of linguistic variables and fuzzy propositions in the *L*-fuzzy concept theory. *Computers & Mathematics with Applications*, 62 (8): 3111-3122, 2011.
2. L. Antoni, S. Krajči, O. Krídlo, B. Macek, and L. Pisková. Relationship between two FCA approaches on heterogeneous formal contexts. *Proceedings of CLA*, 93–102, 2012.
3. L. Antoni, S. Krajči, O. Krídlo, B. Macek, and L. Pisková. On heterogeneous formal contexts. *Fuzzy Sets and Systems*, 234: 22–33, 2014.
4. R. Bělohlávek. Fuzzy concepts and conceptual structures: induced similarities. *Proceedings of Joint Conference on Information Sciences*, 179–182, 1998.
5. R. Bělohlávek. Lattices of fixed points of fuzzy Galois connections. *Mathematical Logic Quartely*, 47(1):111–116, 2001.
6. R. Bělohlávek. Concept lattices and order in fuzzy logic. *Annals of Pure and Applied Logic*, 128(1–3):277-298, 2004.
7. R. Bělohlávek. Lattice-type fuzzy order is uniquely given by its 1-cut: proof and consequences. *Fuzzy Sets and Systems*, 143:447–458, 2004.
8. R. Bělohlávek. Sup-t-norm and inf-residuum are one type of relational product: unifying framework and consequences. *Fuzzy Sets and Systems*, 197: 45–58, 2012.
9. R. Bělohlávek. Ordinally equivalent data: A measurement-theoretic look at formal concept analysis of fuzzy attributes. *International Journal of Approximate Reasoning*, 54 (9): 1496–1506, 2013.
10. Simple proof of basic theorem for general concept lattices by Cartesian representation MDAI 2012, Girona, Catalonia, Spain, November 21-23, 2012, LNCS 7647(2012), 294-305.
11. P. Butka, J. Pócs and J. Pócsová. Representation of fuzzy concept lattices in the framework of classical FCA. *Journal of Applied Mathematics*, Article ID 236725, 7 pages, 2013.
12. P. Butka and J. Pócs. Generalization of one-sided concept lattices. *Computing and Informatics*, 32(2): 355–370, 2013.
13. C. Carpineto, and G. Romano. *Concept Data Analysis. Theory and Applications.* J. Wiley, 2004.
14. J.C. Díaz-Moreno, J. Medina, M. Ojeda-Aciego. On basic conditions to generate multi-adjoint concept lattices via Galois connections. *International Journal of General Systems*, 43(2): 149–161, 2014.
15. D. Dubois and H. Prade. Possibility theory and formal concept analysis: Characterizing independent sub-contexts. *Fuzzy sets and systems*, 196, 4–16, 2012.
16. B. Ganter and R. Wille. *Formal concept analysis.* Springer–Verlag, 1999.
17. J. Konečný and M. Ojeda-Aciego. Isotone L-bonds. *Proceedings of CLA*, 153-162, 2013.
18. J. Konečný, M. Krupka. Block Relations in Fuzzy Setting *Proceedings of CLA*, 115-130, 2011.

19. J. Konečný. Information Processing and Management of Uncertainty in Knowledge-Based Systems Comunications Computer and Information Sciences, vol. 444, 2014, pp 71-80.
20. O. Krídlo and M. Ojeda-Aciego. On the L-fuzzy generalization of Chu correspondences. *International Journal of Computer Mathematics*, 88(9):1808-1818, 2011.
21. O. Krídlo, S. Krajči, and M. Ojeda-Aciego. The category of L-Chu correspondences and the structure of L-bonds. *Fundamenta Informaticae*, 115(4):297-325, 2012.
22. O. Krídlo and M. Ojeda-Aciego. Linking L-Chu correspondences and completely lattice L-ordered sets. *Proceedings of CLA*, 233–244, 2012.
23. O. Krídlo and M. Ojeda-Aciego. CRL-Chu correspondences. *Proceedings of CLA*, 105–116, 2013.
24. O. Krídlo, P. Mihalčin, S. Krajči and L. Antoni. Formal Concept analysis of higher order. *Proceedings of CLA*, 117-128, 2013.
25. O. Krídlo and M. Ojeda-Aciego. Revising the Link between L-Chu correspondences and completely lattice L-ordered sets. *Annals of Mathematics and Artificial Inteligence*, DOI 10.1007/s10472-014-9416-8, 2014.
26. M. Krötzsch, P. Hitzler, and G.-Q. Zhang. Morphisms in context. *Lecture Notes in Computer Science*, 3596:223–237, 2005.
27. J. Medina and M. Ojeda-Aciego. Multi-adjoint t-concept lattices. *Information Sciences*, 180:712–725, 2010.
28. J. Medina and M. Ojeda-Aciego. On multi-adjoint concept lattices based on heterogeneous conjunctors. *Fuzzy Sets and Systems*, 208: 95–110, 2012.
29. J. Medina and M. Ojeda-Aciego. Dual multi-adjoint concept lattices. *Information Sciences*, 225, 47–54, 2013.
30. H. Mori. Chu Correspondences. *Hokkaido Matematical Journal*, 37:147–214, 2008.
31. H. Mori. Functorial properties of Formal Concept Analysis. Proc ICCS, *Lecture Notes in Computer Science*, 4604:505–508, 2007.
32. J. Pócs. Note on generating fuzzy concept lattices via Galois connections. *Information Sciences* 185 (1):128–136, 2012.
33. J. Pócs. On possible generalization of fuzzy concept lattices. *Information Sciences*, 210: 89–98, 2012.

Reverse Engineering Feature Models from Software Configurations using Formal Concept Analysis

R. AL-msie'deen[1], M. Huchard[1], A.-D. Seriai[1], C. Urtado[2] and S. Vauttier[2]

[1]LIRMM / CNRS & Montpellier 2 University, France
Al-msiedee, huchard, Seriai@lirmm.fr
[2]LGI2P / Ecole des Mines d'Alès, Nîmes, France
Christelle.Urtado, Sylvain.Vauttier@mines-ales.fr

Abstract. Companies often develop in a non-disciplined manner a set of software variants that share some features and differ in others to meet variant-specific requirements. To exploit existing software variants and manage them coherently as a software product line, a feature model must be built as a first step. To do so, it is necessary to extract mandatory and optional features from the code of the variants in addition to associate each feature implementation with its name. In previous work, we automatically extracted a set of feature implementations as a set of source code elements of software variants and documented the mined feature implementations based on the use-case diagrams of these variants. In this paper, we propose an automatic approach to organize the mined documented features into a feature model. The feature model is a tree which highlights mandatory features, optional features and feature groups (and, or, xor groups). The feature model is completed with requirement and mutual exclusion constraints. We rely on Formal Concept Analysis and software configurations to mine a unique and consistent feature model. To validate our approach, we apply it on several case studies. The results of this evaluation validate the relevance and performance of our proposal as most of the features and their associated constraints are correctly identified.

Keywords: Software Product Line, Feature Models, Software Product Variants, Formal Concept Analysis, Product-by-feature matrix.

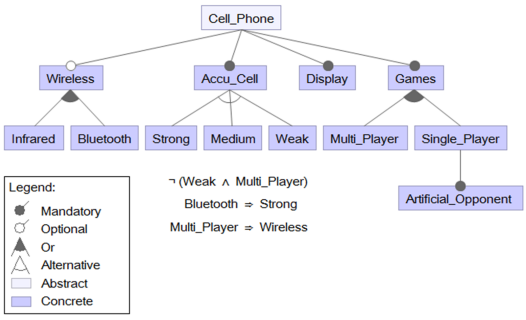
1 Introduction

To exploit existing software variants and build a software product line (SPL), a feature model (FM) must be built as a first step. To do so, it is necessary to extract mandatory and optional features in addition to associate each feature with its name. In our previous work [1,2], we have presented an approach called REVPLINE [1] to identify and document features from the object-oriented source code of a collection of software product variants.

[1] REVPLINE stands for RE-engineering Software Variants into Software Product Line.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

Dependencies between features need to be expressed via a FM which is a *de facto* standard formalism [3,4]. A FM is a tree-like hierarchy of features and constraints between them (*cf.* left side of Figure 1). FMs aim at describing the variability of a SPL in terms of features. A FM defines which feature combinations lead to valid products within the SPL (*cf.* right side of Figure 1). We illustrate our approach with the cell phone SPL FM and its 16 valid product configurations (*cf.* Figure 1) [5].



	Cell_Phone	Wireless	Infrared	Bluetooth	Accu_Cell	Strong	Medium	Weak	Display	Games	Multi_Player	Single_Player	Artificial_Opponent
P-1	×	×	×		×	×			×	×	×		
P-2	×	×		×	×	×			×	×	×		
P-3	×	×	×	×	×	×			×	×	×		
P-4	×	×	×		×		×		×	×	×		
P-5	×				×			×	×	×		×	×
P-6	×				×	×			×	×		×	×
P-7	×	×	×		×	×			×	×		×	×
P-8	×	×		×	×	×			×	×		×	×
P-9	×	×	×	×	×	×			×	×		×	×
P-10	×				×		×		×	×		×	×
P-11	×	×	×		×		×		×	×		×	×
P-12	×	×	×		×			×	×	×		×	×
P-13	×	×	×		×		×		×	×	×	×	×
P-14	×	×	×		×	×			×	×	×	×	×
P-15	×	×		×	×	×			×	×	×	×	×
P-16	×	×	×	×	×	×			×	×	×	×	×

Fig. 1. Valid product configurations of cell phone SPL feature model [5].

Figure 1 shows the FM of the *cell phone* SPL [5]. The *Cell_Phone* feature is the root feature of this FM; hence it is selected in every program configuration. It has three mandatory child features (*i.e.*, the *Accu_Cell*, *Display* and *Games* features), which are also selected in every product configuration as their parent is always included. The children of the *Accu_Cell* feature form an *exclusive-or* relation, meaning that the programs of this SPL include exactly one out of the three *Strong*, *Medium* or *Weak* features. The *Multi_Player* and *Single_Player* features constitute an *inclusive-or*, which necessitates that at least one of these two features is selected in any valid program configuration. *Single_Player* has *Artificial_Opponent* as a mandatory child feature. The *Wireless* feature is an optional child feature of root; hence it may or may not be selected. Its *Infrared* and *Bluetooth* child features form an *inclusive-or* relation, meaning that if a program includes the *Wireless* feature then at least one of its two child features has to be selected as well. The cell phone SPL also introduces three cross-tree constraints. While the *Multi_Player* feature cannot be selected together with the *Weak* feature, it cannot be selected without the *Wireless* feature. Lastly, the *Bluetooth* feature requires the *Strong* feature.

Galois lattices and concept lattices [6] are core structures of a data analysis framework (Formal Concept Analysis) for extracting an ordered set of con-

cepts from a dataset, called a formal context, composed of objects described by attributes. In our approach, we consider the AOC-poset (for Attribute-Object-Concept poset) [7], which is the sub-order of the concept lattice restricted to attribute-concepts and object-concepts. Attribute-concepts (*resp.* object-concepts) are the highest (*resp.* lowest) concepts that introduce each attribute (*resp.* object). AOC-posets scale much better than lattices. For applying Formal Concept Analysis (FCA) we used the Eclipse eRCA platform[2].

Manual construction of a FM is both time-consuming and error-prone [8], even for a small set of configurations [9]. The existing approaches to extract FM from product configurations [8,10] suffer from a lot of challenges. The main challenge is that numerous candidate FMs can be extracted from the same input product configurations, yet only a few of them are meaningful and correct, while in our work we synthesize an accurate and meaningful FM using FCA. Moreover the majority of these approaches extract a basic FM without constraints between its features [11] while, in our work, we extract all kinds of FM constraints.

The remainder of this paper is structured as follows: Section 2 presents the reverse engineering FM process step-by-step. Next, Section 3 presents the way that we propose to evaluate the obtained FMs. Section **??** describes the experimentation and threats to the validity. Section 4 discusses the related work. Finally, in Section 5, we conclude this paper.

2 Step-by-Step FM Reverse Engineering

This section presents step-by-step the FM reverse engineering process. According to our approach, we identify the FM in seven steps as detailed in the following, using strong properties of FCA to group features among product configurations. The AOC-poset is built from a set of known products, and thus does not represent all possible products. Thus, the FM structure has to be considered only as a candidate feature organization that can be proposed to an expert. The algorithm is designed such that all existing products (used for construction of candidate FM) are covered by the FM. Besides, it allows to define possible unused close variants.

The first step of our FM extraction process is the identification of the AOC-poset. First, a *formal context*, where objects are software product variants and attributes are features (*cf.* Figure 1), is defined. The corresponding *AOC-poset* is then calculated. The intent of each concept represents features common to two or more products or unique to one product. As AOC-posets are ordered, the intent of the most general (*i.e.*, top) concept gathers mandatory features that are common to all products. The intents of all the remaining concepts represent the optional features. The extent of each of these concepts is the set of products sharing these features (*cf.* Figure 2). In the following algorithms, for a Concept C, we call $intent(C)$, $extent(C)$, $simplified\ intent(C)$, and $simplified\ extent(C)$ its associated sets. Efficient algorithms can be found in [7].

The other steps are presented in the next sections.

[2] The eRCA : `http://code.google.com/p/erca/`

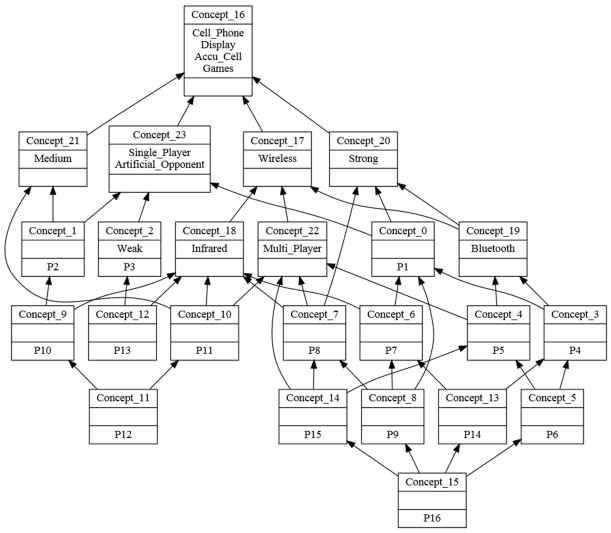


Fig. 2. The AOC-poset for the formal context of Figure 1.

2.1 Extracting root feature and mandatory features

Algorithm 1 is a simple algorithm for building the Base node (*cf.* Figure 3). Features in the top concept of the AOC-poset (Concept_16) are used in every product configuration. The *Cell_Phone* feature is the root feature of the cell phone FM (line 5). Then a mandatory Base node is created (lines 8,9). It is linked to nodes created to represent all the other features in the top concept, *i.e.*, Accu_Cell, Display and Games (lines 12-16).

2.2 Extracting atomic set of features (AND-group)

Algorithm 2 is a simple algorithm for building AND-groups of features (excluding all the mandatory features, line 3). An AND-group of features is created (line 8) to group optional features that appear in the same simplified intent (test line 6), meaning that these features are always used together in all the product configurations where they appear. Lines 12-16, nodes are created for every feature of the AND-group and they are attached to an *And* node. For instance, *Concept_23* in Figure 2 has a simplified intent with two features, *Single_Player* and *Artificial_Opponent*, leading to the *And* node of Figure 3.

2.3 Extracting exclusive-or relation

Features that form exclusive-or relation can be identified in the concept lattice using the meet (denoted by $\sqcap$) lattice operation [12], which amounts to compute

Algorithm 1: ComputeRootAndMandatoryFeature

```
// Top concept ⊤
∃ F ∈ A, which represents the name of the soft. family with F in feature set of ⊤
Data: AOC_K, ≤_s: the AOC-poset associated with K
Result: part of the FM containing root and mandatory features
// Compute the root Feature
CFS ← intent(⊤)
Create node root, label (root) ← F, type (root) ← abstract
CFS′ ← CFS \ {F}
if CFS′ ≠ ∅ then
    Create node base with label (base) ← "Base"
    type (base) ← abstract
    Create edge e = (root, base)
    type (e) ← mandatory
    for each F_e in CFS′ do
        Create node feature, with label (feature) ← F_e
        type (feature) ← concrete
        create edge e = (base, feature)
        type (e) ← mandatory
```

Algorithm 2: Compute*AtomicSetOfFeatures* (and groups)

```
Data: AOC_K, ≤_s: the AOC-poset associated with K
Result: part of the FM with and groups of features
// Compute atomic set of features
// Feature List (FL) is the list of all features (FL = A in K=(O, A, R)).
FL′ ← FL \ CFS // FL \ intent(⊤)
AsF ← ∅
int count ← 1
for each concept C ≠ ⊤ such that | simplified intent(C) | ≥ 2 do
    AsF ← AsF ∪ simplified intent(C)
    Create node and with label (and) ← "AND"+ count
    type (and) ← abstract
    create edge e = (root, and)
    type (e) ← optional
    for each F in simplified intent(C) do
        create node feature, with label (feature) ← F
        type (feature) ← concrete
        create edge e =(and, feature)
        type (e) ← mandatory
```

the greatest lower bounds in the AOC-poset. If a feature A is introduced in concept C_1, a feature B is introduced in concept C_2 and $C_1 \sqcap C_2 = \bot$ (and extent$(\bot) = \emptyset$), that is, if the bottom of the lattice is the greatest lower bound of C_1 and C_2, the two features never occur together in a product. In our current

approach, we only build a single Xor group of features, when any group of mutually exclusive features exists. Computing exclude constraints (*see* Section 2.6) will deal with the many cases where several Xor group of features exist (a set of exclude constraints defining mutual exclusion is equivalent to a Xor group).

Algorithm 3 is a simple algorithm for building the single Xor group of features. The principle is to traverse the set of super-concepts of each minimum elements of the AOC-poset and to keep the concepts that are the super-concepts of only one minimum concept. Only features that are not used in the previous steps are considered in FL" (line 2). Lines 6-10, in our example, we consider the three minimum concepts Concept_11, Concept_12 and Concept_15. The many SSC sets are the sets of super-concepts for Concept_11, Concept_12 and Concept_15. $\mathcal{C}xor$ is the set of all concepts, except Concept_11, Concept_12 and Concept_15. Lines 11-15 only keep in $\mathcal{C}xor$ concepts that do not appear in two SSC sets. $\mathcal{C}xor$ contains concepts number 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 13, 14, 19, 20 and 21. Line 16 eliminates Concept_19 which is not a maximum. As there are three features (Medium, Strong, Weak, from Concept_21, Concept_20, and Concept_2 respectively) that are in FL" and in the simplified intent of concepts of $\mathcal{C}xor$ (line 18), an Xor node is created and linked to the root (lines 19-26). Then, on lines 27-33, nodes are created for the features and linked to the Xor node. Figure 3 shows this Xor node.

2.4 Extracting inclusive-or relation

Optional features are features that are used in some (but not all) product configurations. There are many ways of finding and organizing them. Algorithm 4 is a simple algorithm for building the Or group of features. In our approach, we pruned the AOC-poset by removing the top concept, concepts that correspond to AND groups of features, and concepts that correspond to features that form an exclusive-or relation. The remaining concepts define features that are grouped (lines 8-12) into an Or node (created and linked to the root on lines 4-7). In the AOC-poset of Figure 2, the *Wireless*, *Infrared*, *Bluetooth*, and *Multi_Player* features form an inclusive-or relation (*cf.* Figure 3).

2.5 Extracting require constraints

Algorithm 5 is a simple algorithm for identifying require constraints. A require constraint, *e.g.*, saying "variable feature A always requires variable feature B", can be extracted from the lattice via implications. We say that A implies B (written $A \rightarrow B$). The require constraints can be identified in the AOC-poset: when a feature F_1 is introduced in a subconcept of the concept that introduces another feature F_2, there is an implication $F_1 \rightarrow F_2$. We only consider the transitive reduction of the AOC-poset limited to Attribute-concepts (line 2) and features that are in simplified intents (line 3-4). In the AOC-poset of Figure 2, we find 6 require constraints from the transitive reduction of the AOC-poset to

Algorithm 3: Compute*Exclusive-or*Relation (Xor)

Data: AOC_K, $\leq_s$: the AOC-poset associated with K
Result: part of the FM with *XOR* group of features

// Compute exclusive-or relation
FL$''$ ← FL$'$ \ AsFs
$\mathcal{C}xor \leftarrow \emptyset$
SSCS ← ∅ // set of super-concept sets
Minimum-set ← ∅
for *each minimum of* AOC_K *denoted by* m **do**
    Let *SSC* the set of super-concepts of m (except $\top$)
    SSCS ← SSCS ∪ {SSC}
    Minimum-set ← Minimum-set ∪ {m}
    $\mathcal{C}xor \leftarrow \mathcal{C}xor \cup$ SSC
while *SSCS* $\neq \emptyset$ **do**
    SSC-1 ← any element in (SSCS)
    SSCS ← SSCS \ SSC-1
    for *each SSC-2 in SSCS* **do**
        $\mathcal{C}xor \leftarrow \mathcal{C}xor$ \ (SSC-1 ∩ SSC-2)
$\mathcal{C}xor \leftarrow$ Max($\mathcal{C}xor$)
XFS ← ∅
if $|\mathcal{C}xor| > 1$ *and* $|FL'' \cap \cup_{C \in \mathcal{C}xor} simplified\ intent(C)| > 1$ **then**
    Create node *xor* with label (*xor*) ← "XOR"
    type (*xor*) ← abstract
    create edge e = (*root*, *xor*)
    // if all products are covered by $\mathcal{C}xor$
    if $\cup_{C \in \mathcal{C}xor} extent(C) = O$ **then**
        type (e) ← mandatory
    else
        type (e) ← optional
    for *each concept* $C \in \mathcal{C}xor$ **do**
        for *each F in simplified intent*($\mathcal{C}$) $\cap FL''$ **do**
            create node *feature*, with label (*feature*) ← F
            type (*feature*) ← concrete
            create edge e = (*xor*, *feature*)
            type (e) ← alternative
            XFS ← XFS ∪ F

attribute-concepts (*cf.* Figure 3). Remark that implications ending to mandatory features are useless because they are represented in the FM by the Base node.

2.6 Extracting exclude constraints

In our current proposal, we compute binary exclude constraints $\neg(A \wedge B)$ under the condition that A and B are not both linked to the *Or* group. To mine

Algorithm 4: ComputeInclusive-orRelation (Or)

```
Data: AOC_K, ≤_s: the AOC-poset associated with K
Result: part of the FM with OR group of features
1  // Compute inclusive-or relation
2  FL''' ← FL'' \ XFS
3  if FL''' ≠ ∅ then
4      Create node or with label (or) ← "OR"
5      type (or) ← abstract
6      create edge e = (root, or)
7      type (e) ← optional
8      for each F in FL''' do
9          create node feature, with label (feature) ← F
10         type (feature) ← concrete
11         create edge e = (or, feature)
12         type (e) ← Or
```

Algorithm 5: Compute*Require*Constraint ($Requires$)

```
Data: AC_K, ≤_s: the AC-poset associated with K
Result: Require - the set of require constraints
1  Require ← ∅
2  for each edge (C1, C2) = e in transitive reduction of AC-poset do
3  for all f1, f2 with f1 ∈ simplified intent(C1) and f2 ∈ simplified intent(C2) do
4      Require ← Require ∪ {f1 ⟶ f2}
```

exclude constraints from an AOC-poset, we use the meet[3] of the introducers of the two involved features. For example, the meet of Concept_2 which introduces *Weak* and Concept_22 which introduces *Multi_Player* is the bottom (in the whole lattice). In the AOC-poset they don't have a common lower bound. We can thus deduce $\neg(Weak \wedge Multi\_Player)$. In the AOC-poset of Figure 2, there are three exclude constraints (*cf.* Figure 3). Algorithm 6 is a simple algorithm for identifying exclude constraints. It compares features that are below the OR group with each set of features in the intent of a minimum (line 4), in order to determine which are incompatible: this is the case for a pair (f1, f2) where f1 is in the OR group and not in the minimum intent, and f2 is in the minimum intent but not in the OR group (lines 6-10). Figure 3 shows the resulting FM based on the product configurations of Figure 1.

[3] in the lattice

Algorithm 6: Compute*Exclude*Constraint (*Excludes*)

Data: AOC_K, $\leq_s$: the AOC-poset associated with K
Result: Exclude - the set of exclude constraints.

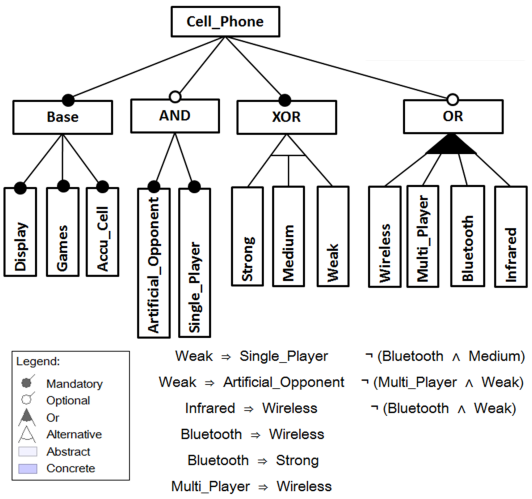
```
// Minimum-set from Algorithm 3
// FL''' from Algorithm 4
Exclude ← ∅
for each P ∈ Minimum-set do
    Pintent ← intent(P) \ intent(⊤)
    Opt-feat-set ← FL''' \ (FL''' ∩ Pintent)
    Super-feat-set ← Pintent \ (FL''' ∩ Pintent)
    if Opt-feat-set ≠ ∅ and Super-feat-set ≠ ∅ then
        for each f1 ∈ Opt-feat-set, f2 ∈ Super-feat-set do
            Exclude ← Exclude ∪ {¬(f1 ∧ f2)}
```

3 Experimentation

In order to evaluate the mined FM we rely on the SPLOT homepage[4] and the FAMA Tool[5]. Our implementation[6] converts the FM that has been drawn using SPLOT homepage into the format of FAMA. Then, we can easily generate a file containing all valid product configurations [13]. Figure 3 shows all valid product configurations for the mined FM by our approach (the first 16 product configurations are the same as in Figure 1). We compare the sets of configurations defined by the two FMs (*i.e.*, the initial FM compared to the mined FM). The mined FM introduces 15 extra product configurations which correspond to feature selection constraints that have not been detected by our algorithm.

Evaluation Metrics: In our work, we rely on *precision*, *recall* and *F-measure* metrics to evaluate the mined FM. All measures have values in [0, 1]. If recall equals 1, all relevant product configurations are retrieved. However, some retrieved product configurations might not be relevant. If precision equals 1, all retrieved product configurations are relevant. Nevertheless, relevant product configurations might not be retrieved. If F-Measure equals 1, all relevant product configurations are retrieved. However, some retrieved product configurations might not be relevant. F-Measure defines a trade-off between precision and recall, so that it gives a high value only in cases where both recall and precision are high. The result of the product configurations that are identified by the mined cell phone FM is as follow: (*precision*: 0.51), (*recall*: 1.00) and (*F-Measure*: 0.68). The recall measure is 1 by construction, due to the fact that the algorithm was designed to cover existing products.

[4] SPLOT homepage : `http://gsd.uwaterloo.ca:8088/SPLOT/`
[5] FAMA Tool Suite : `http://www.isa.us.es/fama/`
[6] Source Code : `https://code.google.com/p/sxfmtofama/`



	Cell_Phone	Wireless	Infrared	Bluetooth	Accu_Cell	Strong	Medium	Weak	Display	Games	Multi_Player	Single_Player	Artificial_Opponent
P-17	×				×	×			×	×			
P-18	×	×			×	×			×	×			
P-19	×	×	×		×	×			×	×			
P-20	×	×		×	×	×			×	×			
P-21	×	×	×	×	×	×			×	×			
P-22	×				×		×		×	×			
P-23	×	×			×		×		×	×			
P-24	×	×	×		×		×		×	×			
P-25	×	×			×	×			×	×	×		
P-26	×	×			×		×		×	×	×		
P-27	×	×			×	×			×	×		×	×
P-28	×	×			×		×		×	×		×	×
P-29	×	×			×			×	×	×		×	×
P-30	×	×			×	×			×	×	×	×	×
P-31	×	×			×		×		×	×	×	×	×

Fig. 3. The mined FM and its extra product configurations.

To validate our approach[7], we ran experiments on 7 case studies: ArgoUML-SPL [1], mobile media software variants [2], public health complaint-SPL[8], video on demand-SPL [8,3,14], wiki engines [10], DC motor [11] and cell phone-SPL [5]. Table 1 summarizes the obtained results.

Results show that precision appears to be not very high for all case studies. This means that many of the identified product configurations of the mined FM are extra configurations (not in the initial set that is defined by the original FM). Considering the recall metric, its value is 1 for all case studies. This means that product configurations defined by the initial FM are included in the product configurations derived from the mined FM. Experiments show that if the generated AOC-poset has only one bottom concept there is no exclusive-or relation or exclude constraints from the given product configurations. In our work, the mined FM defines more configurations than the initial FM. The reason behind this limitation is that some feature selection constraints are not detected. Nevertheless, the AOC-poset contains information for going beyond this limitation. We plan to enhance our algorithm to deal with that issue, at the price of an increase of complexity.

4 Related Work

For the sake of brevity, we describe only the work that most closely relates to ours. The majority of existing approaches are designed to reverse engineer FM

[7] Source code: `https://code.google.com/p/refmfpc/`
[8] `http://www.ic.unicamp.br/~tizzei/phc/`

Table 1. The results of configurations that are identified by the mined FMs.

#	case study	Number of Products	Number of Features	Group of Features				CTCs		Execution times (in ms)	Evaluation Metrics		
				Base	Atomic Set of Features	Inclusive-or	Exclusive-or	Requires	Excludes		Precision	Recall	F-Measure
1	ArgoUML-SPL	20	11	×		×		×		509	0.60	1.00	0.75
2	Mobile media	8	18	×	×	×				441	0.68	1.00	0.80
3	Health complaint-SPL	10	16	×	×	×		×		439	0.57	1.00	0.72
4	Video on demand	16	12	×	×	×		×		572	0.66	1.00	0.80
5	Wiki engines	8	21	×	×	×	×	×	×	555	0.54	1.00	0.70
6	DC motor	10	15			×		×		444	0.83	1.00	0.90
7	Cell phone-SPL	16	13	×	×	×	×	×	×	486	0.51	1.00	0.68

from high level models (*e.g.*, product descriptions) [10,14]. Some approaches offer an acceptable solution but are not able to identify important parts of FM such as cross-tree constraints, and-group, or-group, xor-group [11]. The main challenge of works that reverse engineer FMs from product configurations ([8,3]) is that numerous candidate FMs can be extracted from the same input configurations, yet only a few of them are meaningful and correct. The majority of existing approaches are designed to identify the dependencies between features regardless of FM hierarchy [8]. Work that relies on FCA to extract a FM does not fully exploit resulting lattices. In [11], authors rely on FCA to extract a basic FM without cross-tree constraints, while in [12], authors use FCA as a tool to understand the variability of existing SPL based on product configurations. Their work does not produce FMs. In our work, we rely on FCA to extract FMs from the software configurations. The resulting FMs exactly describe the given product configuration set. The proposed approach is able to identify all parts of FMs.

5 Conclusion

In this paper, we proposed an automatic approach to extract FMs from software variants configurations. We rely on FCA to extract FMs including configuration constraints. We have implemented our approach and evaluated its produced results on several case studies. The results of this evaluation showed that the resulting FMs exactly describe the given product configuration set. The FMs are generated in very short time, because our FCA tool (based on traversals of the AOC-poset) scales significantly better than the standard FCA approaches to calculate and traverse the lattices. The current work extracts a FM with two levels of hierarchy. As a perspective of this work, we plan to enhance the extracted FM by increasing the levels of hierarchy based on AOC-poset structure and to avoid allowing the FM to represent extra configurations.

Acknowledgment The authors would like to thank the reviewers for their valuable remarks that helped improve the paper. This work has been supported by the CUTTER ANR-10-BLAN-0219 project.

References

1. Al-Msie'deen, R., Seriai, A., Huchard, M., Urtado, C., Vauttier, S., Salman, H.E.: Mining features from the object-oriented source code of a collection of software variants using formal concept analysis and latent semantic indexing. In: SEKE '13. (2013) 244–249
2. Al-Msie'deen, R., Seriai, A., Huchard, M., Urtado, C., Vauttier, S.: Documenting the mined feature implementations from the object-oriented source code of a collection of software product variants. In: SEKE '14. (2014) 264–269
3. Acher, M., Baudry, B., Heymans, P., Cleve, A., Hainaut, J.L.: Support for reverse engineering and maintaining feature models. In: VaMoS '13, New York, NY, USA, ACM (2013) 20:1–20:8
4. She, S., Lotufo, R., Berger, T., Wasowski, A., Czarnecki, K.: Reverse engineering feature models. In: ICSE '11, New York, NY, USA, ACM (2011) 461–470
5. Haslinger, E.N.: Reverse engineering feature models from program configurations. Master's thesis, Johannes Kepler University Linz, Linz, Austria (September 2012)
6. Ganter, B., Wille, R.: Formal concept analysis - mathematical foundations. Springer (1999)
7. Berry, A., Gutierrez, A., Huchard, M., Napoli, A., Sigayret, A.: Hermes: a simple and efficient algorithm for building the AOC-poset of a binary relation, Annals of Mathematics and Artificial Intelligence (may 2014)
8. Haslinger, E.N., Lopez-Herrejon, R.E., Egyed, A.: Reverse engineering feature models from programs' feature sets. In: WCRE '11, IEEE (2011) 308–312
9. Andersen, N., Czarnecki, K., She, S., Wasowski, A.: Efficient synthesis of feature models. In: SPLC (1), ACM (2012) 106–115
10. Acher, M., Cleve, A., Perrouin, G., Heymans, P., Vanbeneden, C., Collet, P., Lahire, P.: On extracting feature models from product descriptions. In: VaMoS '12, New York, NY, USA, ACM (2012) 45–54
11. Ryssel, U., Ploennigs, J., Kabitzsch, K.: Extraction of feature models from formal contexts. In: SPLC '11, New York, NY, USA, ACM (2011) 4:1–4:8
12. Loesch, F., Ploedereder, E.: Optimization of variability in software product lines. In: SPLC '07, Washington, DC, USA, IEEE Computer Society (2007) 151–162
13. Benavides, D., Segura, S., Ruiz-Cortés, A.: Automated analysis of feature models 20 years later: A literature review. Inf. Syst. **35**(6) (September 2010) 615–636
14. Lopez-Herrejon, R.E., Galindo, J.A., Benavides, D., Segura, S., Egyed, A.: Reverse engineering feature models with evolutionary algorithms: An exploratory study. In: SSBSE, Springer (2012) 168–182

An Algorithm for the Multi-Relational Boolean Factor Analysis based on Essential Elements

Martin Trnecka, Marketa Trneckova

Data Analysis and Modeling Lab (DAMOL)
Department of Computer Science, Palacky University, Olomouc
martin.trnecka@gmail.com, marketa.trneckova@gmail.com

Abstract. The Multi-Relational Boolean factor analysis is a method from the family of matrix decomposition methods which enables us analyze binary multi-relational data, i.e. binary data which are composed from many binary data tables interconnected via relation. In this paper we present a new Boolean matrix factorization algorithm for this kind of data, which use the new knowledge from the theory of the Boolean factor analysis, so-called essential elements. We show on real dataset that utilizing essential elements in the algorithm leads to better results in terms of quality and the number of obtained multi-relational factors.

1 Introduction

The Boolean matrix factorization (or decomposition), also known as the Boolean factor analysis, has gained interest in the data mining community. Methods for decomposition of multi-relational data, i.e. complex data composed from many data tables interconnected via relations between objects or attributes of this data tables, were intensively studied, especially in the past few years. Multi-relational data is a more truthful and therefore often also more powerful representation of reality. An example of this kind of data can be an arbitrary relational database. In this paper we are focused on the subset of multi-relational data, more precisely on the multi-relational Boolean data. In this case data tables and relations between them contain only 0s and 1s.

It is important to say that many real-word data sets are more complex than one simple data table. Relations between this tables are crucial, because they carry additional information about the relationship between data and this information is important for understanding data as a whole. For this reason methods which can analyze multi-relational data usually takes into account relations between data tables unlike classical Boolean matrix factorization methods which can handle only one data table.

The Multi-Relational Boolean matrix factorization (MBMF) is used for many data mining purposes. The basic task is to find new variables hidden in data, called multi-relational factors, which explain or describe the original input data. There exist several ways how to represent multi-relational factors. In this work

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

we adopt settings from [7], where is the multi-relational factor represented as an ordered set of classic factors from data tables, always one factor from each data table. The fact, that classic factors are connected into multi-relational factor is matter of semantic of relation between data tables.

The main problem is how to connect classic factors into one multi-relational. The main aim of this work is to propose a new algorithm which utilize so-called essential elements from the theory of Boolean matrices. The essential elements provide information about factors which cover a particular part of data tables. This information can be used for a better connection of classic factors into one multi-relational factor.

Another thing is the number of obtained factors. In classical settings we want the number of obtained factors as small as a possible. In the literature can be found two main views on this requirement. In the first case we want to obtain the particular number of factors. In the second case we want to obtain factors that explain prescribed portion of data. In both cases we want to obtain the most important factors. For more details see [1]. We emphasize this fact and we reflect it in designing of our algorithm. Both views can be transferred to multi-relational case. The first one is straightforward, the second one is a little bit problematic because multi-relational factors may not be able explain the whole data. This is correct, because multi-relational factors carry different information than classical factors. We discuss this issue later in the paper.

2 Preliminaries and basic notions

We assume familiarity with the basic notions of the Formal concept analysis [4], which provides a basic framework for dealing with factors and the Boolean matrix factorization (BMF) [2]. The main goal of classical BMF is to find a decomposition $C = A \circ B$, where C is input data table, A represent object-factor data table (or matrix) and B represent factor-attribute data table (or matrix). The product $\circ$ is the Boolean matrix product, defined by

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

where $\bigvee$ denotes maximum (truth function of logical disjunction) and $\cdot$ is the usual product (truth function of logical conjunction). Decomposition C into $A \circ B$ corresponds to discovery factors which explain the data. Factors in classical BMF can be seen as formal concepts [2], i.e. entity with the extent part and the intent part. This leads to clear interpretation of factors. Another benefit of using FCA as a basic framework is that matrices A and B can be constructed from the subset of all formal concepts. Let

$$\mathcal{F} = \{\langle A_1, B_1 \rangle, \ldots, \langle A_k, B_k \rangle\} \subseteq \mathcal{B}(X, Y, C),$$

where $\mathcal{B}(X, Y, C)$ represents a set of all formal concepts of data table, which can be seen as a formal context $\langle X, Y, C \rangle$, where X is a set of objects, Y is a set of attributes and C is a binary relation between X and Y. Matrices A and B are constructed in the following way:

$$(A)_{il} = \begin{cases} 1 \text{ if } i \in A_l \\ 0 \text{ if } i \notin A_l \end{cases} \quad (B)_{lj} = \begin{cases} 1 \text{ if } j \in B_l \\ 0 \text{ if } j \notin B_l \end{cases}$$

for $l = 1, \ldots, k$. In other words, A is composed from characteristic vectors A_l. Similarly for B.

In a multi-relation environment we have a set of input data tables C_1, $C_2, \ldots C_n$ and a set of relations $\mathcal{R}_{ij}$, where $i, j \in \{1, \ldots, n\}$, between C_i and C_j. The multi-relation factor on data tables $C_1, C_2, \ldots C_n$ is an ordered n-tuple $\langle F_1^{i_1}, F_2^{i_2}, \ldots F_n^{i_n} \rangle$, where $F_j^{ij} \in \mathcal{F}_j$, $j \in \{1, \ldots, n\}$ ($\mathcal{F}_j$ denotes a set of classic factors of data table C_j) and satisfying relations $\mathcal{R}_{C_l C_{l+1}}$ or $\mathcal{R}_{C_{l+1} C_l}$ for $l \in \{1, \ldots, n-1\}$.

Example 1. Let us have two data tables C_1 (Table 1) and C_2 (Table 2). Moreover, we consider relation $\mathcal{R}_{C_1 C_2}$ (Table 3) between objects of the first data table and attributes of the second one.

Table 1: C_1

	a	b	c	d
1		×	×	×
2	×		×	
3		×		×
4	×	×	×	×

Table 2: C_2

	e	f	g	h
5	×			×
6		×	×	
7	×	×	×	
8			×	×

Table 3: $\mathcal{R}_{C_1 C_2}$

	e	f	g	h
1		×	×	
2	×		×	
3	×	×		×
4	×	×	×	×

Classic factors of data table C_1 are for example: $F_1^{C_1} = \langle \{1, 4\}, \{b, c, d\} \rangle$, $F_2^{C_1} = \langle \{2, 4\}, \{a, c\} \rangle$, $F_3^{C_1} = \langle \{1, 3, 4\}, \{b, d\} \rangle$ and factors of the second table C_2 are: $F_1^{C_2} = \langle \{6, 7\}, \{f, g\} \rangle$, $F_2^{C_2} = \langle \{5\}, \{e, h\} \rangle$, $F_3^{C_2} = \langle \{5, 7\}, \{e\} \rangle$, $F_4^{C_2} = \langle \{8\}, \{g, h\} \rangle$. These factors can be connected with using a relation $\mathcal{R}_{C_1 C_2}$ into multi-relational factors in several ways. In [7] were introduced three approaches how to manage this connections. We use the narrow approach from [7], which seems to be the most natural, and we obtain two multi-relational factors $\langle F_1^{C_1}, F_1^{C_2} \rangle$ and $\langle F_3^{C_1}, F_1^{C_2} \rangle$. The idea of the narrow approach is very simple. We connect two factors $F_i^{C_1}$ and $F_j^{C_2}$ if the non-empty set of attributes (if such exist), which are common (in the relation $\mathcal{R}_{C_1 C_2}$) to all objects from the first factor $F_i^{C_1}$, is the subset of attributes of the second factor $F_j^{C_2}$.

The previous example also demonstrate the most problematic part of MBMF. Usually is problematic to connect all factors from each data table. The result of this is a small number of connections between them. This leads to problematic selection of quality multi-relational factors. The reason for a small number of connections between factors is that classic factors are selected without taking relation into account.

Another very important notion for our work are so-called essential elements presented in [1]. Essential elements in the Boolean data table are entries in this data table which are sufficient for covering the whole data table by factors

(concepts), i.e. if we take factors which cover all these entries, we automatically cover all entries of the input data table. Formally, essential elements in the data table $\langle X, Y, C\rangle$ are defined via minimal intervals in the concept lattice. The entry C_{ij} is essential iff interval bounded by formal concepts $\langle i^{\uparrow\downarrow}, i^{\uparrow}\rangle$ and $\langle j^{\downarrow}, j^{\downarrow\uparrow}\rangle$ is non-empty and minimal w.r.t. $\subseteq$ (if it is not contained in any other interval). We denote this interval by $\mathcal{I}_{ij}$. If the table entry C_{ij} is essential, then interval $\mathcal{I}_{ij}$ represents the set of all formal concepts (factors) which cover this entry. Very interesting property of essential elements, which is used in our algorithm, is that is sufficient take only one arbitrary concept from each interval to create exact Boolean decomposition of $\langle X, Y, C\rangle$. For more details about essential elements we refer to [1].

3 Related work

There are several papers about classical BMF [1, 2, 5, 8, 10, 12], but this methods can handle only one data table. In the literature, we can found a wide range of theoretical and application papers about the multi-relation data analysis (see overview [3]), but many times were shown that these approaches are suitable only for ordinal data. The multi-relational Boolean factor analysis is more specific. The most relevant paper for our work is [7], where was introduced the basic idea that multi-relational factors are composed from classical factors which are interconnected via relation between data tables. There were also introduced three approaches how to create multi-relational factors, but an effective algorithm is missing.

The Boolean multi-relational patterns and its extraction are subject of a paper [11]. Differently from our approach data are represented via k-partite graphs. There are considered only relations between attributes and data tables contain only one single attribute. Patterns in [11] are different from our multi-relational factors (are represented as k-clique in data) and also carry different information. In [11] there is also considered other kind of measure of quality of obtained patterns which is based on entropy.

Another relevant work is [6] where were introduced the Relational Formal Concept Analysis as a tool for analyzing multi-relational data. Unlike from [6] our approach extracts a different kind of patterns. For more details see [7]. MBMF is mentioned indirectly in a very specific and limited form in [9] as the Joint Subspace Matrix Factorization.

Generally the idea of connection patterns from various data tables is not new. It can be found in the social network analysis or in the field of recommendation systems. The main advantage of our approach is that patterns are Boolean factors that carry significant information and the second important advantage is that we deliver the most important factors (factors which describe the biggest portion of input data) before others, i.e. the first obtained factor is the most important.

4 Algorithm for MBMF

Before we present the algorithm for the MBMF we show on a simple example basic ideas that are behind the algorithm. For this purpose we take the example from the previous part. As we mentioned above if we take tables C_1, C_2 and relation $R_{C_1C_2}$, we obtain with the narrow approach two connections between factors, i.e. two multi-relational factors. These factors explain only 60 percent of data. There usually exist more factorizations of Boolean data table. Factors in our example were obtained with using GreConD algorithm from [2]. GreConD algorithm select in each iteration a factor which covers the biggest part of still uncovered data. Now we are in the situation, where we want to obtain a different set of factors, with more connections between them. For this purpose we can use essential elements. Firstly we compute essential parts of C_1 (denoted $Ess(C_1)$) and C_2 (denoted $Ess(C_1)$). With the essential part of data table we mean all essential elements (tables 1 and 2).

Table 4: $Ess(C_1)$

	a	b	c	d
1			×	
2	×			
3		×		×
4				

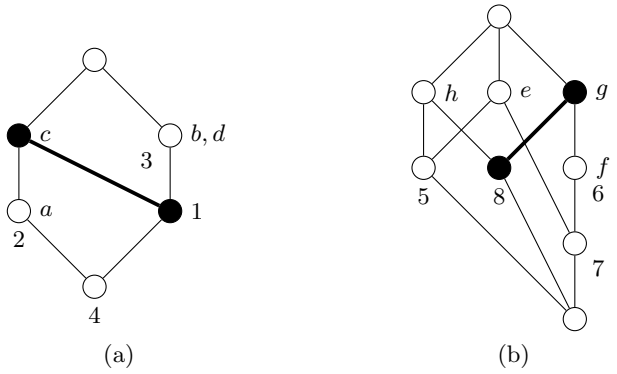
Table 5: $Ess(C_2)$

	e	f	g	h
5	×			×
6		×		
7	×			
8			×	×

Each essential element in $Ess(C_1)$ is defined via interval in concept lattice of C_1 (Fig. 1a) and similarly for essential elements in $Ess(C_2)$ (Fig 1b). In Fig. 1a is highlighted interval $\mathcal{I}_{1c}$ corresponding to essential element $(C_1)_{1c}$. In Fig. 1b is highlighted interval corresponding to essential element $(C_2)_{8g}$. Let us note that concept lattices here are only for illustration purpose. For computing $Ess(C_1)$ and $Ess(C_2)$ is not necessary to construct concept lattices at all. Now, if we use the fact that we can take an arbitrary concept (factor) from each interval to obtain a complete factorization of data table, we have several options which concepts can be connect into one. More precisely we can take two intervals and try to connect each concept from the first interval with concepts from the second one. Again, we obtain full factorization of input data tables, but now we can select factors with regard to a relation between them.

For example, if we take highlighted intervals, we obtain possibly four connections. First highlighted interval contains two concepts $c_1 = \langle\{1,2,4\},\{c\}\rangle$ and $c_2 = \langle\{1,4\},\{b,c,d\}\rangle$. Second consist of concepts $d_1 = \langle\{6,7,8\},\{g\}\rangle$ and $d_2 = \langle\{8\},\{g,h\}\rangle$. Only two connections ($c_1$ with d_1 and c_1 with d_2) satisfy relation $\mathcal{R}_{C_1C_2}$, i.e. can be connected.

For two intervals it is not necessary to try all combination of factors. If we are not able to connect concept $\langle A, B\rangle$ from the first interval with concept $\langle C, D\rangle$ from the second interval, we are not able connect $\langle A, B\rangle$ with any concept $\langle E, F\rangle$ from the second interval, where $\langle C, D\rangle \subseteq \langle E, F\rangle$. Also if we are not



(a) (b)

Fig. 1: Concept lattices of C_1 (a) and C_2 (b)

able to connect concept $\langle A, B\rangle$ from the first interval with concept $\langle E, F\rangle$ from the second interval, we are not able connect any concept $\langle C, D\rangle$ from the first interval, where $\langle C, D\rangle \subseteq \langle A, B\rangle$, with concept $\langle E, F\rangle$. Let us note that $\subseteq$ is classical subconcept-superconcept ordering.

Even if we take this search space reduction into account, search in this intervals is still time consuming. We propose an heuristic approach which takes attribute concepts in intervals of the second data table, i.e. the bottom elements in each interval. In intervals of the first data table we take greatest concepts which can be connected via relation, i.e. set of common attributes in relation is non-empty. The idea behind this heuristic is that a bigger set of objects possibly have a smaller set of common attributes in a relation and this leads to bigger probability to connect this factor with some factor from the second data table, moreover, if we take factor which contains the biggest set of attributes in intervals of the second data table.

Because we do not want to construct the whole concept lattice and search in it, we compute candidates for greatest element directly from relation $\mathcal{R}_{C_1C_2}$. We take all objects belonging to the top element of interval $\mathcal{I}_{ij}$ from the first data table and compute how many of them belong to each attribute in the relation. We take into account only attributes belonging to object i. We take as candidate the greatest set of objects belonging to some attribute in a relation, which satisfies that if we compute a closure of this set in the first data table, resulting set of objects do not have empty set of common attributes in a relation.

Applying this heuristic on data from the example, we obtain three factors in the first data table, $F_1^{C_1} = \langle\{2, 4\}, \{a, c\}\rangle$, $F_2^{C_1} = \langle\{1, 3, 4\}, \{c, d\}\rangle$, $F_3^{C_1} = \langle\{1, 2, 4\}, \{c\}\rangle$ and four factors $F_1^{C_2} = \langle\{5\}, \{e, h\}\rangle$, $F_2^{C_2} = \langle\{6, 7\}, \{f, g\}\rangle$, $F_3^{C_2} = \langle\{7\}, \{e, f, g\}\rangle$, $F_4^{C_2} = \langle\{8\}, \{g, h\}\rangle$ from the second one. Between this factors, there are six connections satisfying the relation. These connections are shown in table 6.

We form multi-relational factors in a greedy manner. In each step we connect factors, which cover the biggest part of still uncovered part of data tables C_1 and

Table 6: Connections between factors

	$F_1^{C_2}$	$F_2^{C_2}$	$F_3^{C_2}$	$F_4^{C_2}$
$F_1^{C_1}$			×	
$F_2^{C_1}$		×	×	
$F_3^{C_1}$		×	×	×

C_2. Firstly, we obtain multi-relational factor $\langle F_2^{C_1}, F_2^{C_2} \rangle$ which covers 50 percent of the data. Then we obtain factor $\langle F_3^{C_1}, F_4^{C_2} \rangle$ which covers together with first factor 75 percent of the data and last we obtain factor $\langle F_1^{C_1}, F_3^{C_2} \rangle$. All these factors cover 90 percent of the data. By adding other factors we do not obtain better coverage of input data. These three factors cover the same part of input data as six connections from table 6.

Remark 1. As we mentioned above and what we can see in the example, multi-relational factors are not always able to explain the whole data. This is due to nature of data. Simply there is no information how to connect some classic factors, e.g. in the example no set of objects from C_1 has in $\mathcal{R}_{C_1C_2}$ a set of common attributes equal to $\{e, h\}$ (or only $\{e\}$ or only $\{h\}$). From this reason we are not able to connect any factor from C_1 with factor $F_1^{C_2}$.

Remark 2. In previous part we explain the idea of the algorithm on a object-attribute relation between data tables. It is also possible consider different kind of relation, e.g. object-object, attribute-object or attribute-attribute relation. Without loss of generality we present the algorithm only for the object-attribute relation. Modification to a different kind of relation is very simple.

Now we are going to describe the pseudo-code (Algorithm 1) of our algorithm for MBMF. Input to this algorithm are two Boolean data tables C_1 and C_2, binary relation $\mathcal{R}_{C_1C_2}$ between them and a number $p \in [0, 1]$ which represent how large part of C_1 and C_2 we want to cover by multi-relational factors, e.g. value 0.9 mean that we want to cover 90 percent of entries in input data tables. Output of this algorithm is a set $\mathcal{M}$ of multi-relational factors that covers the prescribed portion of input data (if it is possible to obtain prescribed coverage). The first computed factor covers the biggest part of data.

First, in lines 1-2 we compute essential part of C_1 and C_2. In lines 2-4 we initialize variables U_{C_1} and U_{C_2}. These variables are used for storing information about still uncovered part of input data. We repeat the main loop (lines 5-18) until we obtain a required coverage or until it is possible to add new multi-relational factors which cover still uncovered part (lines 12-14).

In the main loop for each essential element we select the best candidate from interval $\mathcal{I}_{ij}$ from the first data table in the greedy manner described in the algorithm idea, i.e. we take the greatest concept which can be connected via relation. Than we try to connect this candidate with factors from the second data table. We compute cover function and we add to $\mathcal{M}$ the multi-relational factor maximizing this coverage.

In lines 16-17 we remove from U_{C_1} and U_{C_2} entries which are covered by actually added multi-relational factor.

Algorithm 1: Algorithm for the multi-relational Boolean factors analysis

Input: Boolean matrices C_1, C_2 and relation $R_{C_1C_2}$ between them and $p \in [0,1]$
Output: set $\mathcal{M}$ of multi-relational factors

1 $E_{C_1} \leftarrow Ess(C_1)$
2 $E_{C_2} \leftarrow Ess(C_2)$
3 $U_{C_1} \leftarrow C_1$
4 $U_{C_2} \leftarrow C_2$
5 **while** $(|U_{C_1}| + |U_{C_2}|)/(|C_1| + |C_2|) \geq p$ **do**
6 **foreach** *essential element* $(E_{C_1})_{ij}$ **do**
7 compute the best candidate $\langle a, b\rangle$ from interval $\mathcal{I}_{ij}$
8 **end**
9 $\langle A, B\rangle \leftarrow$ select one from set of candidates which maximize cover of C_1
10 select non-empty row i in E_{C_2} for which is $A^{\uparrow R_{C_1C_2}} \subseteq (C_2)_{i\_}^{\downarrow\uparrow C_2}$ and which maximize cover of C_1 and C_2
11 $\langle C, D\rangle \leftarrow \langle (C_2)_{i\_}^{\uparrow\downarrow C_2}, (C_2)_{i\_}^{\uparrow C_2}\rangle$
12 **if** *value of cover function for* C_1 *and* C_2 *is equal to zero* **then**
13 **break**
14 **end**
15 **add** $\langle\langle A, B\rangle, \langle C, D\rangle\rangle$ **to** $\mathcal{M}$
16 **set** $(U_{C_1})_{ij} = 0$ where $i \in A$ and $j \in B$
17 **set** $(U_{C_1})_{ij} = 0$ where $i \in C$ and $j \in D$
18 **end**
19 **return** $\mathcal{F}$

Our implementation of the algorithm follows the pseudo-code conceptually, but not in details. For example we speed up the algorithm by precomputing candidates or instead computing candidates for each essential elements, we compute candidates for essential areas, i.e. essential elements which are covered by one formal concept.

Remark 3. The input of our algorithm are two Boolean data tables and one relation between them. In general we can have more data tables and relations. Generalization of our algorithm for such input is possible. Due to lack of space we mentioned only an idea of this generalization. For the input data tables $C_1, C_2, \ldots, C_n$ and relations $\mathcal{R}_{C_iC_{i+1}}, i \in \{1, 2, \ldots, n-1\}$ we firstly compute multi-relational factors for C_{n-1} and C_n. Then iteratively compute multi-relational factors for C_{n-2} and C_{n-1}. From this pairs we construct n-tuple multi-relational factor.

We do not make a detail analysis of the time complexity of the algorithm. Even our slow implementation in MATLAB is fast enough for factorization usually large datasets in a few minutes.

5 Experimental evaluation

For experimental evaluation of our algorithm we use in a data minig community well known real dataset MovieLens[1]. This dataset is composed of two data tables that represent a set of users and their attributes, e.g. gender, age, sex, occupation and a set of movies again with their attributes, e.g. the year of production or genre. Last part of this dataset is a relation between this data sets. This relation contains 1000209 anonymous ratings of approximately 3900 movies (3952) made by 6040 MovieLens users who joined to MovieLens in 2000. Each user has at least 20 ratings. Ratings are made on a 5-star scale (values 1-5, 1 means, that user does not like a movie and 5 means that he likes a movie).

Originally data tables Users and Movies are categorical. Age is grouped into 7 categories such as "Under 18", "18-24", "25-34", "35-44", "45-49", "50-55" and "56+". Sex is from set {Male, Female}. Occupation is chosen from the following choices: "other" or not specified, "academic/educator", "artist", "clerical/admin", "college/grad student", "customer service", "doctor/health care", "executive/managerial", "farmer", "homemaker", "K-12 student", "lawyer", "programmer", "retired", "sales/marketing", "scientist", "self-employed", "technician/engineer", "tradesman/craftsman", "unemployed" and "writer". Film genres are following: "Action", "Adventure", "Animation", "Children's", "Comedy", "Crime", "Documentary", "Drama", "Fantasy", "Film-Noir", "Horror", "Musical", "Mystery", "Romance", "Sci-Fi", "Thriller", "War" and "Western". Year of production is from 1919 to 2000. We grouped years into 8 categories "1919-1930", "1931-1940", "1941-1950", "1951-1960", "1961-1970", "1971-1980", "1981-1990" and "1991-2000".

We convert the ordinal relation in to binary one. We use three different scaling. The first is that user rates a movie. The second is that a user does not like a movie (he rates movie with 1-2 stars). The last one is that user likes a movie (rates 4-5). This does not mean, that users do like (respective do not like) some genre, it means, that movies from this genre are or are not worth to see. We took the middle size version of the MovieLens dataset and we made a restriction to 3000 users and movies that were rated by that users. We take users, who rate movies the most, and we obtain dimension of the first data table 3000×30 and dimension of the second data table is 3671×26. Let us just note that for obtaining object-attribute relation we need to transpose Movies data table.

Relation "user rates a movie" make sense, because user rates a movie if he has seen it. We can understand this relation as user has seen movie. We get 29 multi-relational factors, that cover almost 100% of data (99.97%). Values of coverage, i.e. how large part of input data is covered can be seen in figure 2.

[1] http://grouplens.org/datasets/movielens/

Graphs in figure 3 show coverage of Users data table and Movies data table separately.

We can also see that for explaining more than 90 percent of data are sufficient 17 factors. This is significant reduction of input data.

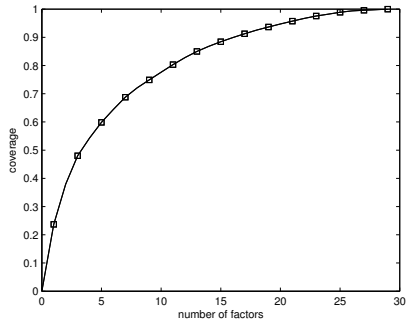
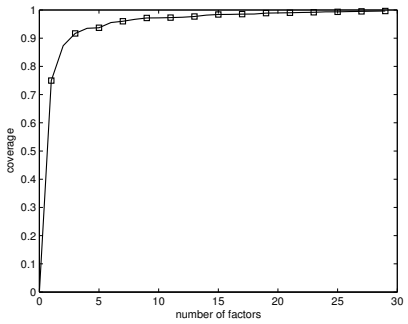
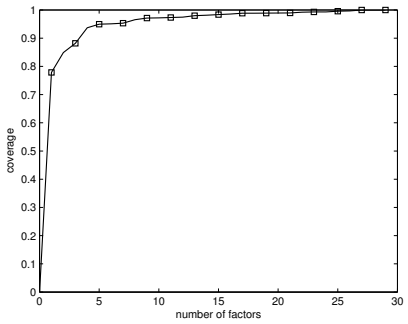


Fig. 2: Cumulative coverage of input data



(a) Coverage of Users data table



(b) Coverage of Movies data table

Fig. 3: Coverage of input data tables

The most important factors are:

- Males rate new movies (movies from 1991 to 2000).
- Young adult users (ages 25-34) rate drama movies.

– Females rate comedy movies.
– Youth users (18-24) rate action movies.

Another interesting factors are:

– Old users (from category 56+) rate movies from their childhood (movies from 1941 to 1950).
– Users in age range 50-55 rate children's movies. Users in this age usually have grand children.
– K-12 students rate animation movies.

Due to lack of space, we skip details about factors in relation "user does not like a movie" and relation "user does like a movie". In the first relation we get 30 factors, that covers 99.99% of data. In the second one, we get 29 factors, covering 99.96% of data. Compute all multi-relational factors on this datasets take approximately 5 minutes.

Remark 4. In case of MovieLens we are able to reconstruct input data tables almost wholly for each three relations. Interesting question is what about relation, i.e. can we reconstruct relation between data tables? Answer is yes, we can. Multi-relational factor carry also information about the relation between data tables. So we can reconstruct it, but with some error. This error is a result of choosing the narrow approach.

Reconstruction error of relation is interesting information and can be minimize if we take this error into account in phase of computing coverage. In other words we want maximal coverage with minimal relation reconstruction error. This leads to more complicated algorithm because we need weights to compute a value of utility function. We implement also this variant of algorithm. Requirement of minimal reconstruction error and maximal coverage seems to be contradictory, but this claim need more detailed study. Also it is necessary to determine correct weight settings. We left this issue for the extended version of this paper.

6 Conclusion and Future Research

In this paper, we present new algorithm for multi-relational Boolean matrix factorization, that uses essential elements from binary matrices for constructing better multi-relational factors, with regard to relations between each data table. We test the algorithm on, in data mining well known, dataset MovieLens. We obtain from these experiments interesting and easy interpretable results, moreover, the number of obtained multi-relational factors needed for explaining almost whole data is reasonable small.

A future research shall include the following topics: generalization of the algorithm for MBMF for ordinal data, especially data over residuated lattices. Construction of algorithm which takes into account reconstruction error of the

relation between data tables. Test the potential of this method in recommendation systems. And last but not least create not crisp operator for connecting classic factors into multi-relational factors.

Acknowledgment
We acknowledge support by IGA of Palacky University, No. PrF 2014 034.

References

1. Belohlavek R., Trnecka M.: From-Below Approximations in Boolean Matrix Factorization: Geometry and New Algorithm. http://arxiv.org/abs/1306.4905, 2013.
2. Belohlavek R., Vychodil V.: Discovery of optimal factors in binary data via a novel method of matrix decomposition. J. Comput. Syst. Sci. 76(1), 3–20, 2010.
3. Džeroski S.: Multi-Relational Data Mining: An Introduction. ACM SIGKDD Explorations Newsletter, 1(5), 1–16, 2003.
4. Ganter B., Wille R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin, 1999.
5. Geerts F., Goethals B., Mielikäinen T.: Tiling databases, Proceedings of Discovery Science 2004, pp. 278–289, 2004.
6. Hacene M. R., Huchard M., Napoli A., Valtechev P.: Relational concept analysis: mining concept lattices from multi-relational data. Ann. Math. Artif. Intell. 67(1), 81–108, 2013.
7. Krmelova M., Trnecka M.: Boolean Factor Analysis of Multi-Relational Data. In: M. Ojeda-Aciego, J. Outrata (Eds.): CLA 2013: Proceedings of the 10th International Conference on Concept Lattices and Their Applications, pp. 187–198, 2013.
8. Lucchese C., Orlando S., Perego R.: Mining top-K patterns from binary datasets in presence of noise. SIAM DM 2010, pp. 165–176, 2010.
9. Miettinen P.: On Finding Joint Subspace Boolean Matrix Factorizations. Proc. SIAM International Conference on Data Mining (SDM2012), pp. 954-965, 2012.
10. Miettinen P., Mielikäinen T., Gionis A., Das G., Mannila H.: The discrete basis problem, IEEE Trans. Knowledge and Data Eng. 20(10), 1348–1362, 2008.
11. Spyropoulou E., De Bie T.: Interesting Multi-relational Patterns. In Proceedings of the 2011 IEEE 11th International Conference on Data Mining, ICDM '11, pp. 675–684, 2011.
12. Xiang Y., Jin R., Fuhry D., Dragan F. F.: Summarizing transactional databases with overlapped hyperrectangles, Data Mining and Knowledge Discovery 23(2), 215–251, 2011.

On Concept Lattices as Information Channels

Francisco J. Valverde-Albacete[1]*, Carmen Peláez-Moreno[2], and Anselmo Peñas[1]

[1] Departamento de Lenguajes y Sistemas Informáticos
Universidad Nacional de Educación a Distancia, c/ Juan del Rosal, 16. 28040 Madrid, Spain {fva,anselmo}@lsi.uned.es
[2] Departamento de Teoría de la Señal y de las Comunicaciones
Universidad Carlos III de Madrid, 28911 Leganés, Spain
carmen@tsc.uc3m.es

Abstract. This paper explores the idea that a concept lattice is an information channel between objects and attributes. For this purpose we study the behaviour of incidences in L-formal contexts where L is the range of an information-theoretic entropy function. Examples of such data abound in machine learning and data mining, e.g. confusion matrices of multi-class classifiers or document-term matrices. We use a well-motivated information-theoretic heuristic, the maximization of mutual information, that in our conclusions provides a flavour of feature selection providing and information-theory explanation of an established practice in Data Mining, Natural Language Processing and Information Retrieval applications, viz. stop-wording and frequency thresholding. We also introduce a post-clustering class identification in the presence of confusions and a flavour of term selection for a multi-label document classification task.

1 Introduction

Information Theory (IT) was born as a theory to improve the efficiency of (man-made) communication channels [1, 2], but it soon found wider application [3]. This paper is about using the model of a communication channel in IT to explore the formal contexts and concept lattices of Formal Concept Analysis as realisations of information channels between objects and attributes. Given the highly unspecified nature of both the latter abstractions such a model will bring new insights into a number of problems, but we are specifically aiming at machine learning and data mining applications [4, 5].

The *metaphor* of a concept lattice as a communication channel between objects and attributes is already implicit in [6, 7]. In there, adjoint sublattices were already considered as subchannels in charge of transmitting individual acoustical features, and some efforts were done to model such features explicitly [7],

* FJVA and AP are supported by EU FP7 project LiMoSINe (contract 288024) for this work. CPM has been supported by the Spanish Government-Comisión Interministerial de Ciencia y Tecnología project TEC2011-26807.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

but no conclusive results were achieved. The difficulty rose from a thresholding parameter φ that controls the lattice-inducing technique and was originally fixed by interactive exploration, a procedure hard to relate to the optimization of a utility or cost function, as required in modern machine learning.

In this paper we set this problem against the backdrop of direct mutual information maximization—using techniques and insights developed since [6, 7]—for matrices whose entries are frequency counts. These counts appear frequently in statistics, data mining and machine learning, for instance, in the form of document-term matrices in Information Retrieval [8], confusion matrices for classifiers in perceptual studies, data mining and machine learning [9], or simply two-mode contingency tables with count entries. Such matrices are called *aggregable* in [4], in the sense that any group of rows or columns can be aggregated together to form another matrix whose frequencies are obtained from the data of the elements in the groups. We will use this feature to easily build count and probability distributions whose mutual information can be maximized, following the heuristic motivated above, to improve classification tasks. Note that maximizing mutual information (over all possible joint distributions) is intimately related to the concept of *channel capacity* as defined by Shannon [2].

For this purpose, in Sec. 2 we cast the problem of analysing the transfer of information through the two modes of contingency tables as that of analysing a particular type of formal context. First we present in Sec. 2.1 the model of the task to be solved, then we present aggregable data, as usually found in machine learning applications in Sec. 2.2, and then introduce the entropic encoding to make it amenable to FCA. As an application, in Sec. 3.1 we explore the particular problem of *supervised clustering* as that of transferring the labels from a set of input patterns to the labels of the output classes. Specifically we address the problem of assigning labels to mixed clusters given the distribution of the input labels in them. We end with a discussion and a summary of contributions and conclusions.

2 Theory

2.1 Classification optimization by mutual information maximization

Consider the following, standard supervised classification setting: we have two domains X and Y, m instances of i.i.d. samples $S = \{(x_i, y_i)\}_{i=1}^{m} \subseteq X \times Y$, and we want to learn a function $h : X \to Y$, the *hypothesis*, with certain "good" qualities, to estimate the *class* Y from X , the measurements of Y , or *features*.

A very productive model to solve this problem is to consider two probability spaces $\mathcal{Y} = \langle Y, P_Y \rangle$ and $\mathcal{X} = \langle X, P_X \rangle$ with $Y \sim P_Y$ and $X \sim P_X$, and suppose that there exists the product space $\langle X \times Y, P_{XY} \rangle$ wherefrom the i.i.d. samples of S have been obtained. So our problem is solved by estimating the random variable $\hat{Y} = h(X)$, and a "good" estimation is that which obtains a low error probability on every possible pair $P(\hat{Y} \neq Y) \to 0$.

Since working with probabilities might be difficult, we might prefer to use a (surrogate) loss function that quantifies the cost of this difference $\mathcal{L}(\hat{y} =$

$h(x), y)$ and try to minimize the expectation of this loss, called the risk $R(h) = E[\mathcal{L}(h(x), y)]$ over a class of functions $h \in \mathcal{H}$, $h^* = \min_{h \in \mathcal{H}} R(h)$. Consequently, this process is called *empirical risk minimization*.

An alternate criterion is to maximize the mutual information between Y and $\hat{Y}$ [10]. This is clearly seen from Fano's inequality [11], serving as a lower bound, and the Hellman-Raviv upper bound [12],

$$\frac{H_{P_{\hat{Y}}} - I_{P_{Y\hat{Y}}} - 1}{H_{U_{\hat{Y}}}} \leq P(\hat{Y} \neq Y) \leq \frac{1}{2} H_{P_{\hat{Y}|Y}}$$

where $U_{\hat{Y}}$ is the uniform distribution on the support of $\hat{Y}$, $H_{P_{XX}}$ denotes the different entropies involved and $I_{P_{Y\hat{Y}}}$ is the mutual information of the joint probability distribution.

2.2 Processing aggregable data

If the original rows and columns of contingency tables represent atomic events, their groupings represent complex events and this structure is compatible with the underlying sigma algebras that would transform the matrix into a joint distribution of probabilities, hence these data can be also interpreted as joint probabilities, when row- and column-normalized.

When insufficient data is available for counting, the estimation of empirical probabilities from this kind of data is problematic, and complex probability estimation schemes have to be used. Even if data galore were available, we still have to deal with the problem of rarely seen events and their difficult probability estimation. However, probabilities are, perhaps, the best data that we can plug onto data mining or machine learning techniques, be they for supervised or unsupervised tasks.

The weighted Pointwise Mutual Information. Recall the formula for the mutual information between two random variables $I_{P_{XY}} = \mathbf{E}_{P_{XY}}[I_{XY}(x, y)]$ where $I_{XY}(x, y) = \log \frac{P_{XY}(x,y)}{P_X(x) \cdot P_Y(y)}$ is the *pointwise mutual information, (PMI).*

Remember that $-\infty \leq I_{XY}(x, y) < \infty$ with $I_{XY}(x, y) = 0$ being the case where X and Y are independent. The negative values are caused by phenomena less represented in the joint data than in independent pairs as captured by the marginals. The extreme value $I_{XY}(x, y) = -\infty$ is generated when the joint probability is negative even if the marginals are not. These are instances that capture "negative" association whence to maximize the expectation we might consider disposing of them.

On the other hand, on count data the PMI has an unexpected and unwanted effect: it is very high for *hapax legomena* phenomena that are encountered only once in a tallying, and in general it has a high value for phenomena with low counts of whose statistical behaviour we are less certain.

However, we know that

$$I_{P_{XY}} = \sum_{x,y} P_{XY}(x,y) \cdot I_{XY}(x,y) = \sum_{x,y} P_{XY}(x,y) \log \frac{P_{XY}(x,y)}{P_X(x) \cdot P_Y(y)}$$

and this is *always a positive quantity, regardless of the individual values of* $I_{XY}(x,y)$. This suggests calling *weighted pointwise mutual information, (wPMI)* the quantity

$$\text{wPMI}(x,y) = P_{XY}(x,y) \log \frac{P_{XY}(x,y)}{P_X(x) \cdot P_Y(y)} \tag{1}$$

and using it as the subject of optimization or exploration to do so. Note that pairs of phenomena whose joint probability are close to independent, as judged by the pointwise information, will be given a very low value in the $wPMI$, and that the deleterious character of hapaxes on $I_{P_{XY}}$ is lessened by the influence of the joint probability.

2.3 Visualizing mutual information maximization

For a joint distribution $P_{Y\hat{Y}}(y,\hat{y})$, [13] introduced a balance equation binding the mutual information between two variables $I_{P_{Y\hat{Y}}}$, the sum of their conditional entropies $VI_{P_{Y\hat{Y}}} = H_{P_{Y|\hat{Y}}} + H_{P_{\hat{Y}|Y}}$ and the sum of their entropic distance between their distributions and uniformity $\Delta H_{P_{Y\hat{Y}}} = (H_{U_Y} - H_{P_Y}) + (H_{U_{\hat{Y}}} - H_{P_{\hat{Y}}})$,

$$\log(H_{U_Y}) + \log(H_{U_{\hat{Y}}}) = \Delta H_{P_{Y\hat{Y}}} + 2 * I_{P_{Y\hat{Y}}} + VI_{P_{Y\hat{Y}}} \ .$$

By normalizing in the total entropy $\log(H_{U_Y}) + \log(H_{U_{\hat{Y}}})$ we may obtain the equation of the 2-simplex that can be represented as a De Finetti diagram like that of Fig. 2.(a), as the point in the 2-simplex corresponding to coordinates

$$F(P_{Y\hat{Y}}) = [\Delta H'_{P_{Y\hat{Y}}}, 2 * I'_{P_{Y\hat{Y}}}, VI'_{P_{Y\hat{Y}}}]$$

where the primes represent the normalization described above.

The axis of this representation were chosen so that the height of the 2-simples—an equilateral triangle—is proportional to the mutual information between the variables so a maximization process is extremely easy to represent (as in Fig. 2): given a parameter φ whereby to maximize $I_{P_{Y\hat{Y}}}$ (as a variable), draw the trace of the evaluation of the coordinates in the ET of the distributions that it generates, and choose the φ^* that produces the highest point in the triangle. This technique is used in Sec. 3.1, but other intuitions can be gained from this representation as described in [14].

2.4 Exploring the space of joint distributions

Since the space of count distributions is so vast, we need a technique to explore it in a principled way. For that purpose we use $\mathcal{K}$-Formal Concept Analysis

(KFCA). This is a technique to explore L-valued contexts where L is a complete idempotent semifield using a free parameter called the threshold of existence [15, 13].

We proceed in a similar manner to Fuzzy FCA: For L-context $\langle Y, \hat{Y}, R\rangle$, consider two spaces L^Y and $L^{\hat{Y}}$, representing, respectively, L-valued sets of objects and attributes. Pairs of such sets of objects and attributes that fulfil certain polars equation have been proven to define dually-ordered lattices of closed L-sets in the manner of FCA [3].

Since the actual lattices of object sets and attributes are so vast, KFCA uses a simplified representation for them: for the singleton sets in each of the spaces δ_y, for $y \in Y$ and $\delta_{\hat{y}}$, for $\hat{y} \in \hat{Y}$, we use the L-polars to generate their object- $\gamma^{\varphi}_{Y}(y)$ and attribute-concept $\mu^{\varphi}_{\hat{Y}}(\hat{y})$, respectively, and obtain a *structural φ-context* $\mathbb{K}^{\varphi} = \langle Y, \hat{Y}, R^{\varphi}\rangle$, where $yR^{\varphi}\hat{y} \iff \gamma^{\varphi}_{Y}(y) \leq \mu^{\varphi}_{\hat{Y}}(\hat{y})$ [4].

In this particular case we consider the min-plus idempotent semifield and the L-context $\langle Y, \hat{Y}, wPMI\rangle$ where wPMI is the weighted Pointwise Mutual Information relation between atomic events in the sigma lattices of Y and $\hat{Y}$ of Sec. 2.2, whence the degree or threshold of existence is *a certain amount of entropy required for concepts to surpass* for them to be considered.

The following step amounts to an *entropy conformation* of the joint distribution, that is, a redistribution of the probability masses in the joint distribution to obtain certain entropic properties. Specifically, we use the (binary) φ-formal context to filter out certain counts in the contingency table to obtain a *conformal contingency table* $N^{\varphi}_{Y\hat{Y}}(y, \hat{y}) = N_{Y\hat{Y}}(y, \hat{y}) \odot \mathbb{K}^{\varphi}$, where $\odot$ represents here the Hadamard (pointwise) product. For each conformal $N^{\varphi}_{Y\hat{Y}}(y, \hat{y})$ we will obtain a certain point $F(\varphi)$ in the ET to be represented as described in Sec. 2.3.

3 Application

We next present two envisaged applications of the technique of MI Maximization.

3.1 Cluster identification

Confusion matrices are special contingency tables whose two modes refer to the same underlying set of labels[4]. We now put forward a procedure to maximize the information transmitted from a set of "ground truth" patterns acting as objects with respect to "perceived patterns" which act as attributes. As noted in the introduction, this is just one of the possible points of view about this problem.

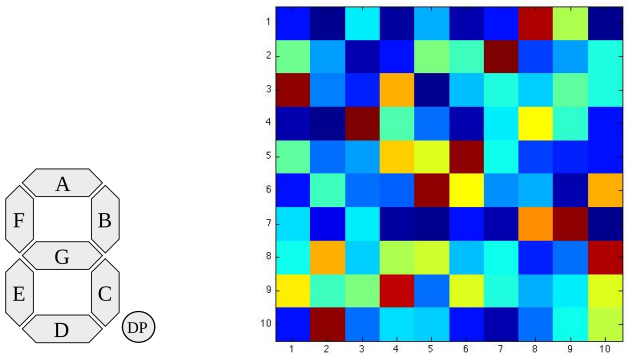
Consider the following scenario, there is a clustering task for which extrinsic evaluation is possible, that is, there is a gold standard partitioning of the input data. One way to evaluate the clustering solution is to obtain a confusion

[3] Refer to [13] for an in-depth discussion of the mathematics of idempotent semifields and the different kinds of Galois connections that they generate.

[4] And a *structural φ-lattice* $\underline{\mathfrak{B}}^{\varphi}(\mathbb{K}^{\varphi})$ as its concept lattice, but this is not important in the present application

matrix out of this gold standard, in the following way: If the number of classes is known—a realistic assumption in the presence of a gold standard—then the MI optimization procedure can be used to obtain the assignments between the classes in the gold standard and the clusters of the procedure, resulting in cluster identification.

For the purpose of testing the procedure, we used the segmented numeral data from [16]. This is a task of human visual confusions between numbers as displayed by seven-segment LED displays, as shown in Fig. 1.(a). The entry in the count matrix $N_{CK}(c,k) = n_{ck}$ counts the number times that an instance of class c was confused with class k. Figure 1.(b) shows a heatmap presentation of the original confusion matrix and column-reshuffled variants. Note that the confusion matrix is diagonally-dominant, that is $n_{ii} > \sum_{j,j\neq i} n_{ij}$ and likewise for column i .



(a) (b)

Fig. 1: Segmented numeral display (a) from [16] and the column-reshuffled confusion matrix (b) of the human-perception experiment. Cluster identification is already evident in this human-visualization aid, but the method here presented is unsupervised.

To test the MI optimization procedure, we randomly permuted the confusion matrix columns: the objective was to recover the inverse of this random permutation from the MI optimization process so that the original order could be restored. This amounts to an assignment between classes and induced clusters, and we claim that it can be done by means of the mutual information maximization procedure sketched above.

For that purpose, we estimated $P_{CK}(c,k)$ using the empirical estimate

$$\hat{P}_{CK}(c,k) \approx \frac{N_{CK}(c,k)}{n}$$

where n is the number of instances to be clustered $n = \sum_{ck} N_{CK}(c,k)$, and then we obtained its empirical PMI

$$\hat{I}_{CK}(c,k) = \log \hat{P}_{CK}(c,k)$$

and its weighted PMI

$$wPMI_{CK}(c,k) = \hat{P}_{CK}(c,k) \cdot \hat{I}_{CK}(c,k) \ .$$

Next, we used the procedure of Sec. 2.4 to explore the empirical wPMI and select the threshold value which maximizes the MI. Figure 2.(a) shows the trajectory of the different conformed confusion matrices as φ ranges in $[0,\infty)$ on the ET: we clearly see how for this balanced task dataset the exploration results in a monotonous increase in MI in the thresholding range until a value that produces the maximum MI, at $wPMI^* = 0.1366$. The discrete set of points stems from the limited range of counts in the data.

We chose this value as threshold and obtained the binary matrix which is the assignment from classes to clusters and vice-versa shown in Fig. 2.(b). Note that in this particular instance, the φ^*-concept lattice is just a diamond lattice reflecting the perfect identification of classes and clusters. In general, with contingency tables where modes have different cardinalities, this will not be the case.

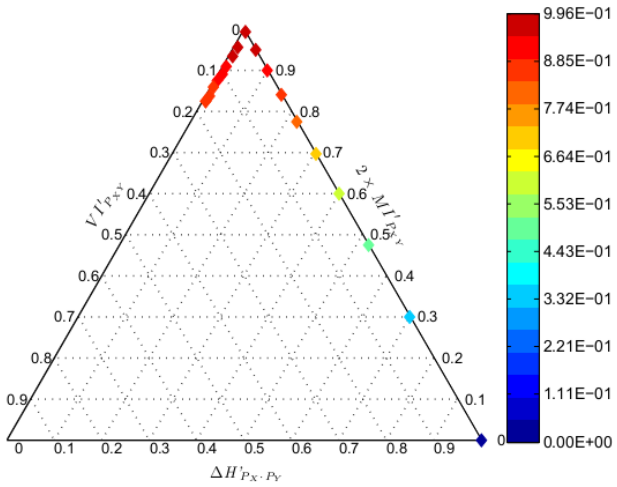
3.2 Entropy conformation for count matrices

The case where the contingency matrix is squared and diagonally dominant, as in the previous example, is too specific: we need to show that for a generic, rectangular count contingency matrix, entropy maximization is feasible and meaningful.

The first investigation should be on how to carry the maximization process. For that purpose, we use a modified version of the Reuters-21578 [5] that has already been stop-listed and stemmed. This is a multi-label classification dataset [17] describing each document as a bag-of-terms and some categorizations labels, the latter unused in our present discussion.

We considered the document-term matrix for training, a count distribution with $D = 7\,770$ documents and $T = 5\,180$, terms. Its non-conformed entropy coordinates are $F(N_{DT}) = [0.1070, 0.3584, 0.5346]$ as shown in the deep blue circle to the left of Fig. 3. We carried out a joint-mutual information maximization process by exploring at the same time a max-plus threshold—the count has to be bigger thant the threshold to be considered—and a min-plus threshold—the count has to be less than the threshold. The rationale for this is a well-tested hypothesis in the bag-of-term model: very common terms (high frequency) do not select well for documents, while very scarce terms (low frequency) are too specific and biased to denote the general "aboutness" of a document. Both should be filtered out of the document-term matrix.

[5] `http://www.daviddlewis.com/resources/testcollections/reuters21578/readme.txt`. Visited 24/06/2014.



(a)

$$\mathbb{K}^{\varphi *} = \begin{bmatrix} 0&0&0&0&1&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0 \end{bmatrix}$$

(b)

Fig. 2: Trajectory of the evolution of MI transmission for the segmented numeral data as the exploration threshold is raised in the wPMI matrix (a), and maximal MI cluster assignment matrix at wPMI = 1.366 bits (b) for column-shuffled Segmented Numerals. The resulting concept lattice is just a diamond lattice identifying classes and clusters and not shown.

Instead of count-based individual term filtering we carry a joint term-document pair selection process: for a document-matrix, we calculate its overall weighted PMI matrix, and only those pairs (d, t) whose wPMI lies in between a lower ϕ and an upper φ thresholds are considered important for later processing. For each such pairs, we created an indicator matrix $I(d, t)$ that is 1 iff $\phi \leq wMI(d, t) \leq \varphi$, and we used the Kronecker multiplication to filter out non-conforming pairs from the final entropy calculation,

$$\hat{MI}'_{P_{DT}} = \sum_{d,t} wPMI_{DT}(d, t) \cdot I(d, t)$$

Figure 3 represents the trace of that process as we explore a grid of 10×10 different values of ϕ and φ (the same set of values for both). The grid was obtained by equal width binning of the whole range of $wPMI_{DT}(d, t)$ in the original wMI matrix as defined in [18].

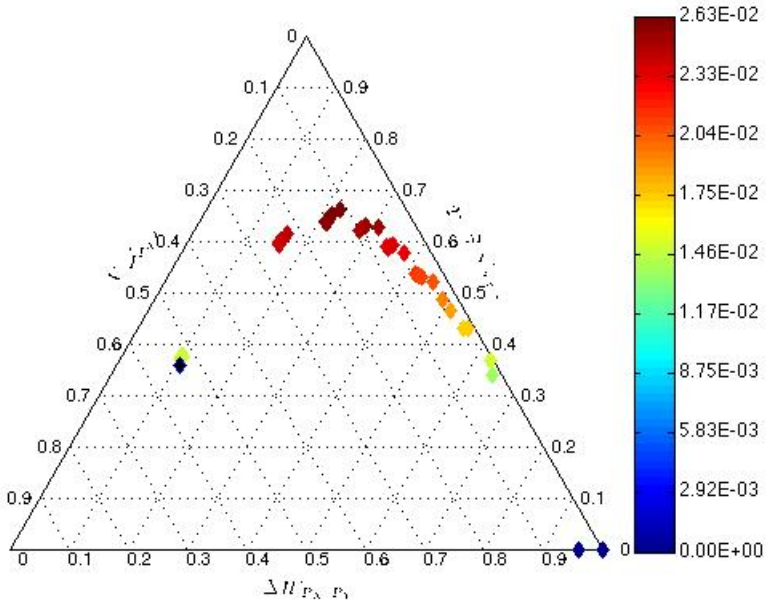


Fig. 3: Trace of the entropy conformation process for a count matrix. The blue dot to the left is the original level of entropy. For a wide range of pairs (ϕ, φ) the entropy of the conformed count matrix is greater than the original one, and we can actually find a value where it is maximized.

We can see how $\hat{MI}'_{P_{DT}}$ reaches a maximum over two values and then decreases again, going even below the original mutual information value. We read

two different facts in this illustration: that the grid used is effective in obtaining approximations to ϕ and φ for MI maximization, and that not every possible pair of values is a good solution for the process.

All in all, this procedure shows that MI maximization is feasible by tracking its in the ET. We do not present any results in this paper as to the effectiveness of the process for further processing tasks, which should be evaluated on the extrinsic measures on the Reuters multi-labelling task.

4 Discussion

We now discuss the applications selected in a wider context. Although less pervasive than its unsupervised version, the basic task of supervised clustering has application, for instance, in tree-induction for supervised classification [5, 18] or unsupervised clustering evaluation using a gold-set [19]. Cluster identification in Sec. 3.1 is a sometimes-fussy sub-procedure in clustering which our proposal solves elegantly.

The feasibility study on mutual information conformation of Sec. 3.2 is a necessary step for further processing—binary or multi-labelling classification—but as of this paper unevaluated. Further work should concentrate on leveraging the boost in mutual information to lower the classification error, as suggested in the theoretical sections.

Besides, the use of two simultaneous, thresholds on different algebras makes it difficult to justify the procedure on FCA terms: this does not conform to the definition of any lattice-inducing polars that we know of, so this feature should be looked into critically. Despite this fact, the procedure of conformation "makes sense", at least for this textual classification task.

Note that the concept of "information channel" that we have developed in this paper is *not* what Communication Theory usually considers. In there, "input symbols" enter the channel and come out as "output symbols", hence input has a sort of ontological primacy over output symbols in that the former *cause* the latter. If there is anything particular about FCA as an epistemological theory is that *it does not prejudge the ontological primacy of objects over attributes or* vice versa. Perhaps the better notion is that a formal concept is an *information co-channel* between objects and attributes, in the sense that the information "flows" both from objects to attributes and vice versa, as per the true symmetric nature of mutual information: receiving information about one of the modes decreases the uncertainty of the other.

The previous paragraph notwithstanding, we will often find ourselves in application scenarios in which one of the modes will be primary with respect to the other, in which case the analogies with communication models will be more evident. This is one of the cases that we explore in this paper, and that first pointed at in [6, 7].

Contingency tables are an instance of *aggregable* data tables [4, §0.3.4]. It seems clear that not just counts, but any non-negative entry aggregable table can be treated with the tools here presented, e.g. concentrations of solutes. In that

case, the neat interpretation related to MI maximization will not be available, but analogue ones can be found.

A tangential approach to the definition of entropies in (non-Boolean) lattices has been taken by [20, 21, 22, 23, 24]. These works approach the definition of measures, and in particular entropy measures, in general lattices instead of finite sigma algebras (that is, Boolean lattices). [22] and [24] specifically address the issue of defining them in concept lattices, but the rest provide other heuristic foundations for the definition of such measures which surely must do without some of the more familiar properties of the Shannon (probability-based) entropy.

5 Conclusions and further work

We have presented an incipient model of L-formal contexts of aggregable data and their related concept lattices as information channels. Using KFCA as the exploration technique and the Entropy Triangle as the representation and visualization technique we can follow the maximization procedure on confusion matrices in general, and in confusion matrices for cluster identification in particular.

We present both the basic theory and two proof-of-concept applications in this respect: a first one cluster identification, fully interpretable in the framework of concept lattices, and another, entropy conformation for rectangular matrices more difficultly embeddable in this framework.

Future applications will extend the analysis of count contingency tables, like document-term matrices, where our entropy-conformation can be likened to feature selection techniques.

Bibliography

[1] Shannon, C.E.: A mathematical theory of Communication. The Bell System Technical Journal **XXVII** (1948) 379–423

[2] Shannon, C., Weaver, W.: A mathematical model of communication. The University of Illinois Press (1949)

[3] Brillouin, L.: Science and Information Theory. Second Edition. Courier Dover Publications (1962)

[4] Mirkin, B.: Mathematical Classification and Clustering. Volume 11 of Nonconvex Optimization and Its Applications. Kluwer Academic Publishers (1996)

[5] Mirkin, B.: Core Concepts in Data Analysis: Summarization, Correlation and Visualization. Summarization, Correlation and Visualization. Springer, London (2011)

[6] Peláez-Moreno, C., García-Moral, A.I., Valverde-Albacete, F.J.: Analyzing phonetic confusions using Formal Concept Analysis. Journal of the Acoustical Society of America **128** (2010) 1377–1390

[7] Peláez-Moreno, C., Valverde-Albacete, F.J.: Detecting features from confusion matrices using generalized formal concept analysis. In Corchado, E., Grana-Romay, M., Savio, A.M., eds.: Hybrid Artificial Intelligence Systems. 5th International Conference, HAIS 2010, San Sebastián, Spain, June 23-25, 2010. Proceedings, Part II. Volume 6077 of LNAI., Springer (2010) 375–382
[8] Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
[9] Japkowicz, N., Shah, M.: Evaluating Learning Algorithms: A Classification Perspective. Cambridge University Press (2011)
[10] Frénay, B., Doquire, G., Verleysen, M.: Theoretical and empirical study on the potential inadequacy of mutual information for feature selection in classification. NEUROCOMPUTING **112** (2013) 64–78
[11] M Fano, R.: Transmission of Information: A Statistical Theory of Communication. The MIT Press (1961)
[12] Feder, M., Merhav, N.: Relations between entropy and error probability. IEEE Transactions on Information Theory **40** (1994) 259–266
[13] Valverde-Albacete, F.J., Peláez-Moreno, C.: Two information-theoretic tools to assess the performance of multi-class classifiers. Pattern Recognition Letters **31** (2010) 1665–1671
[14] Valverde-Albacete, F.J., Peláez-Moreno, C.: 100% classification accuracy considered harmful: the normalized information transfer factor explains the accuracy paradox. PLOS ONE (2014)
[15] Valverde-Albacete, F.J., Peláez-Moreno, C.: Galois connections between semimodules and applications in data mining. In Kusnetzov, S., Schmidt, S., eds.: Formal Concept Analysis. Proceedings of the 5th International Conference on Formal Concept Analysis, ICFCA 2007, Clermont-Ferrand, France. Volume 4390 of LNAI., Springer (2007) 181–196
[16] Keren, G., Baggen, S.: Recognition models of alphanumeric characters. PERCEPT PSYCHOPHYS **29** (1981) 234–246
[17] Tsoumakas, G., Katakis, I.: Multi-label classification: An overview. International Journal of Data Warehousing and ... (2007)
[18] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. SIGKDD Explorations **11** (2009)
[19] Meila, M.: Comparing clusterings—an information based distance. Journal of Multivariate Analysis **28** (2007) 875–893
[20] Knuth, K.: Valuations on Lattices and their Application to Information Theory. Fuzzy Systems, IEEE International Conference on (2006) 217–224
[21] Grabisch, M.: Belief functions on lattices. International Journal Of Intelligent Systems **24** (2009) 76–95
[22] Kwuida, L., Schmidt, S.E.: Valuations and closure operators on finite lattices. Discrete Applied Mathematics **159** (2011) 990–1001
[23] Simovici, D.: Entropies on Bounded Lattices. Multiple-Valued Logic (ISMVL), 2011 41st IEEE International Symposium on (2011) 307–312
[24] Simovici, D.A., Fomenky, P., Kunz, W.: Polarities, axiallities and marketability of items. In: Proceedings of Data Warehousing and Knowledge Discovery - DaWaK. Volume 7448 of LNCS. Springer (2012) 243–252

Using Closed Itemsets for Implicit User Authentication in Web Browsing

O. Coupelon[1], D. Dia[1,2], F. Labernia[2], Y. Loiseau[2], and O. Raynaud[2]

[1] Almerys, 46 rue du Ressort, 63967 Clermont-Ferrand
{olivier.coupelon,diye.dia}@almerys.com
[2] Blaise Pascal University, 24 Avenue des Landais, 63170 Aubière
{loiseau,raynaud}@isima.fr, fabien.labernia@gmail.com

Abstract. Faced with both identity theft and the theft of means of authentication, users of digital services are starting to look rather suspiciously at online systems. The behavior is made up of a series of observable actions of an Internet user and, taken as a whole, the most frequent of these actions amount to habit. Habit and reputation offer ways of recognizing the user. The introduction of an implicit means of authentication based upon the user's behavior allows web sites and businesses to rationalize the risks they take when authorizing access to critical functionalities. In this paper, we propose a new model for implicit authentication of web users based on extraction of closed patterns. On a data set of web navigation connection logs of 3,000 users over a six-month period we follow the experimental protocol described in [1] to compute performance of our model.

1 Introduction

In order to achieve productivity gains, companies are encouraging their customers to access their services via the Internet. It is accepted that on-line services are more immediate and more user-friendly than accessing these services via a brick and mortar agency, which involves going there and, more often than not, waiting around [2]. Nevertheless, access to these services does pose security problems. Certain services provide access to sensitive data such as banking data, for which it is absolutely essential to authenticate the users concerned. However identity thefts are becoming more and more numerous [3]. We can distinguish two paradigms for increasing access security. The first one consists of making access protocols stronger by relying, for example, on external devices for transmitting access codes that are supplementary to the login/password pair. Nevertheless, these processes are detrimental to the user-friendliness and usability of the services. The number of transactions abandoned before reaching the end of the process is increasing and exchange volumes are decreasing. The second paradigm consists to the contrary of simplifying the identification processes in order to increase the exchange volumes. By way of examples, we can mention single-click payment [2] [4] or using RFID chips for contactless payments. Where these two paradigms meet is where we find implicit means of authentication.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

A means of authentication is a process that makes it possible to ensure that the identity declared in the event of access is indeed the user's identity. Traditionally, a user authenticates himself or herself by providing proof of identity [5]. This process is called "explicit authentication". In contrast, implicit authentication does not require anything from the user but instead studies his or her behavior, the trail left by the user's actions, and then either does or does not validate the declared identity. An implicit means of authentication cannot replace traditional means of authentication as it is necessary for the user to have access to his or her service so that the person's behavior may be studied and their identity can either be validated or rejected. To the contrary, if it is effective, it would enable stronger authentication modes to be avoided (such as chip cards and PIN numbers), which are detrimental to the usability of services. The challenge is to detect identity theft as quickly as possible and, to the contrary, to validate a legitimate identity for as long a time as possible.

This contribution is organized as follows: in section 2 we shall offer a state-of-the-art about implicit authentication and user's profile in web browsing. Then we propose a learning model for implicit authentication of web users we are dealing with in section 3. In section 4, we compare several methods for building profiles of each user. We faithfully reproduce the experimental study conducted in [1] and we analyze all of our results. Finally, in section 5, we shall resume our results and discuss our future work.

2 Related works

In his survey of implicit authentication for mobile devices ([6]), the author says of an authentication system that it is *implicit* if the system does not make demands of the user (see Table 1).

Implicit authentication systems were studied very quickly for mobile phones. In [7] and [8], the authors studied behaviour based on variables specific to smartphones such as calls, SMS's, browsing between applications, location, and the time of day. Experiments were conducted based on the data for 50 users over a period of 12 days. The data were gathered using an application installed by users who were volunteers. The users' profiles were built up from how frequently positive or negative events occurred and the location. Within this context, a positive event is an event consistent with the information gathered upstream. By way of an example, calling a number which is in the phone's directory is a positive event. The results of this study show that based on ten or so actions, you can detect fraudulent use of a smartphone with an accuracy of 95%. In a quite different context, the authors of [9] relied on a Bayesian classification in order to associate a behaviour class with each video streaming user. The data set is simulated and consists of 1,000 users over 100 days. The variables taken into account are the quality of the flow, the type of program, the duration of the session, the type of user, and the popularity of the video. The results are mixed, because the model proposed admits to an accuracy rate of 50%.

Feature	*Capturing Method*	*Implicit/Explicit*	*Spoofing Threats*	*Problems*
Passcode	Keyboard input	Explicit	Keyloggers, Shoulder Surfing	Guessable passwords
Token	Hardware device	Mainly explicit, implicit possible	None	Easily stolen or lost
Face & Iris	Camera	Both	Picture of the legitimate user	Lighting situation and make-up
Keystroke	Keyboard	Implicit, explicit possible	Typing imitation (difficult)	Long training phase, reliability
Location	GPS, infrastructure	Implicit	Informed strangers	Traveling, precision
Network	Software protocol (e.g. WireShark)	Implicit	Informed strangers	Precision

Table 1. Comparison of different authentication methods

The particular context of implicit authentication for web browsing was studied in [1], [10], [11] and [12]. In [1], the author adopted the domain name, the number of pages viewed, the session start time, and its duration, as characteristic variables. The data set, which was gathered by a service provider, consisted of 300 first connections by 2,798 users over a period of 12 months. The user profiles consisted of patterns with a size of 1. The author compares several pattern selection approaches like the support and the lift approaches. The study shows that for small, anonymous behavioural patterns (involving up to twenty or so sites visited), the most effective models are still traditional classification models like decision trees. On the other hand, whenever anonymous behaviour exceeds 70 or so sites, the support and lift-based classification models are more accurate. The study conducted in [12] states that the size of the data set remains a determining parameter. Their study, conducted on 10 users over a one-month period, did not enable them to build a significant model for distinguishing users. The authors also concluded that no variable taken individually enables a user to be authenticated. Drawing inspiration from a study conducted in [1], the authors of [13] studied several techniques for spying on a user who holds a dynamic IP address, based on behavioural models. The methods compared are seeking motives, the nearest neighbours technique, and the multinomial Bayesian classifier. The data set consisted of DNS requests from 3,600 users over a two-month period. In this study, only the most significant variables and the most popular host names were considered. The accuracy rates for the models proposed were satisfactory.

The study that we conduct in this paper also forms part of a continuation of the work by [1]. We faithfully reproduce his experimental protocol on our data and we compare performance of our classification algorithm to his specific models.

3 Models

We propose an intuitive learning model architecture for user authentication. From a data set of web browsing logs we compute a set of *own patterns* for each user. A pattern is a set of frequently visited sites. The size of pattern may vary. Thanks to these *profiles* we are able to provide an authentication for anonymous sessions. We then compute confusion matrices and we provide precisions of the models. In our present study, we compare performance of a naive Bayes classifier to variations on k-nearest neighbors algorithms. More precisely, the studied parameters are selection process of user *own patterns*, computation process of *user profiles* and distance functions computed for classification stage. Figure 1 outlines the framework of the machine learning process.

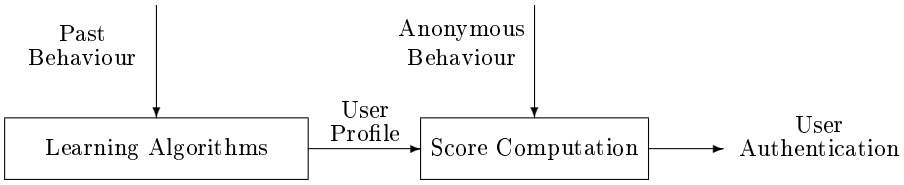


Fig. 1. Architecture

3.1 Formal framework

We call a session a set of visited web sites at a specific time by a given user u_i such as $i \in [1,n]$ and n is the number of users. The size of a session is limited and equal to 10. The learning database of each user u_i takes the form of a set of sessions denoted $\mathcal{S}_{u_i}$ and is built from log data[3]. We call $\mathcal{S} = \bigcup_i \mathcal{S}_{u_i}$ the whole set of sessions of the database.
We call W_{u_i} the whole set of web sites visited at least once by user u_i and we call $W = \bigcup_i W_{u_i}$ the whole set of visited sites. The order of visited web sites is not taken into account by this model.

Definition 1 (k-pattern). *Let W be a set of visited web sites and $\mathcal{S}$ be a set of sessions on W. A subset P of W is called a $k-$pattern where k is the size of P. A session S in $\mathcal{S}$ is said to contain a $k-$pattern P if $P \subseteq S$.*

Definition 2 (Support and relative support (lift)). *We define the support of a pattern P as the percentage of sessions in $\mathcal{S}$ containing P (by extension we give the support of a pattern in the set of sessions of a given user u_i):*

$$support_{\mathcal{S}}(P) = \frac{||\{S \in \mathcal{S} \mid P \subseteq S\}||}{||\mathcal{S}||} \qquad support_{S_{u_i}}(P) = \frac{||\{S \in \mathcal{S}_{u_i} \mid P \subseteq S\}||}{||\mathcal{S}_{u_i}||}$$

[3] Cf. section 4.1

For a given user the relative strength of a pattern is equivalent to the lift in a context of association rules (i.e. the support of the pattern within this user divided by the support of the pattern across all users). More formally:

$$lift_{\mathcal{S}_{u_i}\mathcal{S}}(P) = \frac{support_{S_{u_i}}(P)}{support_{\mathcal{S}}(P)}$$

The *support* measures the strength of a pattern in behavioral description of a given user. The *relative support* mitigates support measure by considering the pattern's support on the whole sessions set. The stronger the global support of a pattern, the lesser characteristic of a specific user.
The tf-idf is a numerical statistic that is intended to reflect how relevant a word is to a document in a corpus. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the whole corpus ([14]). In our context, a word becomes a pattern, a document becomes a set of sessions $\mathcal{S}_{u_i}$ of a given user and the corpus becomes the whole set $\mathcal{S}$ of all sessions.

Definition 3 ($tf \times idf$). *Let P be a pattern, let U be a set of users and $U_p \subseteq U$ such that $\forall u_i \in U_p$, $support_{S_{u_i}}(P) \neq 0$. Let $\mathcal{S}_{u_i}$ be a set of sessions of a given user u_i and $\mathcal{S}$ a whole set of sessions. The normalized term frequency denoted $tf(P)$ is equal to $support_{S_{u_i}}(P)$ and the inverse document frequency denoted $idf(P)$ is equal to log $(||U||/||U_P||)$. We have:*

$$tf \times idf(P) = support_{S_{u_i}}(P) \times log\left(\frac{||U||}{||U_P||}\right)$$

Definition 4 (Closure system). *Let $\mathcal{S}$ be a collection of sessions on the set W of web sites. We denote $\mathcal{S}^c$ the closure under intersection of $\mathcal{S}$. By adding W in $\mathcal{S}^c$, $\mathcal{S}^c$ is called a closure system.*

Definition 5 (Closure operator). *Let W be a set, a map $C: 2^W \rightarrow 2^W$ is a closure operator on W if for all sets A and B in W we have: $A \subseteq C(A)$, $A \subseteq B \Longrightarrow C(A) \subseteq C(B)$ and $C(C(A)) = C(A)$.*

Theorem 1. *Let $\mathcal{S}^c$ be a closure system on W. Then the map $C_{\mathcal{S}^c}$ defined on 2^W by $\forall A \in 2^W$, $C_{\mathcal{S}^c}(A) = \bigcap\{S \in \mathcal{S}^c \mid A \subseteq S\}$ is a closure operator on W* [4].

Definition 6 (Closed pattern[5]). *Let $\mathcal{S}^c$ be a closure system on W and $C_{\mathcal{S}^c}$ its corresponding closure operator. Let P be a pattern (i.e. a set of visited sites), we said that P is a closed pattern if $C_{\mathcal{S}^c}(P) = P$.*

[4] Refer to the book of [15].

[5] This definition is equivalent to a concept of the formal context $\mathbb{K} = (S,W,I)$ where S is a set of objects, W a set of attributes and I a binary relation between S and W [16].

3.2 Own patterns selection

The first and most important step of our model, called *own patterns selection* is to calculate the set of own patterns for each user u_i. This set of patterns is denoted $\mathcal{P}_{u_i} = \{P_{i,1}, P_{i,2},..., P_{i,p}\}$. In [1], the author states that $p = 10$ should be a reference value and that beyond this value model performance are stable. We shall follow that recommendation. In [1], 10 frequent $1 - patterns$ are selected for each user. The aim of our study is to show that it could be more efficient to select closed $k - patterns$. But, the number of closed patterns should be strong, so we compare three heuristics (H_1, H_2 and H_3) to select the 10 closed patterns of each user. For each heuristic, closed patterns are computed thanks to Charm algorithm ([17]) provided on the Coron platform ([18]). Only closed patterns with a size lower than or equal to 7 are considered. These heuristics are presented here:

1. 10 $1 - patterns$ with the largest *support* values (as in [1])
2. H_1: 40 closed $k - patterns$ with the largest *tf-idf* values.
3. H_2: 10 filtered closed $k - patterns$ with the largest *support* and maximal values by inclusion set operator.
4. H_3: 10 filtered closed $k - patterns$ with the largest *tf-idf* and minimal values by inclusion set operator.

Algorithm 1 describes the process of H_1 to select the 40 own patterns for a given user. With H_1, the model performance is improved when p increases up to 40. $p = 10$ is the better choice for H_2 and H_3. The best results are from H_1.

Algorithm 1: H_1: 40 closed $k - patterns$ with the largest *tf-idf* values.

Data: $\mathcal{C}_{u_i}$: the set of closed itemsets of user u_i from Charm;
p : the number of selected own patterns;
Result: $\mathcal{P}_{u_i}$: the set of own patterns of user u_i;
1 **begin**
2 Compute the $tf \times idf$ for each pattern from Charm;
3 Sort the list of patterns in descending order according to the $tf \times idf$ value;
4 Return the top p patterns;

3.3 User profiles computation

We define and we denote $\mathcal{P}_{all} = \bigcup_i P_{u_i}$ the whole set of own patterns. The set $\mathcal{P}_{all}$ allows us to define a common space in which all users could be embedded. More formally, $\mathcal{P}_{all}$ defines a vector space $\mathcal{V}$ of size $all = ||\mathcal{P}_{all}||$ where a given user u_i is represented as a vector $V_{u_i} = (m_{i,1},m_{i,2},...,m_{i,all})$.
The second step of our model, called *user profile computation*, is to compute, for each user u_i, a numerical value for each component $m_{i,j}$ of the vector V_{u_i}. i is the user id, $j \in [1,all]$ is a pattern *id* and m stands for a given *measure*. In this paper, we compare two measures proposed in [1]: the *support* and the *lift*.

$$m_{i,j} = support_{S_{u_i}}(P_j) \qquad and \qquad m_{i,j} = lift_{\mathcal{S}_{u_i}\mathcal{S}}(P_j)$$

3.4 Authentication stage

In our model, the authentication step is based on the identification. For that purpose, our model guesses the user corresponding to an anonymous set of sessions, then it checks if the guessed identity corresponds to the real identity. From this set of sessions we have to build a test profile and to find the nearest user profile defined during the learning step.

Test sessions Performance of our models are calculated on anonymous data sets of growing size.The more information available, the better the classification will be. The first data set consists of only one session, the second consists of 10 sessions, the third one consists of 20 sessions, and the last one consists of 30 sessions. For the test phase, all sessions have the same size of 10 sites.

Building test profile Let $\mathcal{S}$ be the whole set of sessions from the learning data set. Let $\mathcal{S}_{u_t}$ be an anonymous set of sessions and $V_{u_t} = (m_{t,1},m_{t,2},...,m_{t,all})$ its corresponding profile vector. We will compare two approaches to build the anonymous test profile, the *support* and the *lift*:

$$\forall i,\ m_{t,i} = support_{S_{u_t}}(P_i) \qquad and \qquad \forall i,\ m_{t,i} = lift_{\mathcal{S}_{u_t}\mathcal{S}} = \frac{support_{S_{u_t}}(P_i)}{support_{\mathcal{S}}(P_i)}$$

Distance functions Let $V_{u_i} = (m_{i,1},m_{i,2},...,m_{i,all})$ and $V_{u_t} = (m_{t,1},m_{t,2},...,m_{t,all})$ be two profiles. We denoted $Dis_{Euclidean}(V_{u_i},V_{u_t})$ the Euclidean distance and we denote $Sim_{Cosine}(V_{u_i},V_{u_t})$ the cosine similarity function. We have:

$$Dis_{Euclidean}(V_{u_i},V_{u_t}) = \sqrt{\sum_j (m_{t,j} - m_{i,j})^2}$$

$$Sim_{Cosine}(V_{u_i},V_{u_t}) = \frac{\sum_j (m_{t,j} \times m_{i,j})}{\sqrt{\sum_j (m_{t,j})^2 \times \sum_j (m_{i,j})^2}}$$

4 Experimental results

4.1 Data set

Our data set is comprised of the web navigation connection logs of 3,000 users over a six-month period. We have at our disposal the domain name visited and each user ID. From the variables of day and time of connection we have constructed connection sessions for each user. A session is therefore a set of web

sites visited. The number of visited web sites per session is limited and equal to 10. For the relevance of our study we used Adblock[6] filters to remove all domains regarded as advertising. The majority of users from this data set are not sufficiently active to be of relevance. Therefore, as in [1], we have limited our study to the 2% of most active users and obtained the significant session sets for 52 users. The 30 users most active (who have a large number of sessions) among those 52 users are used in this paper. Table 2 gives the detailed statistics for this data set.

7698 sessions	**Minimum**	**Maximum**	**Mean**	**Standard deviation**
Size	10	10	10	0
#sessions/users	101	733	257	289

Table 2. Descriptive statistics of the used data set: size of sessions (number of visited web sites) and number of sessions per user, for 30 users.

4.2 Experimental protocol: a description

Algorithm 2 (see appendix) describes our experimental protocol. The first loop sets the size of the set of users among which a group of anonymous sessions will be classified. The second one sets the size of this sessions group. Finally, the third loop sets the number of iterations used to compute the average accuracy rate. The loop on line 10 computes the specific patterns of each user and establishes the profiles vector. The loop on line 13 computes the vector's components for each user. The nested loops on lines 16 and 18 classify test data and compute the accuracy rate.

4.3 Comparative performance of H_1, H_2 and H_3

From own patterns of each user we compute the set $\mathcal{P}_{all}$ as the whole set of own patterns which defines the profile vector of each user. We use the support of a pattern as numerical value for each components (cf. section 3.3). Following Table 3 provides the size of the profile vector and the distribution of own patterns according to size for each heuristic. With 30 users and 10 own patterns per user, the maximal size of the profile is 300.

	Number of own patterns	|1|	|2|	|3|	|4|	|5|	|6|	|7|
H_1	199	18%	31%	26%	16%	7%	2%	0%
H_2	167	57%	29%	9%	3%	1%	1%	0%
H_3	199	24%	20%	18%	14%	10%	9%	5%

Table 3. Profile vector size and the distribution of own patterns according to size.

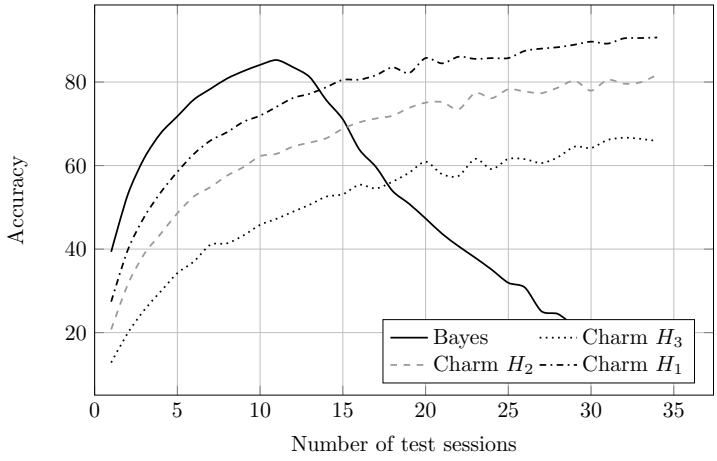


Fig. 2. Comparative performance of H_1, H_2 and H_3. These observations are plotted on an X-Y graph with number of sessions of the anonymous set on the X-axis and accuracy rate on the Y-axis. Measured values are smoothed on 50 executions.

Figure 2 shows that naive Bayes classifier is the most effective if the group of test sessions is from 1 to 13 sessions (10 to 130 visited web sites). This result is in line with the study in [1]. Finally, this graph clearly shows that heuristic H_1 certainly stands out from H_2 and H_3. So, the best heuristic is to choose owns patterns amongst closed patterns with the largest $tf \times idf$ values. As a consequence, the majority of patterns are small-sized patterns (two or three sites) (cf. Table 3). But accuracy rates are much higher.

4.4 Comparative performance with [1]

In [1], the author compares, in particular, two methods of profile vector calculus. In both cases, the own patterns are size 1 and are chosen amongst the most frequent. The first method, named support-based profiling, uses the corresponding support pattern as the numerical value for each component of the profile vector. The second method, called lift-based profiling, uses the lift measure. In order to compare the performances of the H_1 model with the two models support-based profiling and lift-based profiling, we have accurately replicated the experimental protocol described in [1] on our own data set. The results are given in Table 4.

The data of Table 4 highlight that the H_1 heuristic allows rates that are perceptibly better than those of the two models proposed in [1] in all possible scenarios. Nevertheless, the Bayes classifier remains the most efficient when the session group is size 1 in compliance with [1]. Figure 3 allows a clearer understanding of the moment the Bayes curve crosses the H_1 heuristic curve.

[6] http://adblock-listefr.com/

# of users		1	10	20	30
2	Support	65	89	95	97
	Lift	67	90	97	98
	Charm H_1	72	98	**99**	**100**
	Bayes	**85**	**99**	73	61
5	Support	40	74	83	88
	Lift	41	78	86	88
	Charm H_1	49	90	**95**	**98**
	Bayes	**67**	**96**	56	34
10	Support	27	66	79	80
	Lift	29	64	77	80
	Charm H_1	37	83	**92**	**94**
	Bayes	**54**	**91**	51	24
20	Support	19	55	68	75
	Lift	21	58	68	74
	Charm H_1	30	76	**86**	**90**
	Bayes	**43**	**87**	48	19
30	Support	16	53	64	70
	Lift	17	54	64	69
	Charm H_1	26	72	**83**	**89**
	Bayes	**39**	**83**	46	19

Table 4. On left, we find the number of users and the selected model. Each column is defined by the number of sessions of the anonymous data set. Sessions are of size 10. Measured accuracy rate are smoothed on 100 executions. In bold the best values are presented.

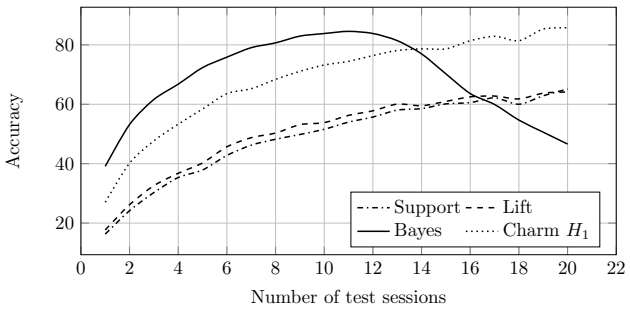


Fig. 3. Comparative performance of Bayes, *support-based profiling*, *lift-based profiling* and H_1. These observations are plotted on an X-Y graph with number of sessions of the anonymous set on the X-axis and accuracy rate on the Y-axis. Number of users is equal to 30. Measured values are smoothed on 50 executions.

4.5 Comparative performance of distance functions

The last figure 4 shows the impact of distance function choice on performances of models.

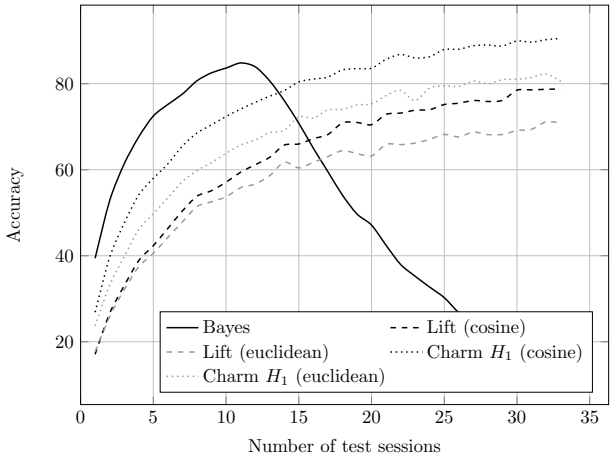


Fig. 4. Comparative performance of both H_1 with cosine similarity and Euclidean distance, Bayes and *lift-based profiling*. These observations are plotted on an X-Y graph with number of sessions of the anonymous set on the X-axis and accuracy rate on the Y-axis. Number of users is equal to 30. Measured values are smoothed on 100 executions.

Figure 4 illustrates the significance of the distance function concerning the performance. Indeed, when used with Euclidean distance, the H_1 method is a bit more precise than the lift one (about 3%). However, performances are improved by using the cosine similarity and their relative ranking is even reversed. H_1 method's performance are then better than lift by 10%.

5 Conclusions and future work

In this study, we proposed a learning model for implicit authentication of web users. We proposed an simple and original algorithm (cf. Algorithm 1) to get a set of own patterns allowing to characterize each web user. The taken patterns have different size and qualify as closed patterns from closure system generated by the set of sessions (cf. Table 3). By reproducing experimental protocol described in [1], we showed that the performances of our model are significantly better than some models proposed in the literature (cf. Table 4). We also showed the key role of the distance function (cf. Figure 4).

This study should be extended in order to improve the obtained results. For a very small sites flow, the results of the solution should be better than results

from Bayes' method. Another way to improve results will be to select other types of variable and to add them to our current dataset. The selection of data has an undeniable impact on the results.

References

1. Yang, Y.C.: Web user behavioral profiling for user identification. Decision Support Systems (49) (2010) pp. 261–271
2. Guvence-Rodoper, C.I., Benbasat, I., Cenfetelli, R.T.: Adoption of B2B Exchanges: Effects of IT-Mediated Website Services, Website Functionality, Benefits, and Costs. ICIS 2008 Proceedings (2008)
3. Lagier, F.: Cybercriminalité : 120.000 victimes d'usurpation d'identité chaque année en france. Le populaire du centre (in French) (2013)
4. Filson, D.: The impact of e-commerce strategies on firm value: Lessons from Amazon.com and its early competitors. The Journal of Business **77**(S2) (2004) pp. S135–S154
5. He, R., Yuan, M., Hu, J., Zhang, H., Kan, Z., Ma, J.: A novel service-oriented AAA architecture. **3** (2003) pp. 2833–2837
6. Stockinger, T.: Implicit authentication on mobile devices. The Media Informatics Advanced Seminar on Ubiquitous Computing (2011)
7. Shi, E., Niu, Y., Jakobsson, M., Chow, R.: Implicit authentication through learning user behavior. M. Burmester et al. (Eds.): ISC 2010, LNCS 6531 (2011) pp. 99–113
8. Jakobsson, M., Shi, E., Golle, P., Chow, R.: Implicit authentication for mobile devices. Proceeding HotSec'09 Proceedings of the 4th USENIX conference on Hot topics in security (2009) pp. 9–9
9. Ullah, I., Bonnet, G., Doyen, G., Gaïti, D.: Un classifieur du comportement des utilisateurs dans les applications pair-à-pair de streaming vidéo. CFIP 2011 - Colloque Francophone sur l'Ingénierie des Protocoles (in French) (2011)
10. Goel, S., Hofman, J.M., Sirer, M.I.: Who does what on the web: A large-scale study of browsing behavior. In: ICWSM. (2012)
11. Kumar, R., Tomkins, A.: A characterization of online browsing behavior. In: Proceedings of the 19th international conference on World wide web, ACM (2010) pp. 561–570
12. Abramson, M., Aha, D.W.: User authentication from web browsing behavior. Proceedings of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference pp. 268–273
13. Herrmann, D., Banse, C., Federrath, H.: Behavior-based tracking: Exploiting characteristic patterns in DNS traffic. Computer & Security (2013) pp. 1–17
14. Salton, G.: Automatic text processing: The transformation, analysis and retrieval of information by computer. Addison Wesley (1989)
15. Davey, B.A., Priestley, H.A.: Introduction to lattices and orders. Cambridge University Press (1991)
16. Ganter, B., Wille, R.: Formal concept analysis, mathematical foundation, Berlin-Heidelberg-NewYork et al.:Springer (1999)
17. Zaki, M.J., Hsiao, C.: Efficient algorithms for mining closed itemsets and their lattice structure. IEEE Transactions on knowledge and data engineering **17**(4) (2002) pp. 462–478
18. Szathmary, L.: Symbolic Data Mining Methods with the Coron Platform. PhD Thesis in Computer Science, University Henri Poincaré – Nancy 1, France (Nov 2006)

Appendix

Algorithm 2: Experiment procedure

Data: $\bigcup_i \mathcal{S}_{u_i}$: all sessions from n users;
X : number of successive executions;
Result: The mean accuracy of select models;

begin
  for *(*$N = \{2, 5, 10, 20, 30\}$*)* **do**
    for *(*$S = \{1, 10, 20, 30\}$*)* **do**
      for *(*$z = 1, \ldots, X$*)* **do**
        Select N random users;
        For each user, select $SN = min(|\mathcal{S}_{u_i}|,\ i = 1, \ldots, N)$;
        Take the $\frac{2}{3}$ of the SN sessions from each users to form the training set;
        Take the rest of SN sessions to form the validation set;
        $\mathcal{P}^k_{all} \leftarrow \emptyset$ (the global profile vector for each model k);
        for *each (*$u_i,\ i = 1, \ldots, N$*)* **do**
          Compute the own patterns $\mathcal{P}^k_{u_i}$ $(1 \leq |\mathcal{P}^k_{u_i}| \leq 10)$;
          $\mathcal{P}^k_{all} \leftarrow \mathcal{P}^k_{all} \cup \mathcal{P}^k_{u_i}$;
        for *each (*$u_i,\ i = 1, \ldots, N$*)* **do**
          Compute the vector $V^k_{u_i}$ with support or lift;
        Initialize to 0 the confusion matrix M^k of the method k;
        for *each (*$u_i,\ i = 1, \ldots, N$ **do**
          Compute the test stream $\mathcal{T}_{u_i}$ ($|T|$ is fixed, $T \in \mathcal{T}_{u_i}$);
          while *(*$\mathcal{T}_{u_i} \neq \emptyset$*)* **do**
            Take SW sessions from $\mathcal{T}_{u_i}$ to compute V^k_T;
            $u_a \leftarrow max(simil(V^k_{u_i}, V^k_T))$ or $min(dist(V^k_{u_i}, V^k_T))$;
            $M^k[u_i][u_a] \leftarrow M^k[u_i][u_a] + 1$;
        Compute the mean accuracy of k from M^k;

The direct-optimal basis via reductions

Estrella Rodríguez-Lorenzo[1], Karell Bertet[2], Pablo Cordero[1], Manuel Enciso[1], and Angel Mora[1]

[1] University of Málaga, Andalucía Tech, Spain,
e-mail: {estrellarodlor,amora}@ctima.uma.es, {pcordero,enciso}@uma.es
[2] Laboratoire 3I, Université de La Rochelle
e-mail: karell.bertet@univ-lr.fr

Abstract. Formal Concept Analysis has become a real approach in the trend Information-Knowledge-Wisdom. It turns around the mining of a data set to built a concept lattice which provides an strong structure of the knowledge. Implications play the role of an alternative specification of this concept lattice and may be managed by means of inference rules. This syntactic treatment is guided by several properties like directness, minimality, optimality, etc. In this work, we propose a method to calculate the direct-optimal basis equivalent to a given Implicational System. Our method deals with unitary and non-unitary implications. Moreover, it shows a better performance that previous methods in the literature by means of the use of Simplification Logic and reduction paradigm, which remains narrow implications in any stage of the process. We have also developed an empirical study to compare our method with previous approaches in the literature.

1 Introduction

Formal Concept Analysis (FCA) is a trending upward area which establishes a proper and fine mixture of formalism, data analysis and knowledge discovering. It is able to analyze and extract information from a context $\mathbf{K}$, rendering a concept lattice. Attribute implications [10] represent implicit knowledge between data and they can be deduced from the concept lattice or using mining techniques from the context directly. An attribute implication is an expression $A \to B$ where A and B are sets of attributes. A context satisfies $A \to B$ if every object that has all the attributes in A has also all the attributes in B.

The study of sets of implications that satisfies some criteria is one of the relevant topics in FCA. An implicational system (IS) of $\mathbf{K}$ is defined as a set Σ of implications of $\mathbf{K}$ from which any valid implication for $\mathbf{K}$ can be deduced by means a syntactic treatment of the implications. This symbolic manipulation introduces the notion of equivalent sets of implications and opens the door to the definition of several criteria to discriminate good sets of implications according to these criteria. Thus, the challenges are the definition of an specific notion of IS, named basis, fulfilling some criteria related with minimality and the introduction of efficient methods to transform an arbitrary IS into a basis.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

For instance, if the criteria is to obtain an IS with minimum cardinal we can build the so-called Duquenne-Guigues (or stem) basis [11]. Each application may induces a different criterium. For instance, in [2, 3] some methods to calculate the direct-optimal basis are introduced, joining minimality and directness in the same notion of basis. In [8] a method to obtain a basis with minimal size in the left-hand size of the implications was proposed.

In this paper, we introduce a method to compute the direct-optimal basis. This kind of basis was introduced in [2,3] and it has two interesting properties: it has the minimum number of attributes and it provides a framework to efficiently compute the closure of a set of attributes. The new method introduced in this paper is strongly based on $\mathbf{SL}_{\text{FD}}$ (Simplification Logic) and they are more efficient than previous methods appeared in the literature.

In the following, first we establish the background necessary for the understanding of the paper (Section 2). In Section 3 $\mathbf{SL}_{\text{FD}}$ is summarized and a motivation of the simplification paradigm to remove redundant attributes is provided. Section 4 is focussed on the methods of Bertet et al. to get a direct-optimal basis. In Section 5 the new method is introduced and a comparison among all the methods is showed. Some conclusions are presented in Section 6.

2 Preliminaries

We assume well-known the main concepts in FCA [10]. Only the concepts necessaries will be introduced. In Formal Concept Analysis (FCA) the relationship between a set of objects and a set of attributes are described using a formal context as follows:

Definition 1. *A formal context is a triple* $\mathbf{K} = (G, M, I)$ *where* G *is a finite set whose elements are named objects,* M *is a finite set whose elements are named attributes and* $I \subseteq G \times M$ *is a binary relation. Thus,* $(o, a) \in I$ *means the object* o *has the attribute* a*.*

This paper focuses on the notion of implication, which can be introduced as follows:

Definition 2. *Let* $\mathbf{K} = (G, M, I)$ *be a formal context and* $A, B \in 2^M$*. The implication* $A \to B$ *holds in* $\mathbf{K}$ *if every object* $o \in G$ *satisfies the following:* $(o, a) \in I$ *for all* $a \in A$ *implies* $(o, b) \in I$ *for all* $b \in B$*.*

An implication $A \to B$ *is said to be unitary if the set* B *is a singleton.*

Implications may be syntactically managed by means of inference systems. The former axiomatic system was Armstrong's Axioms [1]. They allows us to introduce the notion of derivation of an implication from an implicational system, the semantic entailment and the equivalence between two implicational systems in the usual way.

3 Simplification Logic

In [6], Cordero et al. introduced the Simplification Logic, $\mathbf{SL}_{\text{FD}}$, that is, an equivalent logic to the Armstrong's Axioms that avoids the use of transitivity and is guided by the idea of simplifying the set of implications by removing redundant attributes efficiently. This logic has proved to be useful for automated reasoning with implications [7, 8, 12, 13].

Definition 3 (Language). *Given a non-empty finite alphabet S (whose elements are named attributes and denoted by lowercase letters a, b, c, etc.), the language of $\mathbf{SL}_{\text{FD}}$ is $\mathcal{L}_S = \{A \to B \mid A, B \subseteq S\}$.*

Sets of formulas (implications) will be named implicational systems (IS). In order to distinguish between language and metalanguage, inside implications, AB means $A \cup B$ and A-B denotes the set difference $A \setminus B$. Moreover, when no confusion arises, we omit the brackets, e.g. abc denotes the set $\{a, b, c\}$.

Definition 4 (Semantics). *Let $\mathbf{K} = (G, M, I)$ be a context and $A \to B \in \mathcal{L}_S$. The context $\mathbf{K}$ is said to be a model for $A \to B$, denoted $\mathbf{K} \models A \to B$, if $A, B \subseteq M \subseteq S$ and $A \to B$ holds in $\mathbf{K}$.*

For a context $\mathbf{K}$ and an IS Σ, then $\mathbf{K} \models \Sigma$ means $\mathbf{K} \models A \to B$ for all $A \to B \in \Sigma$ and $\Sigma \models A \to B$ denotes that every model for Σ is also a model for $A \to B$. If Σ_1 and Σ_2 are implicational systems, $\Sigma_1 \equiv \Sigma_2$ denotes both IS are equivalent (i.e. $\mathbf{K} \models \Sigma_1$ iff $\mathbf{K} \models \Sigma_2$ for all context $\mathbf{K}$).

Definition 5 (Syntactic derivations). $\mathbf{SL}_{\text{FD}}$ *considers reflexivity axioms*

$$\texttt{[Ref]}\ \frac{B \subseteq A}{A \to B};$$

and the following inference rules named fragmentation, composition and simplification respectively.

$$\texttt{[Frag]}\ \frac{A \to BC}{A \to B};\quad \texttt{[Comp]}\ \frac{A \to B,\ C \to D}{AC \to BD};\quad \texttt{[Simp]}\ \textit{If } A \subseteq C, A \cap B = \emptyset,\ \frac{A \to B,\ C \to D}{C\text{-}B \to D\text{-}B}$$

Given an IS Σ and a formula $A \to B$, $\Sigma \vdash A \to B$ denotes that $A \to B$ can be derived from Σ by using the axiomatic system in a standard way. The above axiomatic system is sound and complete (i.e. $\Sigma \models A \to B$ iff $\Sigma \vdash A \to B$). The main advantage of $\mathbf{SL}_{\text{FD}}$ is that inferences rules may be considered equivalence rules and they are enough to compute all the derivations (see [12] for further details and proofs).

Theorem 1 (Mora et al. [12]). *In $\mathbf{SL}_{\text{FD}}$ logic, the following equivalencies hold:*

1. *Fragmentation Equivalency* **[FrEq]***: $\{A \to B\} \equiv \{A \to B\text{-}A\}$.*
2. *Composition Equivalency* **[CoEq]***: $\{A \to B, A \to C\} \equiv \{A \to BC\}$.*
3. *Simplification Equivalency* **[SiEq]***: If $A \cap B = \emptyset$ and $A \subseteq C$ then*
$$\{A \to B, C \to D\} \equiv \{A \to B, C\text{-}B \to D\text{-}B\}$$
4. *Right Simplification Equivalency* **[rSiEq]***: If $A \cap B = \emptyset$ and $A \subseteq C \cup D$ then*
$$\{A \to B, C \to D\} \equiv \{A \to B, C \to D\text{-}B\}$$

Note that these equivalencies (reading from left to right) remove redundant information. $\mathbf{SL}_{\text{FD}}$ was conceived as a simplification framework.

To conclude this section, we introduce the outstanding notion of closure of a set of attributes, which is strongly related with the syntactic treatment of implications.

Definition 6. *Let $\Sigma \subseteq \mathcal{L}_S$ be an IS and $X \subseteq S$. The closure of X wrt Σ is the largest subset of S, noted X^+_Σ, such that $\Sigma \vdash X \to X^+_\Sigma$.*

We omit the subindex (i.e. we write X^+) when no confusion arise. Given a context $\mathbf{K}$ and an IS Σ satisfying $\mathbf{K} \models A \to B$ iff $\Sigma \vdash A \to B$, it is well-known that the closed sets of attributes wrt Σ are in bijection with the concepts of $\mathbf{K}$.

One of the main topics is the computation of the closure of a set of attributes, and for this reason, it is necessary to have an efficient method to calculate closures. We emphasize for this problem, the works of Bertet et al. in [2,3] and Cordero et al. in [12].

4 Direct-Optimal basis

The study of sets of implications that satisfies some criteria is one of the most important topics in FCA. In [3], Bertet and Monjardet present a survey about implicational systems and basis. They show the equality between five unit basis originating from different works (minimal functional dependencies in database theory, knowledge spaces, etc.) and satisfying various properties including the directness canonical and minimal properties, whence the name canonical direct basis is given to this basis. The direct-optimal basis belong to these five basis. In the following, we show only the concepts used in the rest of the paper of this survey.

Definition 7. *An IS Σ is said to be:*

- minimal *if $\Sigma \smallsetminus \{A \to B\} \not\equiv \Sigma$ for all $A \to B \in \Sigma$.*
- minimum *if $\Sigma' \equiv \Sigma$ implies $|\Sigma| \leq |\Sigma'|$, for all IS Σ'.*
- optimal *if $\Sigma' \equiv \Sigma$ implies $\|\Sigma\| \leq \|\Sigma'\|$, for all IS Σ'.*

where $|\Sigma|$ is the cardinality of Σ and $\|\Sigma\|$ is its size, ie $\|\Sigma\| = \sum_{A \to B \in \Sigma} (|A|+|B|)$.

A minimal set of implications is named a basis, and a minimum basis is then a basis of least cardinality. Let us now introduce the main property used in this paper, namely the direct-optimal property.

Definition 8. *An IS Σ is said to be* direct *if, for all $X \subseteq S$:*

$$X^+ = X \cup \bigcup \{B \mid A \subseteq X \text{ and } A \to B \in \Sigma\}$$

Moreover, Σ is said to be direct-optimal *if it is direct and, for any direct IS Σ', $\Sigma' \equiv \Sigma$ implies $\|\Sigma\| \leq \|\Sigma'\|$.*

In words, Σ is direct if the computation of the closure of any attribute set wrt Σ requires only one iteration, that is, a unique traversal of the set of implications. Obviously, the direct-optimal property is the combination of the directness and optimality properties. In [2], Bertet and Nebut show that a direct-optimal IS is unique and can be obtained from any equivalent IS. We address this procedure in this paper.

As we have said in the preliminaries, one of the most important problems is how to calculate quickly and easily the closure X^+ of any set X because a number of problems related to an IS Σ can be answered by computing closures. For this reason, Bertet et al. propose a type of base called direct-optimal basis [2,3], so one can compute closures of subsets in only one iteration. Section 4.1. presents the basis proposed in [2] by Bertet and Nebut where they work with non-unitary implicational systems (IS). Section 4.2 shows how to obtain a unit direct-optimal basis [3]. In both sections, we illustrate the algorithms needed to obtain a direct-optimal basis equivalent to any implicational system.

4.1 Computing Direct-Optimal basis

In this section, the algorithm proposed by Bertet and Nebut in [2] is showed. The key of the method is the so-called "overlap axiom" that can be directly proved by using the axiomatic system from Definition 5.

`[Overlap]` for all $A, B, C, D \subseteq S$: If $B \cap C \neq \emptyset$, $\dfrac{A \to B, C \to D}{A(C\text{-}B) \to D}$

Then, the direct implicational system generated from an IS Σ is defined as the smallest IS that contains Σ and is closed for `[Overlap]`.

Definition 9. *The direct implicational system Σ_d generated from Σ is defined as the smallest IS such that:*

1. *$\Sigma \subseteq \Sigma_d$ and*
2. *For all $A, B, C, D \subseteq S$, if $A \to B, C \to D \in \Sigma_d$ and $B \cap C \neq \emptyset$ then $A(C\text{-}B) \to D \in \Sigma_d$.*

Function Bertet-Nebut-Direct(Σ)

input : An implicational system Σ on S
output: The direct IS Σ_d on S equivalent to Σ
begin
  $\Sigma_d := \Sigma$
  foreach $A \to B \in \Sigma_d$ **do**
    foreach $C \to D \in \Sigma_d$ **do**
      if $B \cap C \neq \emptyset$ **then** add $A(C\text{-}B) \to D$ to Σ_d;
  return Σ_d

Theorem 2 (Bertet and Nebut [2]). *Let Σ be an implicational system. Then $\Sigma_d =$ `Bertet-Nebut-Direct`(Σ) is a direct basis.*

Moreover, if an IS Σ is direct but not direct-optimal, then there exists an equivalent IS Σ' of smaller size which is direct-optimal. The properties that it must hold are the following:

Theorem 3 (Bertet and Nebut [2]). *A direct IS Σ is direct-optimal if and only if the following properties hold.*

Extensiveness: *for all $A \to B \in \Sigma$, $A \cap B = \emptyset$.*
Isotony: *for all $A \to B, C \to D \in \Sigma$, $C \subsetneq A$ implies $B \cap D = \emptyset$.*
Premise: *if $A \to B, A \to B' \in \Sigma$ then $B = B'$.*
Not empty conclusion: *if $A \to B \in \Sigma$ then $B \neq \emptyset$.*

Function Bertet-Nebut-Minimize(Σ)

```
input  : An implicational system Σ on S
output: An smaller IS Σ_m on S equivalent to Σ
begin
    Σ_m := ∅
    foreach A → B ∈ Σ do
        B' := B
        foreach C → D ∈ Σ do
            if C = A then B' := B' ∪ D;
            if C ⊊ A then B' := B' ∖ D;
        B' := B' ∖ A
        add A → B' to Σ_m
    return Σ_m
```

Function `Bertet-Nebut-DO` computes the direct-optimal basis Σ_{do} generated from an IS Σ. It first computes Σ_d using Function `Bertet-Nebut-Direct` and then minimizes Σ_d using Function `Bertet-Nebut-Minimize`.

Function Bertet-Nebut-DO(Σ)

```
input  : An implicational system Σ on S
output: The direct-optimal IS Σ_do on S equivalent to Σ
begin
    Σ_d = Bertet-Nebut-direct(Σ)
    Σ_do = Bertet-Nebut-Minimize(Σ_d)
    return Σ_do
```

Theorem 4 (Bertet and Nebut [2]). *Let Σ be an implicational system. Then Σ_{do} = `Bertet-Nebut-DO`(Σ) is the unique direct-optimal implicational system equivalent to Σ.*

4.2 Direct-Optimal basis by means of unit implicational systems

In some areas, the management of formulas is limited to unitary ones. Thus, the use of Horn Clauses in Logic Programming is widely accepted. Such a language restriction allows an improvement in the performance of the methods, which are more direct and lighter. Nevertheless, the advantages provided by the

limited languages have a counterpart: a significant growth of the input set. In this section we are going to present new versions of the definitions and methods introduced above restricted to Unit Implicational System (UIS), i.e. set of implications with unitary right-hand sides. An UIS is named proper if it does not contain implications $A \to a$ such that $a \in A$.

In this line, Bertet [4] provided versions for unit implicational systems of Functions `Bertet-Nebut-Direct` and `Bertet-Nebut-Minimize`.

Function Bertet-Unit-Direct(Σ)

input : A proper UIS Σ on S
output: The direct UIS Σ_d on S equivalent to Σ
begin
  $\Sigma_d := \Sigma$
  foreach $A \to a \in \Sigma_d$ **do**
    foreach $Ca \to b \in \Sigma_d$ **do**
      if $a \neq b$ *and* $b \notin A$ **then** add $AC \to b$ to Σ_d;
  return Σ_d

Function Bertet-Unit-Minimize(Σ)

input : A proper UIS Σ on S
output: An smaller UIS Σ_m on S equivalent to Σ
begin
  $\Sigma_m := \Sigma$
  foreach $A \to b \in \Sigma_m$ **do**
    foreach $C \to b \in \Sigma_m$ **do**
      if $A \subsetneq C$ **then** delete $C \to b$ from Σ_m;
  return Σ_m

The above functions was used in [4] to build a method which transforms an arbitrary UIS into an UIS with the same properties that the direct-optimal basis for general IS. Since any non-unit IS can be trivially turned into an UIS, we may encapsulate both functions to provide another method to get a direct-optimal basis from and arbitrary IS. Thus, the following function incorporates a first step to convert any IS into its equivalent UIS and concludes with the converse switch.

Function Bertet-Unit-DO(Σ)

input : An implicational system Σ on S
output: The direct-optimal IS Σ_{do} on S equivalent to Σ
begin
  $\Sigma_u := \{A \to b \mid A \to B \in \Sigma \text{ and } b \in B \smallsetminus A\}$
  $\Sigma_{ud} :=$ Bertet-Unit-Direct(Σ_u)
  $\Sigma_{udo} :=$ Bertet-Unit-Minimize(Σ_{ud})
  $\Sigma_{do} := \{A \to B \mid B = \{b \mid A \to b \in \Sigma\} \neq \emptyset\}$
  return Σ_{do}

Theorem 5 (Bertet [4]). *Let Σ be an IS. Then Σ_{do} = `Bertet-Unit-DO`(Σ) is the unique direct-optimal implicational system equivalent to Σ.*

As we have mentioned at the beginning of this subsection, some authors introduce unitary formulas as a way to provide simpler and more direct methods having a better performance. Thus, in this case, `Bertet-Unit-DO` is more efficient than `Bertet-Nebut-DO`, as we shall see at the end of the paper in Section 5.1.

5 Computing direct-optimal basis by means of reductions

In this paper, our goal is the integration of the techniques proposed by Bertet et al. [2–4] and the Simplification Logic proposed by Cordero et al. [6], that is, the adding of reductions based on the simplification paradigm to build a direct-optimal basis.

In the same way that `Bertet-Unit-DO`, we are going to develop a function to get direct-optimal basis whose first step will be to narrow the implications. However, the use of unit implications has some disadvantages that we are going to avoid by considering another kind of formulas. Thus, we are going to use *reduced* IS and introduce simplification rules which transform it preserving reduceness. A signal which indicates it is a good approach is the fact that at the end of the process, the function renders the direct-optimal basis directly, avoiding the converse switch.

Definition 10. *An IS Σ is* reduced *if $A \to B \in \Sigma$ implies $B \neq \emptyset$ and $A \cap B = \emptyset$ for all $A, B \subseteq S$.*

Obviously, any IS Σ can be turned into a reduced equivalent one Σ_r as follows

$$\Sigma_r := \{A \to B\text{-}A \mid A \to B \in \Sigma, B \not\subseteq A\}$$

The method proposed begins with this transformation and, once the IS is reduced, this property is preserved. For this reason, `[Overlap]` must be substituted. Thus, we introduce a new inference rule covering directness without losing reduceness and, at the same time, it makes progress on the minimization task following the simplification paradigm. The kernel of the new method is the following inference rule, named strong simplification:

$$\texttt{[sSimp]} \quad \text{If } B \cap C \neq \emptyset \text{ and } D \not\subseteq A \cup B, \ \frac{A \to B, C \to D}{A(C\text{-}B) \to D\text{-}(AB)}$$

Regardless the conditions, the inference rule always holds. Nevertheless, the conditions ensure a precise application of the rule in those cases where it is necessary.

Definition 11. *Given a reduced IS Σ, the* direct-reduced implicational system *Σ_{dr} generated from Σ is defined as the smallest IS such that*

1. *$\Sigma \subseteq \Sigma_{dr}$ and*
2. *For all $A, B, C, D \subseteq S$, if $A \to B, C \to D \in \Sigma_{dr}$, $B \cap C \neq \emptyset$ and $D \not\subseteq A \cup B$ then $AC\text{-}B \to D\text{-}(AB) \in \Sigma_{dr}$*

Theorem 6. *Given a reduced IS Σ, then Σ_{dr} =*`Direct-Reduced`*(Σ) is a direct and reduced IS.*

Function Direct-Reduced(Σ)

---

input : A reduced implicational system Σ on S
output: The direct-reduced IS Σ_{dr} on S
begin
  foreach $A \to B \in \Sigma_{dr}$ *and* $C \to D \in \Sigma_{dr}$ **do**
    if $B \cap C \neq \emptyset \neq D \smallsetminus (A \cup B)$ **then** add AC-$B \to D$-(AB) to Σ_{dr};
  return Σ_{dr}

Theorem 1 provides four equivalencies which allow to remove redundant information when they are read from left to right. An implicational system in which these equivalences are used to remove redundant information is going to be named simplified implicational system.

Definition 12. *A reduced IS Σ is* simplified *if the following conditions hold: for all $A, B, C, D \subseteq S$,*

1. *$A \to B$, $A \to C \in \Sigma$ implies $B = C$.*
2. *$A \to B$, $C \to D \in \Sigma$ and $A \subsetneq C$ imply $C \cap B = \emptyset = D \cap B$.*

Then, Function `RD-Simplify` turns any direct-reduced IS into a direct-reduced-simplified equivalent one by systematically applying the equivalences provided in Theorem 1.

Function RD-Simplify(Σ)

---

input : A direct-reduced implicational system Σ on S
output: The direct-reduced-simplified IS Σ_{drs} on S equivalent to Σ
begin
  $\Sigma_{drs} := \emptyset$
  foreach $A \to B \in \Sigma$ **do**
    foreach $C \to D \in \Sigma$ **do**
      if $C = A$ **then** $B := B \cup D$;
      if $C \subsetneq A$ **then** $B := B \smallsetminus D$;
    if $B \neq \emptyset$ **then** add $A \to B$ to Σ_{drs};
  return Σ_{drs}

Function doSimp(Σ)

---

input : An implicational system Σ on S
output: The direct-optimal IS Σ_{do} on S
begin
  $\Sigma_r := \{A \to B\text{-}A \mid A \to B \in \Sigma, B \not\subseteq A\}$
  $\Sigma_{dr} :=$ Direct-Reduced(Σ_r)
  $\Sigma_{do} :=$ RD-Simplify(Σ_{dr})
  return Σ_{do}

Theorem 7. *Let Σ be an implicational system on S. Then, $\Sigma_{do} =$* `doSimp`(Σ) *is the direct-optimal basis equivalent to Σ.*

Note that, unlike `Bertet-Unit-DO` where a final step was needed to revert the effects of the first transformation, `doSimp` do not need to revert the first step. We conclude this section with an experiment which illustrates the advantages of the new method.

5.1 Empirical results

Logic programming has been used as a natural framework in the areas in which it is neccessary to develop automatic deduction methods. The Prolog prototypes provides a declarative and pedagogical point of departure and illustrates the behavior of new techniques in a very fast and easy way.

Some authors have explored the use of Logic Programming in the framework of Formal Concept Analysis. Even, in [5] the authors consider the framework of FCA and its implementation in logic programming as a previous step to achieve the first order logic FCA theory. In Eden et al. [9], the authors present a PROLOG-based prototype tool and show how the tool can utilize formulas to locate pattern instances.

In a first step, the methods proposed in this paper have been developed in a Logic Programming language (Prolog) that is a well-known tool to develop fast prototypes. In our case, the implementation in Prolog is close because the method proposed in this paper is based on logic.

The methods of Bertet et al. [2,3] and our `doSimp` method have been implemented in Swi-Prolog.[1] Since there does not exist a benchmark for implications in this experiment, we have collected some sets of implications from the literature, searching papers and books with works about algorithms for implications, functional dependencies and minimal keys. Now, we are going to show the results of the execution of a first Prolog prototype of Bertet et al. for UIS [3], Bertet et al. for IS [2] and the new doSimp (proposed in this paper) methods.

The following table and figures summarize the results obtained. We show in the columns the results of Prolog: Lips (logical inferences per second lips - used to describe the performance of a logical reasoning system), Time (execution time in seconds), and Comp (the number of couple of implications in which a rule is applied). Areas in Figure 2 show the percentages of each algorithm with respect the number of comparisons.

[1] Available at `http://www.lcc.uma.es/~enciso/doSimp.zip`

Lips/Time/Comp.	Bertet-Nebut-DO			Bertet-Unit-DO			Direct-Reduced		
Ex.1	5297080	1247	1978	116905	0.019	36	**4281**	**0.001**	**12**
Ex.2	2395	0.003	23	923	**0**	3	**606**	**0**	**2**
Ex.3	2183	**0**	15	1440	**0**	**4**	**1122**	**0**	**4**
Ex.a	83403	0.019	297	44109	0.007	33	**3048**	**0.001**	**4**
Ex.a3red	27613	0.005	100	16938	0.003	20	**3698**	**0.001**	**15**
Ex.derivation5	10302	0.002	120	3522	**0.001**	**8**	**1782**	**0.001**	12
Ex.Olomouc	15399581	4528	4337	1526818	0.331	180	**15568**	**0.003**	**72**
Ex.Ganter	116514	0.025	230	72153	0.16	36	**3756**	**0.001**	**12**
Ex.CLA14	102971	0.022	204	7449	0.001	12	**704**	**0**	**3**
Ex.Saedian1	18754	0.004	97	10349	0.002	**14**	**4064**	**0.001**	16
Ex.Saedian2	19452	0.004	160	10549	0.002	**13**	**2619**	**0.001**	**13**
EX.Saedian3	5753962	1262	1986	166566	0.028	67	**24643**	**0.005**	**55**
Ex.Wastl10	1242	**0**	18	381	**0**	**1**	**327**	**0**	**1**
Ex.Wastl13	10543	0.002	86	4674	**0**	10	**1029**	**0**	**5**
Example1	5594556921	7008.890	134175	2662181973	1351.950	5389	**1199498**	**0.197**	**1103**

	IS Bertet − Nebut	*UIS Bertet*	*doSimp*
Lips - logical inferences	374, 760, 194.4	177, 610, 983.3	84, 449.66667
Time of execution (seconds)	467, 728.5	90, 130.03693	0.014
Number of comparisons	9588.4	388.4	88.6

Fig. 1. Summary of the experiment (average)

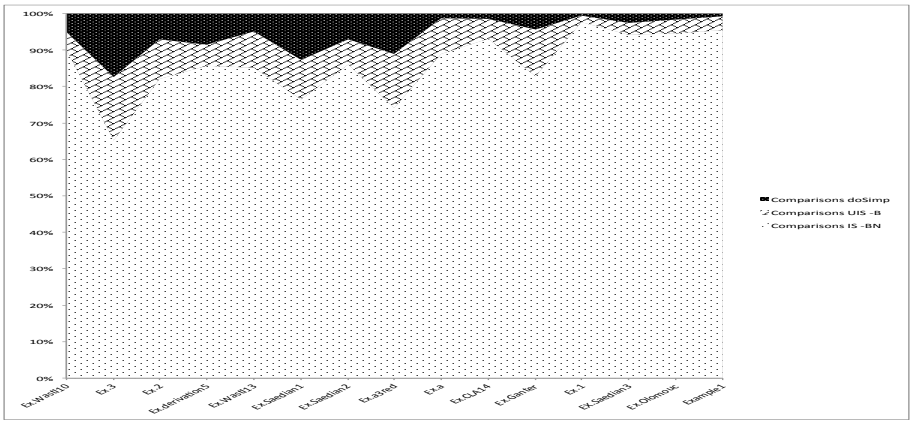


Fig. 2. Results: Comparisons

6 Conclusion

In this work, we have presented another algorithm to calculate the direct-optimal basis in a further way, in the most of the cases, than the algorithms which exist in

the literature. It is shown with a test that we have realised by running different examples with the methods of Bertet et al. for UIS [3], Bertet et al. for IS [2] and the new `doSimp`.

Our aim is to reduce the cost of the algorithm by using the Simplification Logic as a useful tool to work with implications. By the time, we have improved the algorithms that existed but we are going to go on working in that way to try to cut down the cost of our method.

The perspectives we have are improvements by pretreatments: reduction, canonical basis, etc in order to reach our main objective which would be to directly compute the direct-optimal basis without extra implication generation.

Acknowledgment

Supported by grant TIN11-28084 of the Science and Innovation Ministry of Spain.

References

1. W W. Armstrong, *Dependency structures of data base relationships*, Proc. IFIP Congress. North Holland, Amsterdam: 580–583, 1974.
2. K. Bertet, M. Nebut, *Efficient algorithms on the Moore family associated to an implicational system*, DMTCS, 6(2): 315–338, 2004.
3. K. Bertet, B. Monjardet, *The multiple facets of the canonical direct unit implicational basis*, Theor. Comput. Sci., 411(22-24): 2155–2166, 2010.
4. K. Bertet, *Some Algorithmical Aspects Using the Canonical Direct Implicationnal Basis*, CLA:101–114, 2006.
5. L. Chaudron, N. Maille, *1st Order Logic Formal Concept Analysis: from logic programming to theory*, Computer and Informations Science: (13:3),1998.
6. P Cordero, A. Mora, M. Enciso, I.Pérez de Guzmán, *SLFD Logic: Elimination of Data Redundancy in Knowledge Representation*, LNCS, 2527: 141–150, 2002.
7. P. Cordero, M. Enciso, A. Mora, M. Ojeda-Aciego, *Computing Minimal Generators from Implications: a Logic-guided Approach*, CLA: 187–198, 2012.
8. P. Cordero, M. Enciso, A. Mora, M. Ojeda-Aciego, *Computing Left-Minimal Direct Basis of implications*. CLA: 293–298, 2013.
9. A. Eden, Y. Hirshfeld, K. Lundqvist, *EHL99 , LePUS Symbolic Logic Modeling of Object Oriented Architectures: A Case Study*, In: Proc. Second Nordic Workshop on Software Architecture (NOSA'99), 1999.
10. B. Ganter, *Two basic algorithms in concept analysis*, Technische Hochschule, Darmstadt, 1984.
11. J.L. Guigues and V. Duquenne, *Familles minimales d'implications informatives résultant d'un tableau de données binaires*, Math. Sci. Humaines: 95, 5–18, 1986.
12. A. Mora, M. Enciso, P. Cordero, and I. Fortes, *Closure via functional dependence simplification*, I. J.of Computer Mathematics, 89(4): 510–526, 2012.
13. A. Mora, M. Enciso, P. Cordero, and I. Pérez de Guzmán, *An Efficient Preprocessing Transformation for Functional Dependencies Sets Based on the Substitution Paradigm*, LNCS, 3040: 136–146, 2004.

Ordering objects via attribute preferences

Inma P. Cabrera[1], Manuel Ojeda-Aciego[1], and Jozef Pócs[2]

[1] Universidad de Málaga. Andalucía Tech. Spain*
[2] Palacký University, Olomouc, Czech Republic, and
Slovak Academy of Sciences, Košice, Slovakia**

Abstract. We apply recent results on the construction of suitable orderings for the existence of right adjoint to the analysis of the following problem: given a preference ordering on the set of attributes of a given context, we seek an induced preference among the objects which is compatible with the information provided by the context.

1 Introduction

The mathematical study of preferences started almost one century ago with the works of Frisch, who was the first to write down in 1926 a mathematical model about preference relations. On the other hand, the study of adjoints was initiated in the mid of past century, with works by Ore in 1944 (in the framework of lattices and Galois connections) and Kan in 1958 (in the framework of category theory and adjunctions). The most recent of the three theories considered in this work is that of Formal Concept Analysis (FCA), which was initiated in the early 1980s by Ganter and Wille, as a kind of applied lattice theory.

Nowadays FCA has become an important research topic in which a, still growing, pure mathematical machinery has expanded to cover a big range of applications. A number of results are published yearly on very diverse topics such as data mining, semantic web, chemistry, biology or even linguistics.

The first basic notion of FCA is that of a *formal context*, which can be seen as a triple consisting of an initial set of formal objects $\mathcal{B}$, a set of formal attributes $\mathcal{A}$, and an incidence relation $I \subseteq \mathcal{B} \times \mathcal{A}$ indicating which object has which attribute. Every context induces a lattice of formal concepts, which are pairs of subsets of objects and attributes, respectively called *extent* and *intent*, where the extent of a concept contains all the objects shared by the attributes from its intent and vice versa.

Given a preference ordering among the attributes of a context, our contribution in this work focuses on obtaining an induced ordering on the set of objects which, in some sense, is compatible with the context.

After browsing the literature, we have found just a few papers dealing simultaneously with FCA and preferences, but their focus and scope are substantially

* Partially supported by Spanish Ministry of Science and FEDER funds through projects TIN2011-28084 and TIN12-39353-C04-01.
** Partially supported by ESF Fund CZ.1.07/2.3.00/30.0041.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

different to ours. For instance, Obiedkov [11] considered some types of preference grounded on preference logics, proposed their interpretation in terms of formal concept analysis, and provided inference systems for them, studying as well their relation to implications. Later, in [12], he presented a context-based semantics for parameterized *ceteris paribus* preferences over subsets of attributes (preferences which are only required to hold when the alternatives being compared agree on a specified subset of attributes).

Other approaches to preference handling are related to the development of recommender systems. For instance, [8] proposes a novel recommendation model based on the synergistic use of knowledge from a repository which includes the users behavior and items properties. The candidate recommendation set is constructed by using FCA and extended inference rules.

Finally, another set of references deal with extensions of FCA, either to the fuzzy or multi-adjoint case, or to the rough case. For instance, in [2] an approach can be found in which, based on transaction cost analysis, the authors explore the customers' loyalty to either the financial companies or the company financial agents with whom they have established relationship. In a pre-processing stage, factor analysis is used to choose variables, and rough set theory to construct the decision rules; FCA is applied in the post-processing stage from these suitable rules to explore the attribute relationship and the most important factors affecting the preference of customers for deciding whether to choose companies or agents.

Glodeanu has recently proposed in [6] a new method for modelling users' preferences on attributes that contain more than one trait. The modelling of preferences is done within the framework of Formal Fuzzy Concept Analysis, specifically using hedges to decrease the size of the resulting concept lattice as presented in [1].

An alternative generalization which, among other features, allows for specifying preferences in an easy way, is that of multi-adjoint FCA [9,10]. The main idea underlying this approach is to allow to use several adjoint pairs in the definition of the fuzzy concept-forming operators. Should one be interested in certain subset(s) of attributes (or objects), the only required setting is to declare a specific adjoint pair to be used in the computation with values within each subset of preferred items.

The combination of the two last approaches, namely, fuzzy FCA with hedges and the multi-adjoint approach have been recently studied in [7], providing new means to decrease the size of the resulting concept lattices.

This work can be seen as a position paper towards the combination of recent results on the existence of right adjoint for a mapping $f\colon \langle X, \leq_X \rangle \to Y$ from a partially ordered set X to an unstructured set Y, with Formal Concept Analysis, and with the generation of preference orderings.

The structure of this work is the following: in Section 2, the preliminary results related to attribute preferences and the characterization of existence of right adjoint to a mapping from a poset to an unstructured codomain are presented; then, in Section 3 the two approaches above are merged together in order

to produce a method to induce an ordering among the objects in terms of a given preference ordering on attributes and a formal context.

2 Preliminaries

2.1 Preference relations and lectic order on the powerset

We recall the definition of a (total) preference ordering and describe an induced ordering on the corresponding powerset.

In the general approach to preferences, a *preference relation* on a nonempty set A is said to be a binary relation $\preceq \subseteq A \times A$ which is reflexive ($\forall a \in A, a \preceq a$) and total ($\forall a, b \in A, (a \preceq b) \vee (b \preceq a)$).

In this paper, we will consider a simpler notion, in which a preference relation is modeled by a total ordering. Formally, by a *total preference relation* we understand any total ordering of the set A, i.e., a binary relation $\preceq \subseteq A \times A$ such that $\preceq$ is total, reflexive, antisymmetric ($\forall a, b \in A, a \preceq b$ and $b \preceq a$ implies $a = b$), and transitive ($\forall a, b, c \in A, a \preceq b$ and $b \preceq c$ implies $a \preceq c$).

Any total preference relation on a set A induces a total ordering on the powerset 2^A in a natural way.

Definition 1. *Let $\langle A, \preceq \rangle$ be a nonempty set with a total preference relation. A subset X is said to be lectically smaller than a subset Y, denoted $X <_{lec} Y$, if*

$$\max\big((X \smallsetminus Y) \cup (Y \smallsetminus X)\big) \in Y.$$

If $X <_{lec} Y$ or $X = Y$ we will write $X \leq_{lec} Y$.

It is not difficult to show that the set 2^A with the lectic order forms a totally ordered set.

2.2 Building right adjoints

We assume basic knowledge of the properties and constructions related to partially ordered sets.

As we are including the necessary definitions for the development of the construction of adjunctions, we state below the notion of adjunction we will be working with.

Definition 2. *Let $\mathbb{A} = \langle A, \leq_A \rangle$ and $\mathbb{B} = \langle B, \leq_B \rangle$ be posets, $f\colon A \to B$ and $g\colon B \to A$ be two mappings. The pair (f, g) is said to be an* adjunction between $\mathbb{A}$ and $\mathbb{B}$, *denoted by $(f, g)\colon \mathbb{A} \leftrightharpoons \mathbb{B}$, whenever for all $a \in A$ and $b \in B$ we have*

$$f(a) \leq_B b \qquad \text{if and only if} \qquad a \leq_A g(b).$$

The mapping f is called left adjoint *and g is called* right adjoint.

Given a mapping from a poset $\langle A, \leq_A \rangle$ to an unstructured set B, the necessary and sufficient conditions for f to have a right adjoint were given in [5]; the idea was to build it gradually, in terms of the canonical decomposition of $f\colon A \to B$ through A_f, the quotient set[3] of A wrt the *kernel relation* $\equiv_f$, defined as $a \equiv_f b$ if and only if $f(a) = f(b)$:

$$\begin{array}{ccc} A & \xrightarrow{\ f\ } & B \\ {\scriptstyle\pi}\,\vdots & & \vdots\,{\scriptstyle i} \\ A_f & \xrightarrow{\ \varphi\ } & f(A) \end{array}$$

where mapping π is the canonical projection onto the quotient set A_f, defined by $\pi(a) = [a]_f$, φ is the canonical isomorphism of the quotient and the image, defined by $\varphi([a]_f) = f(a)$, and i is the inclusion of the image into the codomain.

The obtained characterization is recalled in the theorem below.

Theorem 1. *Given a poset $\mathbb{A} = \langle A, \leq_A \rangle$ and a mapping $f\colon A \to B$, let $\equiv_f$ be the kernel relation. Then, there exists a poset structure on B, say $\mathbb{B} = \langle B, \leq_B \rangle$, and a mapping $g\colon B \to A$ such that $(f, g)\colon \mathbb{A} \leftrightharpoons \mathbb{B}$ if and only if*

1. *There exists* $\max([a]_f)$ *for all* $a \in A$.
2. *For all* $a_1, a_2 \in A$, $a_1 \leq_A a_2$ *implies* $\max([a_1]_f) \leq_A \max([a_2]_f)$.

If the conditions hold, a suitable ordering on the image of f (that can also be extended to B) can be defined as follows:

$$\begin{array}{ll} b_1 \leq_B b_2 & \text{if and only if} \\ & \quad \text{there exist} \quad a_1 \in f^{-1}(b_1), a_2 \in f^{-1}(b_2) \\ & \qquad\qquad \text{such that} \ \max([a_1]_f) \leq_A \max([a_2]_f). \end{array}$$

It is worth to notice that the theorem above can be easily adapted to characterize existence of Galois connections.

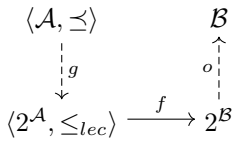
3 Inducing preferences

Given the results introduced in the previous section, here we will merge them so that, given a preference relation on the set of attributes $\mathcal{A}$, an induced ordering is obtained on the set of objects $\mathcal{B}$.

In order to simplify the presentation and minimize technicalities, we will consider a crisp context $\mathbb{C} = (\mathcal{B}, \mathcal{A}, I)$ and a total preference ordering on the set of attributes, say $\langle \mathcal{A}, \preceq \rangle$.

[3] The equivalence class of a under the kernel relation $\equiv_f$ will be denoted as $[a]_f$.

The general idea can be depicted as the diagram below



each of the three stages is explained as follows:

1. To begin with, the preference on attributes allows for generating[4] the corresponding lectic order on $\langle 2^{\mathcal{A}}, \leq_{lec}\rangle$.
2. On this lattice, the usual concept-forming operator f can be defined, see [4], from $\langle 2^{\mathcal{A}}, \leq_{lec}\rangle$ to the (unstructured) powerset of objects of $\mathcal{B}$. Namely, given $A \in 2^{\mathcal{A}}$ we define

$$f(A) = \{b \in \mathcal{B} \mid (b, a) \in I \text{ for all } a \in A\}.$$

   Now, under suitable conditions as stated in [5], there exists an ordering on $2^{\mathcal{B}}$ such that a right adjoint for f exists.
3. Finally, this ordering is projected down to $\mathcal{B}$ to obtain an induced ordering among all the objects.

Summarizing, given a preference ordering of the set of attributes $\langle \mathcal{A}, \preceq\rangle$ and a context, an induced ordering on the set of objects $\mathcal{B}$ is obtained, which is compatible with the context.

It is worth to note that, by considering the inclusion ordering on $2^{\mathcal{A}}$, the inclusion ordering on $2^{\mathcal{B}}$ and the (other) standard concept-forming operator forms a Galois connection, hence the inverse inclusion ordering leads to an adjunction. This means that the proposed approach, in a certain sense, generalizes the standard concept-forming approach.

Some illustrative examples

To begin with, Theorem 1 characterizes when an ordering can be induced in the codomain B so that a right adjoint to a given mapping $f\colon \langle A, \leq_A\rangle \to B$ exists. It is not difficult to find examples in which that situation does not hold.

Example 1. Consider the context $\mathbb{C} = (\{o_1, o_2, o_3\}, \{a_1, a_2\}, I)$, where the incidence relation I is defined as in the left of Figure 1. In addition, consider that attribute a_1 is more preferred than a_2 (which we denote $a_1 \succ a_2$).

For this context it is clear that Property 2 of Theorem 1 does not hold in general. Specifically, if we consider (singleton) sets $A_1 = \{a_1\}$ and $A_2 = \{a_2\}$, then we have $A_2 \leq_{lec} A_1$ but, clearly, $\max[A_2]_f \not\leq_{lec} \max[A_1]_f$, i.e. $A_2'' \not\leq_{lec} A_1''$, since $A_2'' = \{a_1, a_2\}$, whereas $A_1'' = \{a_1\}$.[5]

[4] Hence the g, but notice that this is just a notation, not an actual mapping from $\mathcal{A}$ to $2^{\mathcal{A}}$.

[5] See Section 3.1.

	a_1	a_2
o_1	×	
o_2	×	
o_3	×	×

	a_1	a_2
o_1		×
o_2	×	
o_3	×	×

Fig. 1.

Example 2. Consider an alternative incidence relation defined as in the right of Figure 1. Again, consider that attribute a_1 is more preferred than a_2.

For this alternative context, the previous problem does not arise, since $A_2'' = \{a_2\}$ and $A_1'' = \{a_1\}$. Therefore, an ordering on $2^{\mathcal{B}}$ can be given which, when projected on the set of objects $\mathcal{B}$, leads to $o_1 \leq o_2 \leq o_3$.

The obtained result is compatible with the information given by the incidence relation, in that o_3 has more preferred attributes than o_2 and so on. Anyway, the existence of situations in which it is not possible to induce an ordering on B leads to the more general problem of studying conditions on the context which guarantee its existence.

To begin with, property 1 automatically holds in our approach; the details are given below.

3.1 Checking Property 1

Property 1, i.e. $\max([A]_f)$ exists for all A, always holds in this framework due to the particular definition of f as the standard concept-forming operator.

In effect, given $A \in 2^{\mathcal{A}}$, the equivalence class $[A]_f$ consists of sets of attributes whose image coincides with that of A, this is independent from the particular ordering chosen in $2^{\mathcal{A}}$.

We know that, under the inclusion ordering, the closure of A, denoted A'', is the maximum of $[A]_f$: i.e. $A_i \subseteq A''$ for all $A_i \in [A]_f$. Furthermore, as the inclusion ordering implies lectic ordering we have that $A_i \leq_{lec} A''$ for all $A_i \in [A]_f$, which states that A'' is also the maximum of $[A]_f$ in the chain $\langle 2^{\mathcal{A}}, \leq_{lec}\rangle$.

3.2 Checking Property 2 (a first approach to its complexity)

As shown in the previous examples, property 2 does not always hold.

A first naive step would be simply checking Property 2 in all the pairs of subsets $A_1 \leq_{lec} A_2$. Fortunately, not all of them have to be checked since the lectic ordering contains the inclusion ordering and, for this ordering the property holds (this is just a consequence of the fact that the usual concept-forming operators form a Galois connection), but there are other possibilities to be taken into account, which are pairs of sets of attributes satisfying $A_1 \leq_{lec} A_2$ but $A_1 \not\subseteq A_2$.

Specifically, in order to study the complexity of checking property 2 (by brute force) we have firstly to solve the following

> **Problem:** Given an ordered set $\mathcal{A} = \{a_1, \ldots, a_n\}$ with n elements, we want to count the pairs of subsets A_1 and A_2 such that A_1 is lectically less than A_2 wrt the ordering given by the subscripts of the elements in A, but is NOT included in it.

because those are the cases in which the property is not known to hold and, hence, are called *problematic pairs*.

For the computation, we will interpret a subset as a chain $[d_1, \ldots, d_n]$ of n digits, indicating membership or not to the subset.

The key idea for counting the number of problematic pairs is related to two important places in the chain, for which we introduce a special notation:

1. Digit d_ℓ represents the first attribute a_ℓ which is in A_2 but not in A_1 (the ℓ should recall the first ℓectic discrepancy).
2. Digit d_i represents the first attribute a_i which is in A_1 but not in A_2, that is, the first discrepancy wrt the $i$nclusion ordering.

It is obvious that, in any given pair of subsets, d_ℓ is more preferred, i.e. occurs before, than d_i.

Now, we can state that every attribute more preferred than a_i, except a_ℓ, either belongs to both sets or does not belong to any of them; so in every such position only two possibilities arise (either two 1s or two 0s), this means that a factor 2 is associated to any such digit. In addition, there is no restriction for attributes less preferred than a_i, that is, in every such position four possibilities can occur, and this means that a factor 4 is associated to any such digit.

In order to see the general pattern of possible cases, let us consider a set with four attributes, so $n = 4$. There are three possible positions for d_i, namely, second, third and fourth, which are handled separately.

***i*-discrepancy in 4th digit** In this case, d_ℓ can be in any of the three first places, and the remaining two positions should have coincident values. Then, there are $3 \cdot 2^2$ possibilities.

***i*-discrepancy in 3th digit** Now, there are only two possible positions for d_ℓ, and the remaining one should have coincident values (so two possibilities). In the last digit there is no restriction (4 possibilities). All in all, there are $2 \cdot 2^1 \cdot 4$ cases.

***i*-discrepancy in 2nd digit** Then d_ℓ should be the first one. There are two digits with no restriction, so 4^2 cases.

Summarizing, we have $3 \cdot 2^2 \cdot 4^0 + 3 \cdot 2^1 \cdot 4^1 + 1 \cdot 2^0 \cdot 4^2$ possibilities.

The previous example shows a clear pattern by which the number of problematic cases for n attributes is given by the following expression

$$(n-1) \cdot 2^{n-2} \cdot 4^0 + (n-2) \cdot 2^{n-3} \cdot 4^1 + \cdots + 2 \cdot 2^1 \cdot 4^{n-3} + 1 \cdot 2^0 \cdot 4^{n-2}$$

or, in compressed form as $\displaystyle\sum_{k=1}^{n-1} (n-k) 2^{n+k-3}$.

This sum can be expressed in closed form as follows

$$\begin{aligned}\sum_{k=1}^{n-1}(n-k)2^{n+k-3} &= 2^{n-3}\sum_{k=1}^{n-1}(n-k)2^k \\ &= 2^{n-3}\left(\sum_{k=1}^{n-1}n2^k - \sum_{k=1}^{n-1}k2^k\right) \\ &= 2^{n-3}\left(n\sum_{k=1}^{n-1}2^k - \sum_{k=1}^{n-1}k2^k\right) \\ &\vdots \quad \text{sums of (arithmetic-)geometric progressions} \\ &= 2^{2n-2} - n2^{n-2} - 2^{n-2}\end{aligned}$$

It is clear that, as there are 2^{2n} possible pairs of subsets, the ratio between the number of problematic pairs and the number of possible pairs tends asymptotically to 1/4.

4 Seeking sufficient conditions on the context

We have just seen that checking Property 2 by brute force on every problematic pair has exponential complexity on the size of the set of attributes, therefore we introduce in this section some possible ways to establish sufficient conditions on the context so that the proposed approach can be applied to define an induced ordering on the set of objects.

To begin with, the following result states a partial sufficient condition.

Proposition 1. *Consider a context such that the following conditions hold for all pair of attributes satisfying $b \prec a$ (a is more preferred than b):*

1. *There exists an object in which b holds but a does not.*
2. *Whenever b implies[6] a less preferred attribute, say c, then a implies c as well.*

Then property 2 holds for all the singleton sets of attributes.

Proof. Consider $a_k \prec a_j$, and the singletons $A_k = \{a_k\}$ and $A_j = \{a_j\}$. It is clear that $A_k \leq_{lec} A_j$, therefore we have to show that $A_k'' \leq_{lec} A_j''$.

By contradiction, assume that $A_k'' \not\leq_{lec} A_j''$. This means that the most preferred discrepant attribute between both closures, say a_d, is in A_k'' and not in A_j'', i.e. $a_d \in A_k'' \smallsetminus A_j''$.

We will reason by cases, according to the relative position of a_k and a_d wrt the preference ordering.

[6] As the usual implication of attributes.

- It cannot occur that attribute a_d is more preferred than a_k, since in that case a_d should hold in *every* object satisfying a_k, since $a_d \in A_k'' \smallsetminus A_j''$, contradicting the first hypothesis.
- Furthermore, it cannot be the case that $a_d \prec a_k$ either. We have once again that a_d (now less preferred than a_k) holds in very object satisfying a_k. Now, the second hypothesis states that a_d is also implied by a_j and, that is $a_d \in A_j''$ which contradicts that $a_d \in A_k'' \smallsetminus A_j''$.
- Finally, the case $a_d = a_k$ means that a_j is not a discrepant attribute; now, as it is the case that $a_j \in A_j''$, it should happen that $a_j \in A_k''$ and, hence, a_k should imply a_j, violating hypothesis 1.

□

Obviously, this proposition alone does not imply the fulfillment of property 2, but gives a clue of a general sufficient condition, albeit too strong, which is stated below:

Proposition 2. *Consider a context with a preference ordering such that, for all subset of attributes A, satisfying the following properties:*

1. *There exists an object failing to satisfy the most preferred attribute in A, but satisfying all the other attributes.*
2. *Whenever A implies an attribute, say a_d, then any other subset of attributes, more preferred in the lectic order, implies a_d as well.*

Then property 2 holds.

It is worth to introduce some comments on the conditions used in the previous proposition.

To begin with, the first condition makes sense: as we wish to establish an ordering on the objects, according to a prescribed order of preference among the attributes and information in the context, the not-so-trivial cases are precisely those containing objects failing to satisfy most preferred attributes, but satisfying several less preferred ones. Otherwise, the user should not need any formal tool to choose according to his/her preferences.

Specifically, consider a context containing lines as in Figure 2, again assuming a_1 more preferred than a_2 more preferred than a_3. In such a case, it might not be clear whether to choose car_1 because it satisfies the most preferred attribute (being cheap, but without safety measures like ABS or airbag, and without the comfort of an air-conditioned system) or car_2, which is not cheap but includes safety and comfort measures.

The second hypothesis is reasonable as well, since it somehow implies the coherence of the preference ordering between attributes. It is worth to remark that it is not just a technical requirement which can be avoided by considering contexts without any implied attributes because in practical situations it can make sense to admit certain implications. For instance, back to the previous example of cars and its attributes, it might be convenient consider simultaneously the attributes Automatic Climate Control (ACC) and Air Conditioned (A/C) since, although ACC always implies A/C, it could be the case that a user would be satisfied just with a basic A/C system.

	$Cheap$	ABS	$Airbag$	A/C	ACC
⋮					
car_1	×				
⋮					
car_2		×	×	×	
⋮					

Fig. 2.

5 Conclusions and future work

In this paper, we have sketched a method for inducing an ordering in the set of objects from a preference ordering in the set of attributes which is based on the machinery of FCA and recent results concerning the existence of right adjoints to a given mapping. Some sufficient conditions have been given in order to guarantee that the proposed framework can be applied to a given context.

The problem has been stated in its simplest version, with a specification of preferences as a total ordering, and considering a crisp context. We have just started to scratch the surface of the problem and, to be honest, there is much more work to be done than contributed results to the topic presented in this paper.

We enumerate below a number of possible alternatives to be developed in the short and mid term:

1. Obtain weaker sufficient conditions for property 2 to hold and, if possible, characterize those contexts for which property 2 automatically holds. For this characterisation it seems crucial to obtain information about the greatest discrepant attribute of two given closed sets of attributes.
2. Consider general preference relations (reflexive and total) or even other approaches to the notion of preference, see [3].
3. Consider preference relations which allow to assign weights to each attribute, so that the comparison between objects satisfying different sets of attributes can be made more in consonance with the user.
4. The previous item naturally leads to the consideration of one-sided concept lattices, in which it is possible to specify that objects satisfy attributes only to a certain degree.

   ...

References

1. R. Belohlavek and V. Vychodil. Formal concept analysis and linguistic hedges. *Int. J. General Systems*, 41(5):503–532, 2012.
2. S.-K. Fang, J.-Y. Shyng, W.-S. Lee, and G.-H. Tzeng. Exploring the preference of customers between financial companies and agents based on TCA. *Knowledge-Based Systems*, 27:137–151, 2012.

3. P. Fishburn. Preference structures and their numerical representations. *Theoretical Computer Science*, 217:359–383, 1999.
4. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundation*. Springer Verlag, 1999.
5. F. García-Pardo, I. Cabrera, P. Cordero, M. Ojeda-Aciego, and F. Rodríguez. On the definition of suitable orderings to generate adjunctions over an unstructured codomain. *Information Sciences*, 286:173–187, 2014.
6. C. V. Glodeanu. Exploring users' preferences in a fuzzy setting. *Electronic Notes in Theoretical Computer Science*, 303:37–57, 2014. Proceedings of the Workshop on Algebra, Coalgebra and Topology (WACT 2013).
7. J. Konecny, J. Medina, and M. Ojeda-Aciego. Multi-adjoint concept lattices with heterogeneous conjunctors and hedges. *Annals of Mathematics and Artificial Intelligence*, 2014. Accepted. http://dx.doi.org/10.1007/s10472-014-9405-y.
8. X. Li and T. Murata. A knowledge-based recommendation model utilizing formal concept analysis and association. In *2nd Intl Conf on Computer and Automation Engineering (ICCAE)*, pages 221–226, 2010.
9. J. Medina and M. Ojeda-Aciego. Multi-adjoint t-concept lattices. *Information Sciences*, 180(5):712–725, 2010.
10. J. Medina, M. Ojeda-Aciego, and J. Ruiz-Calviño. Formal concept analysis via multi-adjoint concept lattices. *Fuzzy Sets and Systems*, 160(2):130–144, 2009.
11. S. Obiedkov. Modeling preferences over attribute sets in formal concept analysis. In F. Domenach, D. Ignatov, and J. Poelmans, editors, *Formal Concept Analysis*, volume 7278 of *Lecture Notes in Computer Science*, pages 227–243. Springer Berlin Heidelberg, 2012.
12. S. Obiedkov. Modeling ceteris paribus preferences in formal concept analysis. In P. Cellier, F. Distel, and B. Ganter, editors, *Formal Concept Analysis*, volume 7880 of *Lecture Notes in Computer Science*, pages 188–202. Springer Berlin Heidelberg, 2013.

DFSP: A new algorithm for a swift computation of formal concept set stability

Ilyes DIMASSI, Amira MOUAKHER and Sadok BEN YAHIA

University of Tunis El Manar, LIPAH, Faculty of Sciences of Tunis, Tunis, Tunisia.
sadok.benyahia@fst.rnu.tn

Abstract. Concept lattices are very useful for the task of knowledge discovery in databases. However, the overwhelming number of drawn formal concepts was always an actual hamper towards their effective use. In the aim of filtering out, such endless lists of formal concepts, the stability metric is the most worth of mention one. In this respect, the stability computation of large concepts has been shown to be infeasible due to exponential number of object sets to be processed. The literature only witnesses approaches for the stability computation that heavily rely on the existence of the Galois lattice. In this paper, we introduce a new efficient algorithm, called DFSP, for computing the stability of a set of formal concepts without having at hand the underlying partial relation. The main thrust of the introduced algorithm stands in the smart detection of non generators and their pruning owe to their fulfilment of monotony property within a given equivalence class. To the best of our knowledge, DFSP is the first algorithm that tackled such tough issue. Carried out experiments showed that DFSP efficiently computes the scalability of very large formal concepts extracted from benchmark datasets of the Data Mining field.
Keywords: Formal concept analysis, stability, generators,pruning, tidset.

1 Introduction and motivations

Concept lattices are very useful for the task of knowledge discovery in databases. However, this field is hampered by the overwhelming size of formal concept lists drawn from even moderately sized contexts. In this respect, the stability index has been shown to be efficient for throwing away "bad" concepts. Nevertheless, the computation of such stability index is very consuming and has been shown to be NP-Complete task. Thus, the FCA community paid much attention to the computation of exact and/or approximative of the stability as could witness the recent publications on such issue [1, 2]. At a glance, the recent proposals of approximations of stability computations unveil the actual complexity of such a task. Roughly speaking, the computation of the stability of a concept $C = (A, B)$ comes back to the exploration to a huge space made of the power set of the extension part. In this space, we have to record all the elements in snugness connection with the corresponding part. Clearly, even for an extent with dozens of objects, it does not exit a primitive type for storing the value of a stability[1].

At a glance, the related work flags out approaches that compute the stability of a set of formal concepts organized through the Galois lattice. Doing so, they start computing

[1] The GMP library, https://gmplib.org/, could be of use, in such a case, to store huge values.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

the stability of smallest formal concept (in extent's size terms) and exploit this result for the subsumer concepts until reaching the top formal concept.

In this paper, we introduce a new algorithm, called DFSP, that aims to an efficient straightforward computation of a set of formal concepts. The main thrust of the introduced algorithm stands in the smart detection of non generators and their pruning owe to their fulfilment of monotony property within a given equivalence class. Indeed, we introduce the notion of saturation of non-generators through the detection of the maximal set of a non-generators. Given that each subset of a non generator is also a non generator, the DFSP algorithms sweeps the search space in depth first manner and only stresses on the generators by avoid squandering its efforts on useless non-generators subspaces.

The carried out experiments highlight that DFSP easily handles formal concepts having thousands of objects in their extent part. To the best of our knowledge, DFSP is the first algorithm that handles efficiently and straightforwardly formal concepts for the stability computation.

The remainder of the paper is organized as follows: The next section recalls key notions used throughout this paper as well as the pioneering approaches of the related work. Then, we present in section 3 our algorithm for computing the stability of a set of formal concepts, called DFSP. Section 4 describes the experimental study and the results we obtained. Section 5 concludes the paper and points out our future work.

2 Stability computation: Scrutiny of the Related work

Before scrutinizing the related work that paid attention the stability computation, we provide a simplified definition of some concepts used throughout in this paper, by supposing that the the reader is familiar with FCA basic settings.

Definition 1. (MONOTONIC / ANTI-MONOTONIC CONSTRAINT) *Let Q be a constraint,*

- *Q is anti-monotonic if $\forall I \subseteq \mathcal{I}$, $\forall I_1 \subseteq I$: I fulfils $Q \Rightarrow I_1$ fulfils Q.*
- *Q is monotonic if $\forall\ I \subseteq \mathcal{I}$, $\forall\ I_1 \supseteq I$: I fulfils $Q \Rightarrow I_1$ fulfils Q.*

Definition 2. (EQUIVALENCE CLASS) *An equivalence class is a set of itemsets with same closure (and same image). Let C=(A, B) be a formal concept, for any subset $X \subseteq \mathcal{O}$, $A = X''$ is the largest tidset w.r.t. set inclusion in its equivalence class. Precisely, $A \subseteq \mathcal{O}$ is closed iff $\nexists X$ such as $X \subset A$ with $X' = A'$; $X \subseteq \mathcal{O}$ is a generator iff $\nexists U \subset X$ with $U' = X'$. $G_A = \{X \subseteq A | X' = B\}$ is the set of all generators in the equivalence class.*

Definition 3. (EXTENT FULL SPACE) *Let $\mathcal{K}$ = ($\mathcal{O}$, $\mathcal{I}$, $\mathcal{R}$) be a formal context, $\mathcal{B}(\mathcal{K})$ its concept lattice and $C = (A, B)$ a concept from $\mathcal{B}(\mathcal{K})$ where $|A| = n$. $\mathcal{P}(A) = \{X | X \subseteq A\}$ is the set of all A's subsets and $\mathcal{P}(A) = 2^{|A|}$.*

Stability has been introduced probably for the first time by Kuznetsov [3] and later revisited in [4, 5]. This measure seems to be the most widely used around the FCA community and is applied in numerous applications [6], e.g. biclustering, detection of scientific subcommunities, to cite but a few. Formally, it is defined as follows:

Definition 4. (STABILITY) *The stability index of a given concept describes the proportion of subsets of its objects whose closure is equal to the intent of this concept. This metric reflects the dependency of the intent on particular objects of the extent [4]. Let $\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$ be a formal context and $C = (A, B)$ a formal concept of $\mathcal{K}$. The stability index, σ, of C is defined as follows:*

$$\sigma(A, B) = \frac{|\{C \subseteq A | C' = B\}|}{2^{|A|}} = \frac{|G_A|}{2^{|A|}} \tag{1}$$

The authors of [7] highlighted that a concept that covers fewer objects is normally less stable than do a concept covering more objects.

However, the main hamper towards its intensive use mainly for large datasets, is the complexity of its computation. In fact, it's shown to be a #P-complete problem [3, 5]. In order to compute it for large concept lattices, several works proposed to use estimates and approximations however others tried to find an exact solution using the concept lattice in computing stability.

Roth et al. [8] paid attention to concept's stability as well as other metrics to reduce the size of large concept lattice. They proposed an exact and polynomial algorithm COMPUTESTABILITY that computes the stability indices for every concept of the lattice using the covering graph of a concept lattice. The algorithm traverses the covering graph from the bottom concept upwards. A concept is processed only after the stability indices of all its sub-concepts have been computed. The main drawback of this algorithm that it is essentially quadratic in the number of concepts in the lattice, which may be prohibitively expensive for large lattices. In addition, this algorithm inputs a Galois lattice and such a requirement could not be easily available for very large datasets may be impractical.

Kuznetsov [5] introduced a polynomial algorithm for computing stability using various methods of estimating scientific hypotheses. This algorithm is considered as optimal in the sense that its time complexity is linear and polynomial in the size of the context. Nevertheless, this approach only gives an approximate assessment about the stability index and could not be efficient in exact studies.

Later, Jay et al. [9] applied the concept of stability and iceberg lattices in social network analysis. They used the stability metric to reduce the complexity of the lattice, by filtering out all unstable concepts w.r.t. a given threshold. In this respect, the authors introduced a new definition of the stability using the equivalence classes. Given a concept (A, B), the stability metric measures the number of elements of G that are in the same equivalence class of A where an equivalence class is defined as follows:

Property 1. Using Definition 4, the authors proved the following property:

$$\sigma(A, B) = 1 - \sum_{X \subset A, X = X''} \sigma(X, X') 2^{|X| - |A|}$$

So, once the lattice concept is given, it is possible to compute quickly the stability of concepts using an ascending algorithm.

Roth et al. [10] proposed an algorithm, based on a polynomial heuristic for computing stability index for all concepts using the concept lattice. This algorithm was quite

good in practical applications so far, but in the worst case its complexity is quadratic in the size of the lattice.

Later, Babin and Kuznetsov [1] also suggested a method for approximating concept stability based on a Monte Carlo approach. Their approximate algorithm can run in reasonable time. In their approach, they specified a new parameter called *stability threshold* to reduce the number of the concepts. The results show that the approximations are better when stability threshold is low.

Recently, Buzmakov et al. [2] introduced an efficient way for finding a "good" assessment of concept stability. The authors combined the *bounding method* [9] as well as the *Monte Carlo method* [1] in a complementary way. Once the stability bounds are computed in the lattice, the method that should be applied is chosen according to the most tight of each them.

The main criticism that can be made about the literature's approaches stands in the fact that they are unable to compute stability of concepts in absence of order relation. In fact, the lattice structure is a *sine qua non* condition to proceed with the computation. Beside that, such computation of concept's stability requires visiting all its sub-concepts, i.e., direct and non direct ones. Clearly, doing so is very greedy in computations and memory usage. Nevertheless, building concept lattice is very far from being an easy task [11] and sometimes impossible. Furthermore, the cost of generating a lattice concept remains high as far as the context is composed of a large number of objects [12] and/or a dense incidence relation.

Thus, we introduce a new efficient algorithm, called DFSP that allows computing the stability for a given set of concepts. The latter doesn't need any partial relation between the concepts. The main thrust of the introduced algorithm stands in the smart detection of non generators and their pruning owe to their fulfilment of monotony property within a given equivalence class.

3 DFSP : Depth First Stability Processor algorithm

Before, thoroughly describing the DFSP algorithm, we start by introducing some useful notations that are used in the remainder.

Let $C = (A, B)$ be a formal concept for which the stability index needs to be processed. DFSP algorithm organises $\mathcal{P}(A)$ according to a tidset prefix tree. Each exploration node in the tree is specific to a tidset and represented by the TSNode data structure. TSNode is a recursive structure that keeps track of useful informations about its associated tidset such as its suffix, itemset and a set of immediate supersets. As for the current tidset, its immediate supersets are themselves represented each by a TSNode instance and so on.

Definition 5. (SUFFIX OF TIDSET) *Let $ts = \{t_1, t_2 \dots t_k\}$ be an ordered sequence of objects and n_{ts} its associated exploration node. $Suff(ts) = t_k$ is the last object in ts. TSNode.s is the member property of the data structure TSNode in which $Suff(ts)$ is maintained ; $n_{ts}.s = t_k$.*

	a	b	c	d	e	f	g	h
1		×					×	
2	×		×				×	
3				×	×	×	×	×
4	×			×	×	×	×	×
5	×				×	×	×	×
6		×				×	×	
7	×					×	×	
8	×				×		×	
9	×	×	×	×	×	×	×	×

Table 1. Formal context

Definition 6. (CHILDREN'S NODE) *TSNode.ss is a member of the TSNode structure that holds a set of TSNode instances. TSNode.ss is the set of a TSNode immediate children.*

Definition 7. (NODE INTENT) *Let ts be a tidset and n_{ts} its associated node. TSNode.is is a TSNode member that holds the tidset image. $n_{ts}.is = ts'$.*

Unlike $n_{ts}.s$ which only holds the tidset suffix $Suff(ts)$, $n_{ts}.is$ integrally maintains ts'.

Example 1. *With respect to the formal context depicted in Table 1, we have $ts = \{123\}$ and n_{123} its associated node, then $n_{123}.s = 3$. In addition, we have : $n_{123}.ss$=$\{n_{1234}$, n_{1235}, n_{1236}, n_{1237}, n_{1238}, $n_{1239}\}$. Besides, we also have $n_{123}.is = ts' = \{123\}' = \{g\}$.*

In the following, we present a detailed description of DFSP algorithm. Let us remind that the main idea of our new approach is to provide a simple and very efficient strategy for computing stability through generators enumeration. The DFSP algorithm, whose pseudo-code is sketched by Algorithm 1, operates mainly as follows:

Initially, the sizes of the extent and the intent are stored respectively into n and m (lines 2,3). The root node is then built and it's $root.s$ and $root.is$ members are set to $\emptyset$ (line 4). Then, the objects a_i of A are scanned in order to build the first level nodes. For each object a_i, a child node is created, its member $child.s$ is set to a_i and its member $child.is$ is set to the intent of a_i (line 6, 7). If the size of the intent a'_i is different from m the size of B, then child is a non generator and is added to $root.ss$ the set of the root node immediate children (c.f lines 8, 9). Then the first level generators count is determined using the generators counting formula (line 10). After that, the exploration of space of tidset through the EXPLORETIDSET function updates gc one last time to obtain the final count of generators (line 11). The stability index is determined when the generators count gc is divided by the overall tidset count (line 12).

In the following part, we will explain the fundamental step of the algorithm which is the recursive exploration of tidset's space.

Algorithm 1: Depth First Stability Processor (DFSP)

Data: **TSNode**
-**TSNode.s**: the tidset suffix
-**TSNode.is**: the tidset intent
-**TSNode.ss**: the set of immediate children nodes.
Input:
-$\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$: a formal context.
-$C = (A, B)$: a formal concept.
Results:
-$\mathcal{S}$: the stability of C.

```
1  Begin
2    n := |A|;
3    m := |B|;
4    root.i := root.is := ∅;
5    For i = 1 ... n do
6      nchild.s := a_i;
7      nchild.is := a'_i;
8      If |nchild.is|! = m then
9        root.ss∪ = nchild;
10   gc := Σ_{i=|root.ss|}^{n-1} 2^i;
11   gc := gc + EXPLORETIDSET(root.ss, K, m);
12   S := gc / 2^n;
13   Return S;
14 End
```

3.1 Depth First exploration of the Tidset space

The main goal of EXPLORETIDSET which pseudo-code is sketched through Algorithm 2 is counting generators while minimizing as much as possible the visited tidsets. This is achieved by pruning generators and most importantly by detecting "prunable" non generators. The first invocation for EXPLORETIDSET (c.f line 11 of DFSP) is applied on the root node immediate children.

The tidset space exploration pattern The exploration mechanics are straightforward. To harness this process, lets ignore any possible optimisation that leads to nodes pruning. On the first invocation of EXPLORETIDSET in DFSP, ss contains the nodes $\{n_{a1}, \ldots, n_{an}\}$ associated to unitary first level tidsets $\{a1\},\ldots,\{an\}$ for which EXPLORETIDSET builds immediate children as follows: n_{a1a2} the first immediate child of n_{a1} is built by adding the suffix of n_{a2} to n_{a1}. More generally, n_{aiaj} the j^{th} immediate child of n_{ai} (line 12) is obtained by adding to n_{ai} the suffix of $n_{a(i+j)}$ the j^{th} node following n_{ai} (line 15). For the tidset $\{aiaj\}$ associated to n_{aiaj}, only $Suff(\{aiaj\})$ is stored in $n_{aiaj}.s$ (line 16). Since, $\{aiaj\} = \{ai\}|Suff(\{aj\})$ then $\{aiaj\}' = \{ai\}' \bigcap Suff(\{aj\})'$. $\{aiaj\}'$ is stored integrally in $n_{aiaj}.is$ (line 17). After building $n_{ai}.ss$ from n_{ai} followings, EXPLORETIDSET is recursively applied on $n_{ai}.ss$ (line 22) which only makes sense when $|n_{ai}.ss|$ has at least 2 elements (line 21). After process-

ing the n_{ai} subspace, EXPLORETIDSET moves to the next node $n_{a(i+1)}$ (line 12). The last node is n_{an} is not processed as it has no followings.

Counting and pruning generators As described above, the generation process builds a child node by adding an object to its parent node. A child node is therefore always a superset of its parent. Otherwise, it is known that a superset of a generator is also a generator. Therefore, applying the exploration process on a generator node will inevitably produce generators. The exploration branch starting from that node is said to be monotonous and since we are able to count the population induced from that branch we can save processing power by dismissing these nodes (line 8 in DFSP and line 18 in EXPLORETIDSET). Let's find out how it is possible to count generators that are inferred from a given generator without the need of exploring them. Let n_i be the i^{th} node in ss and n_i is a generator. Building the n_i subspace by exploring its immediate children then its children's children and so on recursively is equivalent to generating all possible supersets of the tidset associated to n_i using the suffixes of n_i following nodes $\{n_{(i+1)}.s,\ldots,n_{(i+1)}.s\}$. The count of all generators in n_i branch (including even n_i) is equal $2^{(n-i-1)}$.

We have to also consider supersets of n_i that are not part of n_i branch but rather in n_k branches $\{k >= 1 \text{ and } k < i\}$ the branches of all nodes that precedes n_i in ss. To avoid locating these nodes and simplify calculations, let's virtually move n_i to the start of ss. All supersets of n_i are now confined in n_i branch and the updated count of n_i supersets is $2^{(n-1)}$. Let n_j be another generator in the same cluster. It is important to count all n_j supersets while avoid including elements that are already counted as part of the n_i branch. By virtually moving n_j after n_i in ss and counting all elements in the n_j branch, it is possible to fulfil both conditions. n_j branch count is $2^{(n-2)}$ and the same process is applied to the remaining generators in the cluster. Doing so leads us to the generalized generators counting formula $gc = \sum_{k=|ss|-1}^{|gs|} 2^k$ where $|gs|$ is the generators count and $|ss|$ is cluster size (line 10 in DFSP and line 20 in EXPLORETIDSET).

Detecting non generators monotony The most significant mop up mechanism in DFSP is non generators pruning. In order to also eliminate non generators, EXPLORETIDSET looks for nodes in ss that when combined together, the resulting clique superset will still be a non generator. Those nodes are said to form a *non generator monotonous clique*. Suppose, we're building the branch of a node from this clique. If we use exclusively nodes from the clique, all nodes in the branch are guaranteed to be subsets of the clique superset. Since a subset of a non generator is also a non generator then all branches in the clique will only contain non generators. Nodes in the clique are pushed at the end of the ss set to insure that the generation process will only use nodes from the clique. Nodes outside the clique are moved away to the beginning of ss. Nodes in the clique are not expanded, since no generator could be found in their branches but are still used to build branches outside the clique.

Algorithm 2: EXPLORETIDSET

Input:
-$\mathcal{K} = (\mathcal{O}, \mathcal{I}, \mathcal{R})$: a formal context.
-ss=a set of TSNode siblings.
-m: the size of the intent.
Results:
-gc : the generators count.

Begin
  $i := ssc := |ss|$;
  $ingpc := gc := 0$;
  $ingpi := \mathcal{I}$;
  While $i < 1$ **do**
    If $|ngpi \cap ss[i].is| = m$ **then**
      MOVETOHEAD(i, ss);
      $ingpc := ingpc + 1$;
    Else
      $i := i - 1$;
      $ngpi := ngpi \cap ss[i].is$;
  For $(i = 1 \ldots ingpc)$ **do**
    $nleft := ss[i]$;
    For $(j = i + 1 \ldots ssc)$ **do**
      $nright := ss[j]$;
      $nchild.s := nright.s$;
      $nchild.is := nleft.is \cap f(nchild.s)$;
      If $(|nchild.is|! = m)$ **then**
        $nleft.ss \cup := nchild$;
    $gc+ = \sum_{k=|nleft.ss|}^{ssc-i-1} 2^k$;
    If $(|nleft.ss| > 1)$ **then**
      $gc+ =$ EXPLORETIDSET$(nleft.ss, \mathcal{K}, m)$;
  Return gc;
End

3.2 Illustrative example

To illustrate our approach, let us consider the formal concept $C_1 = (A_1, B_1)$ from the formal context depicted by Table 1 such that $A_1 = \{3, 4, 5, 6, 7, 9\}$ and $B_1 = \{f, g\}$. As shown in figure 1, the DFSP algorithm operates as follows:

During the first step (1), the root node is created and initialized though the function BUILDTREEROOT (gc=0). Initially, $root.s = \emptyset$ and nodes n_3, n_4, n_5, n_6, n_7 will be created through individual elements of $\{3, 4, 5, 6, 7, 9\}$ (steps (2), (3), (4), (5), (6) and (7)). These nodes are prospective direct children to the root node. Given that all these nodes are non generators, they become in step (8) as effective direct children of root and are decreasingly sorted with respect to their support value. In step (9), non generators forming monotone clique are placed at the end of the list and marked by (*). However,

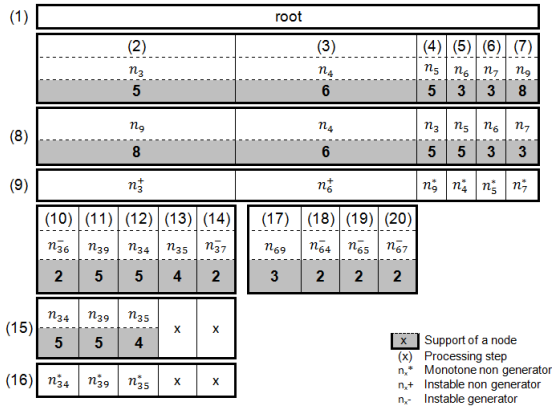


Fig. 1. Illustrative example

instable generators are placed at the beginning of the list and marked by (+). After that, in steps (10), (11), (12), (13) and (14) prospective direct children of node n_3 are created which are, respectively, $n_{36}, n_{39}, n_{34}, n_{35}$ and n_{37}. The count of generators is updated in step (15) ($gc = 2^4+2^3 = 24$). Only nodes n_{34}, n_{35} and n_{39} are left as effective direct children of n_3. The latter are also sorted decreasingly. In step (16), all these effective direct children form a monotone clique and exploration of this branch is stopped. After that, nodes n_{69}, n_{64}, n_{65} and n_{67} are created and count of generators is also updated in step (21) with the tree generators of n_{64}, n_{65} and n_{67} ($gc = gc + 2^3 + 2^2 + 2^1 = 24 + 14 = 38$). Only the node n_{69} is kept in the list of effective direct children of n_6. Indeed, the latter does not fulfil the condition of EXPLORETIDSET to be launched.

4 Experimental results

In this section, we put the focus on the evaluation of the DFSP algorithm by stressing on two complementary aspects : (*i*)Execution time; (*ii*) efficiency of search space pruning. Experiments were carried out on an Intel Xeon PC, CPU E5-2630 2,30 GHz with 16 GB of RAM and Linux system. During the lead experiments, we used some benchmark datasets commonly of extensive use within Data mining. The first three datasets are considered as dense ones, *i.e.*, yielding high number of formal concepts even for a small number of objects and attributes, while the other ones are considered as sparse. The characteristics of these datasets are summarized by Table 2. Thus, for each dataset we report its number of objects, the number of attributes, as well as the number of all formal concepts that may drawn. In addition, we also reported the respective sizes of the smallest and the largest formal concepts (in terms of extent's size). For these considered concepts, we kept track of the number of the actually explored nodes as well as the execution time (the column denoted $|explor.|$).

At a glance, statistics show that the DFSP algorithm is able to process dozens of thousands of objects in a reasonable time. Indeed, the 15596 (respec. 16040) objects

composing the extent of the largest formal concept extracted from the RETAIL (respec. T10I4D100K) dataset are handled in only 27.27 (respec. 68.85) seconds. Even though, the respective cardinalities are close (15596 vs 16040 objects), the difference in execution time is not proportional to this low gap. A preliminary explanation could be the difference in density of both datasets (RETAIL is dense while T10I4D100K is a sparse one). A in-depth study of these performances in connection to the nature of datasets is currently carried out. The most sighting fact is the low number of visited nodes in the associated search space. For example for the MUSHROOM dataset, DFSP algorithm actually handled only 83918 nodes from 2^{1000} potential nodes of the search space, i.e., in numerical terms it comes to only explore infinitely insignificant part equal to 7.8×10^{-297} of the search space. The case of RETAIL and T10I4D100K datasets is also worth of mention. For the respective smallest extracted concepts, DFSP algorithm only explores, $1.14 \times 10^{(-45)}$ and $1.5 \times 10^{(-90)}$ parts of the respective search spaces.

Datasets	**# Attr**	**# Obj**	**# concepts**	**smallest concept**			**largest concept**		
				$\|ext\|$	$\|explor.\|$	**time (sec.)**	$\|ext\|$	$\|explor.\|$	**time (sec.)**
CHESS	75	3196	3316	2630	2362233	0.12	3195	5855899	0.64
MUSHROOM	119	8124	3337	1000	83918	0.10	8124	76749955	11.32
RETAIL	16470	88162	3493	150	164	0.10	15596	64847191	27.27
T10I4D100K	1000	100000	4497	300	306	0.11	6810	19719991	12.77
T40I10D100K	1000	100000	3102	1800	1495324	1.39	16040	92154598	68.85

Table 2. Characteristics of the considered benchmark datasets

These highlights are also confirmed by Figures 2-11. Indeed, Figures 2, 4, 6, 8 and 10 stress on the variation of the Execution time, while Figures 3, 5, 7, 9 and 11 assess what we call the workload which means the efficiency of search space exploration. At a glance, the execution time is in a snugness connection with the reduction of search space, i.e., the variation of the workload has the same tendency as the performance since we consider the visited tidset in the search space as the processing unit. Worth of mention, the performance is rather correlated to the extent's size rather than the exponential nature of the search space.

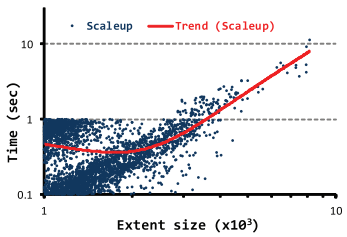


Fig. 2. Mushroom scaleup

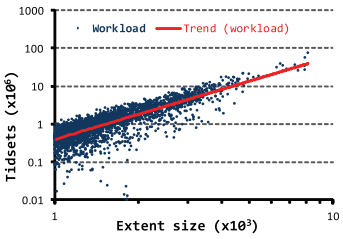


Fig. 3. Mushroom workload

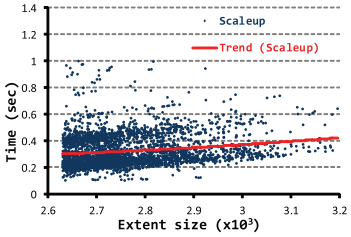


Fig. 4. Chess scaleup

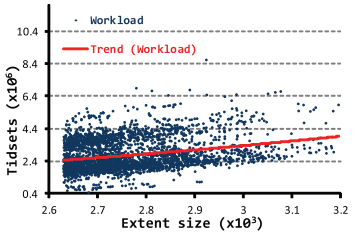


Fig. 5. Chess workload

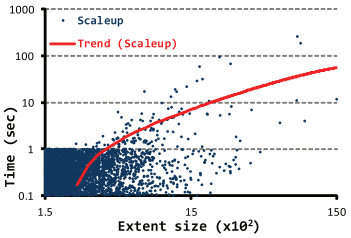


Fig. 6. Retail scaleup

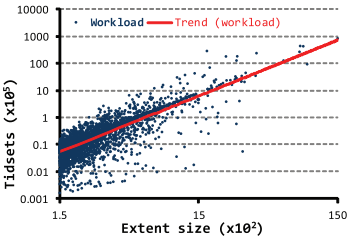


Fig. 7. Retail workload

5 Conclusion and future work

Through the DFSP algorithm, we gaped in the combinatorics of lattices by the showing that most of this sear space could be smartly explored thanks to the saturation of generators. The swift computation of stability encouraged us to integrate the stability as a on-the-fly pruning strategy during mining closed itemsets. We are currently working on a new algorithm for the stability computation given the Galois lattice. The new algorithm only relies on the direct sub-concepts to compute the stability of a concept. Outside the FCA field, the strategy of DFSP would be of benefit for very efficient extraction well known problem of combinatorics : minimal transversals.

References

1. Babin, M.A., Kuznetsov, S.O.: Approximating concept stability. In: Proceedings of the 11th International Conference on Formal Concept Analysis(ICFCA), Dresden, Germany. (2012) 7–15
2. Buzmakov, A., Kuznetsov, S.O., Napoli, A.: Scalable estimates of concept stability. In: Proceedings of the 12th International Conference on Formal Concept Analysis(ICFCA), Cluj-Napoca, Romania. (2014) 157–172

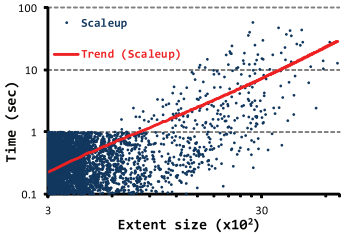


Fig. 8. T10I4D100K scaleup

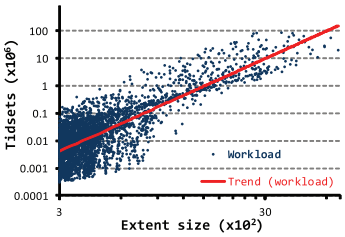


Fig. 9. T10I4D100K workload

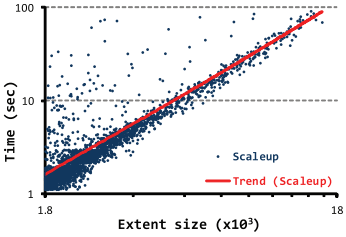


Fig. 10. T40I10D100K scaleup

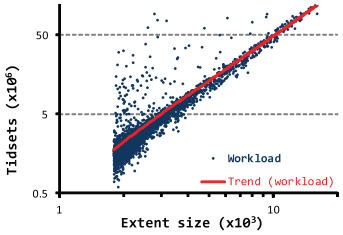


Fig. 11. T40I10D100K workload

3. Kuznetsov, S.O.: Stability as an estimate of the degree of substantiation of hypotheses derived on the basis of operational similarity. Automatic Documentation and Mathematical Linguistics **24** (1990) 62–75
4. Kuznetsov, S.O., Obiedkov, S.A., Roth, C.: Reducing the representation complexity of lattice-based taxonomies. In: Proceedings of the 15th International Conference on Conceptual Structures (ICCS), Sheffield, UK. (2007) 241–254
5. Kuznetsov, S.O.: On stability of a formal concept. Ann. Math. Artif. Intell. **49** (2007) 101–115
6. Buzmakov, A., Kuznetsov, S.O., Napoli, A.: Is concept stability a measure for pattern selection? Procedia Computer Science **31** (2014) 918 – 927
7. Klimushkin, M., Obiedkov, S.A., Roth, C.: Approaches to the selection of relevant concepts in the case of noisy data. In: Proceedings of the 8th International Conference(ICFCA) , Agadir, Morocco. (2010) 255–266
8. Roth, C., Obiedkov, S.A., Kourie, D.G.: Towards concise representation for taxonomies of epistemic communities. In: Proceedings of the 4th International Conference on Concept Lattices and Their Applications (CLA), Hammamet, Tunisia. (2006) 240–255
9. Jay, N., Kohler, F., Napoli, A.: Analysis of social communities with iceberg and stability-based concept lattices. In: Proceedings of the 6th International Conference(ICFCA), Montreal, Canada. (2008) 258–272
10. Roth, C., Obiedkov, S.A., Kourie, D.G.: On succinct representation of knowledge community taxonomies with formal concept analysis. Int. J. Found. Comput. Sci. **19** (2008) 383–404
11. Qiao, S.Y., Wen, S.P., Chen, C.Y., Li, Z.G.: A fast algorithm for building concept lattice. (2003)
12. Demko, C., Bertet, K.: Generation algorithm of a concept lattice with limited object access. In: Proceedings of the 8th International Conference Concept Lattices and Their Applications (CLA), Nancy, France. (2011) 239–250

Attributive and Object Subcontexts in Inferring Good Maximally Redundant Tests

Xenia Naidenova[1] and Vladimir Parkhomenko[2]

[1] Military Medical Academy, Saint-Petersburg, Russia
ksennaid@gmail.com
[2] St. Petersburg State Polytechnical University, Saint-Petersburg, Russia
parhomenko.v@gmail.com

Abstract. Inferring Good Maximally Redundant Classification Tests (GMRTs) as Formal Concepts is considered. Two kinds of classification subcontexts are defined: attributive and object ones. The rules of forming and reducing subcontexts based on the notion of essential attributes and objects are given. They lead to the possibility of the inferring control. In particular, an improved Algorithm for Searching all GMRTs on the basis of attributive subtask is proposed. The hybrid attributive and object approaches are presented. Some computational aspects of algorithms are analyzed.

Keywords: good classification test, Galois lattice, essential attributes and objects, implications, subcontexts

1 Introduction

Good Test Analysis (GTA) deals with the formation of the best descriptions of a given object class (class of positive objects) against the objects which do not belong to this class (class of negative objects) on the basis of lattice theory. We assume that objects are described in terms of values of a given set U of attributes, see an example in Tab.1. The key notion of GTA is the notion of classification. To give a target classification of objects, we use an additional attribute KL $\notin U$. A target attribute partitions a given set of objects into disjoint classes the number of which is equal to the number of values of this attribute. In Tab.1, we have two classes: the objects in whose descriptions the target value k appears and all the other objects.

Denote by M the set of attribute values such that $M = \{\cup \mathrm{dom}(\mathrm{attr}), \mathrm{attr} \in U\}$, where dom(attr) is the set of all values of attr, i.e. a plain scaling in terms of [3]. Let $G = G_+ \cup G_-$ be the set of objects, where G_+ and G_- are the sets of positive and negative objects respectively. Let $P(B), B \subseteq M$, be the set of all the objects in whose descriptions B appears. $P(B)$ is called the interpretation of B in the power set 2^G. If $P(B)$ contains only G_+ objects and the number of these objects is more than 2, then B is called a description of some positive objects or a diagnostic (classification) **test** for G_+ [1]. The words diagnostic (classification) can be omitted in the paper.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

Table 1. Motivating Example of classification

No	Height	Color of Hair	Color of Eyes	KL
1	Low	Blond	Blue	$k(+)$
2	Low	Brown	Blue	$k(-)$
3	Tall	Brown	Hazel	$k(-)$
4	Tall	Blond	Hazel	$k(-)$
5	Tall	Brown	Blue	$k(-)$
6	Low	Blond	Hazel	$k(-)$
7	Tall	Red	Blue	$k(+)$
8	Tall	Blond	Blue	$k(+)$

Let us recall the definition of a good test or good description for a subset of G_+ (via partitions of objects). A subset $B \subseteq M$ of attribute values is a **good test** for a subset of positive objects if it is a test and no such subset $C \subseteq M$ exists, so that $P(B) \subset P(C) \subseteq G_+$ [7].

Sec.2 is devoted to defining a concept of good diagnostic (classification) test as a formal concept. Sec.3 gives the decomposition of good tests inferring based on two kinds of subcontexts of the initial classification context. Sec.4 is devoted to an analysis of algorithms based on using subcontexts including the evaluation of the number of sub-problems to be solved, the depth of recursion, the structure of sub-problems and their ordering, and some others.

2 Good Maximally Redundant Tests as Formal Concepts

Assume that $G = \overline{1,N}$ is the set of objects indices (objects, for short) and $M = \{m_1, m_2, \ldots, m_j, \ldots m_m\}$ is the set of attributes values (values, for short). Each object is described by a set of values from M. The object descriptions are represented by rows of a table whose columns are associated with the attributes taking their values in M.

Let $A \subseteq G$, $B \subseteq M$. Denote by B_i, $B_i \subseteq M$, $i = \overline{1,N}$ the description of object with index i. The Galois connection between the ordered sets $(2^G, \subseteq)$ and $(2^M, \subseteq)$ is defined by the following mappings called derivation operators: for $A \subseteq G$ and $B \subseteq M$, $A' = \text{val}(A) = \{\text{intersection of all } B_i | \, B_i \subseteq M, i \in A\}$ and $B' = \text{obj}(B) = \{i | \, i \in G, B \subseteq B_i\}$. Of course, we have $\text{obj}(B) = \{\text{intersection of all obj}(m) | \, \text{obj}(m) \subseteq G, m \in B\}$.

There are two closure operators [9]: $\text{generalization\_of}(B) = B'' = \text{val}(\text{obj}(B))$ and $\text{generalization\_of}(A) = A'' = \text{obj}(\text{val}(A))$. A set A is closed if $A = \text{obj}(\text{val}(A))$. A set B is closed if $B = \text{val}(\text{obj}(B))$. For $g \in G$ and $m \in M$, $\{g\}'$ is denoted by g' and called object intent, and $\{m\}'$ is denoted by m' and called value extent. Let us recall the main definitions of GTA [7].

A **Diagnostic Test** (DT) for the positive examples G_+ is a pair (A, B) such that $B \subseteq M$, $A = B' \neq \emptyset$, $A \subseteq G_+$, $B \not\subseteq g' \; \forall g \in G_-$. A diagnostic test (A, B)

for G_+ is **maximally redundant** if $obj(B \cup m) \subset A$ for all $m \notin B$ and $m \in M$. A diagnostic test (A, B) for G_+ is **good** if and only if any extension $A_* = A \cup i$, $i \notin A$, $i \in G_+$ implies that $(A_*, val(A_*))$ is not a test for G_+.

In the paper, we deal with Good Maximally Redundant Tests (GMRTs). If a good test (A, B) for G_+ is maximally redundant, then any extension $B_* = B \cup m$, $m \notin B$, $m \in M$ implies that $(\text{obj}(B_*), B_*)$ is not a good test for G_+. Any object description d of $g \in G$ in a given classification context is a maximally redundant set of values because $\forall m \notin d$, $m \in M, \text{obj}(d \cup m)$ is equal to $\emptyset$. GMRT can be regarded as a special type of hypothesis [4]

In Tab.1, ((1, 8), Blond Blue) is a GMRT for $k(+)$, ((4, 6), Blond Hazel) is a DT for $k(-)$ but not a good one, and ((3, 4, 6), Hazel) is a GMRT for $k(-)$.

3 The Decomposition of Inferring GMRTs into Subtasks

There are two possible kinds of subtasks of inferring GMRTs for a set G_+ [8]:

1. given a set of values, where $B \subseteq M$, $\text{obj}(B) \neq \emptyset$, B is not included in any description of negative object, find all GMRTs $(\text{obj}(B_*), B_*)$ such that $B_* \subset B$;
2. given a non-empty set of values $X \subseteq M$ such that $(\text{obj}(X), X)$ is not a test for positive objects, find all GMRTs $(\text{obj}(Y), Y)$ such that $X \subset Y$.

For solving these subtasks we need only form subcontexts of a given classification context. The first subtask is useful to find all GMRTs whose intents are contained in the description d of an object g. This subtask is considered in [2] for fast incremental concept formation, where the definition of subcontexts is given.

We introduce the **projection of a positive object description** d on the set D_+, i.e. descriptions of all positive objects. The $\text{proj}(d)$ is $Z = \{z |\ z = d \cap d_* \neq \emptyset, d_* \in D_+$ and $(\text{obj}(z), z)$ is a test for $G_+\}$.

We also introduce a concept of **value projection** $\text{proj}(m)$ of a given value m on a given set D_+. The value projection is $\text{proj}(m) = \{d |\ m$ appears in d, $d \in D_+\}$.

Algorithm Algorithm for Searching all GMRTs on the basis of attributive subtask (ASTRA), based on value projections, was advanced in [6]. Algoritm DIAGaRa, based on object projections, was proposed in [5]. In what follows, we are interested in using both kinds of subcontexts for inferring all GMRTs for a positive (or negative) class of objects. The following theorem gives the foundation of reducing subcontexts [6].

Theorem 1. *Let* $X \subseteq M$, $(\text{obj}(X), X)$ *be a maximally redundant test for positive objects and* $\text{obj}(m) \subseteq \text{obj}(X)$, $m \in M$. *Then* m *can not belong to any GMRT for positive objects different from* $(obj(X), X)$.

Consider some example of reducing subcontext (see Tab.1). Let $\text{splus}(m)$ be $\text{obj}(m) \cap G_+$ or $\text{obj}(m) \cap G_-$ and SPLUS be $\{\text{splus}(m) |\ m \in M\}$. In Tab.1, we have $\text{SPLUS} = \text{obj}(m) \cap G_- = \{\{3, 4, 6\}, \{2, 3, 5\}, \{3, 4, 5\}, \{2, 5\}, \{4, 6\}, \{2, 6\}\}$ for values "Hazel, Brown, Tall, Blue, Blond, and Low" respectively.

We have val(obj(Hazel)) = Hazel, hence $((3,4,6), \text{Hazel})$ is a DT for G_-. Then value "Blond" can be deleted from consideration, because splus(Blond) $\subset$ splus(Hazel). Delete values Blond and Hazel from consideration. After that the description of object 4 is included in the description of object 8 of G_+ and the description of object 6 is included in the description of object 1 of G_+. Delete objects 4 and 6. Then for values "Brown, Tall, Blue, and Low" respectively SPLUS = $\{\{2,3,5\},\{3,5\},\{2,5\},\{2\}\}$. Now we have val(obj(Brown)) = Brown and $((2,3,5), \text{Brown})$ is a test for G_-. All values are deleted and all GMRTs for G_- have been obtained.

The initial information for finding all the GMRTs contained in a positive object description is the projection of it on the current set D_+. It is essential that the projection is a subset of object descriptions defined on a certain restricted subset t_* of values. Let s_* be the subset of indices of objects whose descriptions produce the projection. In the projection, $\text{splus}(m) = \text{obj}(m) \cap s_*,\ m \in t_*$.

Let STGOOD be the partially ordered set of elements s satisfying the condition that $(s, \text{val}(s))$ is a good test for D_+. The basic recursive procedure for solving any kind of subtask consists of the following steps:

1. Check whether $(s_*, \text{val}(s_*))$ is a test and if so, then s_* is stored in STGOOD if s_* corresponds to a good test at the current step; in this case, the subtask is over. Otherwise go to the next step.
2. The value m can be deleted from the projection if $\text{splus}(m) \subseteq s$ for some $s \in$ STGOOD.
3. For each value m in the projection, check whether $(\text{splus}(m), \text{val}(\text{splus}(m)))$ is a test and if so, then value m is deleted from the projection and splus(m) is stored in STGOOD if it corresponds to a good test at the current step.
4. If at least one value has been deleted from the projection, then the reduction of the projection is necessary. The reduction consists in checking, for each element t of the projection, whether $(\text{obj}(t), t)$ is not a test (as a result of previous eliminating values) and if so, this element is deleted from the projection. If, under reduction, at least one element has been deleted, then Step 2, Step 3, and Step 4 are repeated.
5. Check whether the subtask is over or not. The subtask is over when either the projection is empty or the intersection of all elements of the projection corresponds to a test (see, please, Step 1). If the subtask is not over, then the choice of an object (value) in this projection is selected and the new subtask is formed. The new subsets s_* and t_* are constructed and the basic algorithm runs recursively.

The algorithm of forming STGOOD is based on topological sorting of partially ordered sets. The set TGOOD of all the GMRTs is obtained as follows: $\text{TGOOD} = \{\text{tg} |\ \text{tg} = (s, \text{val}(s)),\ s \in \text{STGOOD}\}$.

4 Selecting and Ordering Subcontexts and Inferring GMRTs

Algorithms for inferring GMRTs are constructed by the rules of selecting and ordering subcontexts of the main classification context. Before entering into the details, let us recall some extra definitions. Let t be a set of values such that $(\text{obj}(t), t)$ is a test for G_+. We say that **the value** $m \in M, m \in t$ **is essential** in t if $(\text{obj}(t \setminus m), (t \setminus m))$ is not a test for a given set of objects. Generally, we are interested in finding the maximal subset $\text{sbmax}(t) \subset t$ such that $(\text{obj}(t), t)$ is a test but $(\text{obj}(\text{sbmax}(t)), \text{sbmax}(t))$ is not a test for a given set of positive objects. Then $\text{sbmin}(t) = t \setminus \text{sbmax}(t)$ is a minimal set of essential values in t. Let $s \subseteq G_+$, assume also that $(s, \text{val}(s))$ is not a test.

The object t_j, $j \in s$ **is said to be an essential** in s if $(s \backslash j, \text{val}(s \backslash j))$ proves to be a test for a given set of positive objects. Generally, we are also interested in finding the maximal subset $\text{sbmax}(s) \subset s$ such that $(s, \text{val}(s))$ is not a test but $(\text{sbmax}(s), \text{val}(\text{sbmax}(s))$ is a test for a given set of positive objects. Then $\text{sbmin}(s) = s \setminus \text{sbmax}(s)$ is a minimal set of essential objects in s.

An Approach for Searching for Initial Content of STGOOD. In the beginning of inferring GMRTs, the set STGOOD is empty. Next we describe the procedure to obtain an initial content of it. This procedure extracts a quasi-maximal subset $s_* \subseteq G_+$ which is the extent of a test for G_+ (maybe not good).

We begin with the first index i_1 of s_*, then we take the next index i_2 of s_* and evaluate the function $\text{to\_be\_test}(\{i_1, i_2\}, \text{val}(\{i_1, i_2\}))$. If the value of the function is true, then we take the next index i_3 of s_* and evaluate the function $\text{to\_be\_test}(\{i_1, i_2, i_3\}, \text{val}(\{i_1, i_2, i_3\}))$. If the value of the function is false, then the index i_2 of s_* is skipped and the function $\text{to\_be\_test}(\{i_1, i_3\}, \text{val}(\{i_1, i_3\})))$ is evaluated. We continue this process until we achieve the last index of s_*.

The complexity of this procedure is evaluated as the production of $||s_*||$ by the complexity of the function to_be_test(). To obtain the initial content of STGOOD, we use the set $\text{SPLUS} = \{\text{splus}(m) | m \in M\}$ and apply the procedure described above to each element of SPLUS.

The idea of using subcontexts in inferring GMRTs, described in Sec.3, can be presented in a pseudo-code form, see Fig.1. It presents a modification of ASTRA. DIAGARA and a hybrid approach can be easily formalized by the same way. The example below describes two general hybrid methods.

The initial part of GenAllGMRTs() is well discussed above. The abbreviation LEV stands for the List (set) of Essential Values. The function $\text{DelObj}(M, G_+)$ returns modified G and $flag$. The variable $flag$ is necessary for switching attributive subtasks. The novelty of ASTRA-2 is mainly based on using LEV. There is the new function ChoiceOfSubtask(). It returns $na := \text{LEV}_j$ with the maximal $2^{splus(\text{LEV}_j)}$. MainContext, defined $\text{FormSubTask}(na, M, G_+)$, consists of object descriptions. There is the auxiliary function $\text{kt}(m) = true$ if $(m' \in G_- = false)$ and $false$ otherwise.

To illustrate this procedure, we use the sets D_+ and D_- represented in Tab.2 and 3 (our illustrative example). In these tables, $M = \{m_1, \ldots, m_{26}\}$. The set SPLUS_0 for positive class of examples is in Tab.4. The initial content of

```
1.Algorithm GenAllGMRTs()
    Input: G, M
    Output: STGOOD
2.  begin
3.      Forming STGOOD ;
4.      Forming and Ordering LEV ;
5.      flag:=1;
6.  end
7.  while true do
8.      while flag=1 do
9.          M, flag DelVal(M, G+);
10.         if flag=1 then
11.             return;
12.         end
13.         G+, flag
            DelObj(M, G+);
14.     end
15.     if M' ⊆ G− or
        G+ ⊆ STGOOD then
16.         return STGOOD;
17.     end
18.     MSUB:=∅;
19.     GSUB:=∅;
20.     ChoiceOfSubtask;
21.     MSUB, GSUB
        FormSubTask(na, M, G+);
22.     GenAllGMRTs();
23.     M:=M\Mna;
24.     G+, flag DelObj(M, G+);
25. end
```

(a) GenAllGMRTs

```
1.Algorithm DelVal()
2.  i := 1;
3.  flag := 0;
4.  while i ≤ 2^M do
5.      if M'_i ⊆ G+ then
6.          M := M\M_i;
7.          flag := 1;
8.      end
9.      else if kt(M'_i ∩ G+) then
10.         j :=1 ;
11.         while j ≤ 2^STGOOD do
12.             if STGOOD_j ⊆
                M'_i ∩ G+ then
13.                 STGOOD :=
                    STGOOD\
                    STGOOD_j
14.             end
15.         end
16.         STGOOD :=
            STGOOD ∪ M'_i ∩ G+;
17.         M := M\M_i;
18.         flag := 1;
19.     return;
20. end
```

(b) DelVal

```
1.Algorithm DelObj()
2.  i := 1;
3.  flag := 0;
4.  while i ≤ 2^G+ do
5.      if G+(i) ⊆ M\LEV then
6.          G+ := G+\G+(i);
7.          flag := 1;
8.      end
9.  end
10. return;
```

(c) DelObj

```
1.Algorithm FormSubTask()
2.  i := 1;
3.  GSUB := M'na ∩ G+;
4.  while i ≤ 2^GSUB do
5.      MSUB := MSUB ∪
        (MainContext(GSUB(i)∩M));
6.  end
7.  return;
```

(d) FormSubTask

Fig. 1. Algorithms of ASTRA-2

STGOOD$_0$ is {(2,10), (3, 10), (3, 8), (4, 12), (1, 4, 7), (1, 5,12), (2, 7, 8), (3, 7, 12), (1, 2, 12, 14), (2, 3, 4, 7), (4, 6, 8, 11)}.

Table 2. The set D_+ of positive object descriptions

G	D_+
1	$m_1\ m_2\ m_5\ m_6\ m_{21}\ m_{23}\ m_{24}\ m_{26}$
2	$m_4\ m_7\ m_8\ m_9\ m_{12}\ m_{14}\ m_{15}\ m_{22}\ m_{23}\ m_{24}\ m_{26}$
3	$m_3\ m_4\ m_7\ m_{12}\ m_{13}\ m_{14}\ m_{15}\ m_{18}\ m_{19}\ m_{24}\ m_{26}$
4	$m_1\ m_4\ m_5\ m_6\ m_7\ m_{12}\ m_{14}\ m_{15}\ m_{16}\ m_{20}\ m_{21}\ m_{24}\ m_{26}$
5	$m_2\ m_6\ m_{23}\ m_{24}$
6	$m_7\ m_{20}\ m_{21}\ m_{26}$
7	$m_3\ m_4\ m_5\ m_6\ m_{12}\ m_{14}\ m_{15}\ m_{20}\ m_{22}\ m_{24}\ m_{26}$
8	$m_3\ m_6\ m_7\ m_8\ m_9\ m_{13}\ m_{14}\ m_{15}\ m_{19}\ m_{20}\ m_{21}\ m_{22}$
9	$m_{16}\ m_{18}\ m_{19}\ m_{20}\ m_{21}\ m_{22}\ m_{26}$
10	$m_2\ m_3\ m_4\ m_5\ m_6\ m_8\ m_9\ m_{13}\ m_{18}\ m_{20}\ m_{21}\ m_{26}$
11	$m_1\ m_2\ m_3\ m_7\ m_{19}\ m_{20}\ m_{21}\ m_{22}\ m_{26}$
12	$m_2\ m_3\ m_{16}\ m_{20}\ m_{21}\ m_{23}\ m_{24}\ m_{26}$
13	$m_1\ m_4\ m_{18}\ m_{19}\ m_{23}\ m_{26}$
14	$m_{23}\ m_{24}\ m_{26}$

In these tables we denote subsets of values $\{m_8, m_9\}$, $\{m_{14}, m_{15}\}$ by m_a and m_b, respectively. Applying operation generalization_of$(s) = s'' = \text{obj}(\text{val}(s))$ to $\forall s \in$ STGOOD, we obtain STGOOD$_1$ = {(2,10), (3, 10), (3, 8), (4, 7, 12), (1, 4, 7), (1, 5,12), (2, 7, 8), (3, 7, 12), (1, 2, 12, 14), (2, 3, 4, 7), (4, 6, 8, 11)}.

By Th.1, we can delete value m_{12} from consideration, see splus(m_{12}) in Tab.4. The initial content of STGOOD allows to decrease the number of using the procedure to_be_test() and the number of putting extents of tests into STGOOD.

The number of subtasks to be solved. This number is determined by the number of essential values in the set M. The quasi-minimal subset of essential values in M can be found by a procedure analogous to the procedure applicable to search for the initial content of STGOOD. We begin with the first value m_1 of M, then we take the next value m_2 of M and evaluate the function to_be_test(obj($\{m_1, m_2\}$), $\{m_1, m_2\}$). If the value of the function is false, then we take the next value m_3 of M and evaluate the function to_be_test(obj($\{m_1, m_2, m_3\}$), $\{m_1, m_2, m_3\}$). If the value of the function is true, then value m_2 of M is skipped and the function to_be_test(obj($\{m1, m3\}$), $\{m1$, $m3\}$) is evaluated. We continue this process until we achieve the last value of M. The complexity of this procedure is evaluated as the production of $||M||$ by the complexity of the function to_be_test(). In Tab.2,3 we have the following LEV: $\{m_{16}, m_{18}, m_{19}, m_{20}, m_{21}, m_{22}, m_{23}, m_{24}, m_{26}\}$.

Table 3. The set D_- of negative object descriptions

G	D_-	G	D_-
15	$m_3m_8m_{16}m_{23}m_{24}$	32	$m_1m_2m_3m_7m_9m_{13}m_{18}$
16	$m_7m_8m_9m_{16}m_{18}$	33	$m_1m_5m_6m_8m_9m_{19}m_{20}m_{22}$
17	$m_1m_{21}m_{22}m_{24}m_{26}$	34	$m_2m_8m_9m_{18}m_{20}m_{21}m_{22}m_{23}m_{26}$
18	$m_1m_7m_8m_9m_{13}m_{16}$	35	$m_1m_2m_4m_5m_6m_7m_9m_{13}m_{16}$
19	$m_2m_6m_7m_9m_{21}m_{23}$	36	$m_1m_2m_6m_7m_8m_{13}m_{16}m_{18}$
20	$m_{19}m_{20}m_{21}m_{22}m_{24}$	37	$m_1m_2m_3m_4m_5m_6m_7m_{12}m_{14}m_{15}m_{16}$
21	$m_1m_{20}m_{21}m_{22}m_{23}m_{24}$	38	$m_1m_2m_3m_4m_5m_6m_9m_{12}m_{13}m_{16}$
22	$m_1m_3m_6m_7m_9m_{16}$	39	$m_1m_2m_3m_4m_5m_6m_{14}m_{15}m_{19}m_{20}m_{23}m_{26}$
23	$m_2m_6m_8m_9m_{14}m_{15}m_{16}$	40	$m_2m_3m_4m_5m_6m_7m_{12}m_{13}m_{14}m_{15}m_{16}$
24	$m_1m_4m_5m_6m_7m_8m_{16}$	41	$m_2m_3m_4m_5m_6m_7m_9m_{12}m_{13}m_{14}m_{15}m_{19}$
25	$m_7m_{13}m_{19}m_{20}m_{22}m_{26}$	42	$m_1m_2m_3m_4m_5m_6m_{12}m_{16}m_{18}m_{19}m_{20}m_{21}m_{26}$
26	$m_1m_2m_3m_5m_6m_7m_{16}$	43	$m_4m_5m_6m_7m_8m_9m_{12}m_{13}m_{14}m_{15}m_{16}$
27	$m_1m_2m_3m_5m_6m_{13}m_{18}$	44	$m_3m_4m_5m_6m_8m_9m_{12}m_{13}m_{14}m_{15}m_{18}m_{19}$
28	$m_1m_3m_7m_{13}m_{19}m_{21}$	45	$m_1m_2m_3m_4m_5m_6m_7m_8m_9m_{12}m_{13}m_{14}m_{15}$
29	$m_1m_4m_5m_6m_7m_8m_{13}m_{16}$	46	$m_1m_3m_4m_5m_6m_7m_{12}m_{13}m_{14}m_{15}m_{16}m_{23}m_{24}$
30	$m_1m_2m_3m_6m_{12}m_{14}m_{15}m_{16}$	47	$m_1m_2m_3m_4m_5m_6m_8m_9m_{12}m_{14}m_{16}m_{18}m_{22}$
31	$m_1m_2m_5m_6m_{14}m_{15}m_{16}m_{26}$	48	$m_2m_8m_9m_{12}m_{14}m_{15}m_{16}$

Table 4. The set SPLUS_0

splus$(m), m \in M$	splus$(m), m \in M$
splus$(m_a) \to \{2, 8, 10\}$	splus$(m_{22}) \to \{2, 7, 8, 9, 11\}$
splus$(m_{13}) \to \{3, 8, 10\}$	splus$(m_{23}) \to \{1, 2, 5, 12, 13, 14\}$
splus$(m_{16}) \to \{4, 9, 12\}$	splus$(m_3) \to \{3, 7, 8, 10, 11, 12\}$
splus$(m_1) \to \{1, 4, 11, 13\}$	splus$(m_4) \to \{2, 3, 4, 7, 10, 13\}$
splus$(m_5) \to \{1, 4, 7, 10\}$	splus$(m_6) \to \{1, 4, 5, 7, 8, 10\}$
splus$(m_{12}) \to \{2, 3, 4, 7\}$	splus$(m_7) \to \{2, 3, 4, 6, 8, 11\}$
splus$(m_{18}) \to \{3, 9, 10, 13\}$	splus$(m_{24}) \to \{1, 2, 3, 4, 5, 7, 12, 14\}$
splus$(m_2) \to \{1, 5, 10, 11, 12\}$	splus$(m_{20}) \to \{4, 6, 7, 8, 9, 10, 11, 12\}$
splus$(m_b) \to \{2, 3, 4, 7, 8\}$	splus$(m_{21}) \to \{1, 4, 6, 8, 9, 10, 11, 12\}$
splus$(m_{19}) \to \{3, 8, 9, 11, 13\}$	splus$(m_{26}) \to \{1, 2, 3, 4, 6, 7, 9, 10, 11, 12, 13, 14\}$

Proposition 1. *Each essential value is included at least in one positive object description.*

Proof. Assume that for an object description $t_i, i \in G_+$, we have $t_i \cap LEV = \emptyset$. Then $t_i \subseteq M \backslash \text{LEV}$. But $M \backslash \text{LEV}$ is included at least in one of the negative object descriptions and, consequently, t_i also possesses this property. But it contradicts to the fact that t_i is a description of a positive object. □

Proposition 2. *Assume that* $X \subseteq M$. *If* $X \cap \text{LEV} = \emptyset$, *then* to_be_test$(X)$ = *false.*

Proposition 2 is the consequence of Proposition 1.

Note that the description of $t_{14} = \{m_{23}, m_{24}, m_{26}\}$ is closed because of obj$\{m_{23}, m_{24}, m_{26}\} = \{1, 2, 12, 14\}$ and val$\{1, 2, 12, 14\} = \{m_{23}, m_{24}, m_{26}\}$. We also know that $s = \{1, 2, 12, 14\}$ is closed too (we obtained this result during generalization of elements of STGOOD. So (obj$(\{m_{23}, m_{24}, m_{26}\})$, $\{m_{23}, m_{24}$, $m_{26}\}$) is a maximally redundant test for positive objects and we can, consequently, delete t_{14} from consideration. As a result of deleting m_{12} and t_{14}, we have the modified set SPLUS (Tab.5).

Table 5. The set SPLUS_1

splus$(m), m \in M$	splus$(m), m \in M$
splus$(m_a) \to \{2, 8, 10\}$	splus$(m_{22}) \to \{2, 7, 8, 9, 11\}$
splus$(m_{13}) \to \{3, 8, 10\}$	splus$(m_{23}) \to \{1, 2, 5, 12, 13\}$
splus$(m_{16}) \to \{4, 9, 12\}$	splus$(m_3) \to \{3, 7, 8, 10, 11, 12\}$
splus$(m_1) \to \{1, 4, 11, 13\}$	splus$(m_4) \to \{2, 3, 4, 7, 10, 13\}$
splus$(m_5) \to \{1, 4, 7, 10\}$	splus$(m_6) \to \{1, 4, 5, 7, 8, 10\}$
	splus$(m_7) \to \{2, 3, 4, 6, 8, 11\}$
splus$(m_{18}) \to \{3, 9, 10, 13\}$	splus$(m_{24}) \to \{1, 2, 3, 4, 5, 7, 12\}$
splus$(m_2) \to \{1, 5, 10, 11, 12\}$	splus$(m_{20}) \to \{4, 6, 7, 8, 9, 10, 11, 12\}$
splus$(m_b) \to \{2, 3, 4, 7, 8\}$	splus$(m_{21}) \to \{1, 4, 6, 8, 9, 10, 11, 12\}$
splus$(m_{19}) \to \{3, 8, 9, 11, 13\}$	splus$(m_{26}) \to \{1, 2, 3, 4, 6, 7, 9, 10, 11, 12, 13\}$

The main question is how we should approach the problem of selecting and ordering subtasks (subcontexts). Consider Tab.6 with auxiliary information. It is clear that if we shall have all the intents of GMRTs entering into descriptions of objects 1, 2, 3, 5, 7, 9, 10, 12, then the main task will be over because the remaining object descriptions (objects 4, 6, 8, 11) give, in their intersection, the intent of already an known test (see, please, the initial content of STGOOD). Thus we have to consider only the subcontexts of essential values associated with object descriptions 1, 2, 3, 5, 7, 9, 10, 12, 13. The number of such subcontexts is 39. But this estimation is not realistic.

Table 6. Auxiliary information

index of object	m_{16}	m_{18}	m_{19}	m_{20}	m_{21}	m_{22}	m_{23}	m_{24}	m_{26}	$\sum m_{ij}$
1					×		×	×	×	4
2						×	×	×	×	4
3		×	×					×	×	4
5							×	×		2
7				×		×		×	×	4
9	×	×	×	×	×	×			×	7
10		×		×	×				×	4
12	×			×	×		×	×	×	4
13		×	×				×		×	4
4	×			×	×			×	×	
6				×	×				×	
8			×	×	×	×			×	
11			×	×	×	×			×	
$\sum d_i$	2	4	3	4	4	3	5	6	8	39

We begin with ordering index of objects by the number of their entering in tests in STGOOD_1, see Tab.7.

Table 7. Ordering index of objects in STGOOD_1

Index of object	**9**	**13**	**5**	**10**	**1**	**2**	**3**	**12**	**7**
The number of entering in STGOOD_1	0	0	1	2	3	4	4	4	5

Then we continue with object descriptions t_9 and t_{13}. Now we should select the subcontexts (subtasks), based on $\text{proj}(t \times m)$, where t is object description containing the smallest number of essential values and m is an essential value in t, entering in the smallest number of object descriptions. After solving each subtask, we have to correct the sets SPLUS, STGOOD, and auxiliary information. So, the first sub-task is $t_9 \times m_{16}$. Solving this sub-task, we have not any new test, but we can delete m_{16} from t_9 and then we solve the sub-task $t_9 \times m_{19}$. As a result, we introduce $s = \{9, 11\}$ in STGOOD and delete t_9 from consideration because of m_{16}, m_{19} are the only essential values in this object description.

In the example (**method 1**), we have the following subtasks (Tab. 8).

Tab.10 shows the sets STGOOD and TGOOD. All subtasks did not require a recursion. A simpler method of ordering contexts is based on the basic recursive procedure for solving any kind of subtask described in the previous section. At

Table 8. The sequence of subtasks (method 1)

N	subcontext	Extent of New Test	Deleted values	Deleted objects
1	$t_9 \times m_{16}$			
2	$t_9 \times m_{19}$	$(9, 11)$		t_9
3	$t_{13} \times m_{18}$			
4	$t_{13} \times m_{19}$	(13)	m_{16}, m_{18}	t_{13}
5	$t_5 \times m_{23}$		m_{23}	
6	$t_5 \times m_{24}$			t_5
7	$t_{10} \times m_{20}$	$(8, 10)$		
8	$t_{10} \times m_{21}$			
9	$t_{10} \times m_{26}$		m_a, m_{13}, m_4, m_5	t_{10}
10	$t_1 \times m_{21}$			
11	$t_1 \times m_{24}$		m_1, m_2	t_1
12	$t_2 \times m_{22}$	$(7, 8, 11)$	m_{22}	
13	$t_2 \times m_{22}$			
14	$t_2 \times m_{24}$			t_2
15	$t_3 \times m_{19}$	$(3, 11)$	m_{19}	
16	$t_3 \times m_{24}$		m_{24}	t_{12}, t_7
17	$t_3 \times m_{26}$			t_3

each level of recursion, we can select the value entering into the greatest number of object descriptions; the object descriptions not containing this value generate the contexts to find GMRTs whose intents are included in them. For our example, value m_{26} does not cover two object descriptions: t_5 and t_8. The initial context is associated with m_{26}. The sequence of subtasks in the basic recursive procedure is in Tab.9 (**method 2**). We assume, in this example, that the GMRT intent of which is equal to t_{14} has been already obtained.

We consider only two possible ways of GMRTs construction based on decomposing the main classification context into subcontexts and ordering them by the use of essential values and objects. It is possible to use the two sets $\mathrm{QT} = \{\{i, j\} \subseteq G_+ | \, (\{i, j\}, \mathrm{val}(\{i, j\})$ is a test for $G_+\}$ and $\mathrm{QAT} = \{\{i, j\} \subseteq G_+ | (\{i, j\}, \mathrm{val}(\{i, j\})$ is not a test for $G_+\}$ for forming subcontexts and their ordering in the form of a tree structure.

5 Conclusion

In this paper, the decomposition of inferring good classification tests into subtasks of the first and second kinds is presented. This decomposition allows, in principle, to transform the process of inferring good tests into a step by step reasoning process.

The rules of forming and reducing subcontexts are given, in this paper. Various possibilities of constructing algorithms for inferring GMRTs with the use of both subcontexts are considered depending on the nature of GMRTs features.

Table 9. The sequence of subtasks (method 2)

N	Context, associated with	Extents of tests obtained	Values deleted from context	Object descriptions deleted from context
1	m_{26}	$(2,10),(3,10),$ $(2,3,4,7),(1,4,7)$	$m_a, m_{13}, m_b,$ m_5, m_6	t_{10}
2	m_{26}, m_{24}	$(3,7,12),$ $(4,7,12)$	$m_3, m_{20}, m_{23}, m_1,$ $m_2, m_4, m_7, m_{16},$ m_{18}, m_{19}, m_{22}	
	Subtask is over; return to the previous context and delete m_{24}			
3	m_{26}, not m_{24}, m_{23}	(13)	$m_3, m_7, m_{16}, m_{18},$ m_{19}, m_{20}, m_{22}	
	Subtask is over; return to the previous context, delete m_{23}			
4	m_{26}, not m_{24}, not m_{23}		$m_2, m_3, m_4, m_{16},$ $m_{18}m_{19}, m_{21}$	
5	$m_{26}, m_{22},$ not $m_{24},$ not m_{23}	(9,11), (7,11)		t_2, t_7
	Subtask is over; return to the previous context and delete m_{22}			
6	$m_{26},$ not $m_{24},$ not $m_{23},$ not m_{22}	(3,11), (4,6,11)	$m_2, m_3, m_4, m_{16},$ m_{18}, m_{19}	t_7, t_9, t_2, t_3
	Subtask is over; we have obtained all GMRTs whose intents contain m_{26}			
7	Context t_5	(1,5,12)		t_5
	Subtask is over; we have found all GMRTs whose intents are contained in t_5 .			
8	Context $t_8 \times m_{22}$	(7,8,11), (2,7,8)	$m_3, m_{20}, m_b, m_6,$ $m_a, m_{13}, m_{19}, m_{21}$	
	Subtask is over; return to the previous context and delete m_{22}			
9	Context t_8 without m_{22}	(8,10)	m_a	t_2, t_7
10	Context $t_8 \times m_{21}$ without m_{22}	(4,6,8,11)	m_7, m_{13}, m_{19}	t_6, t_{10}, t_{11}
	Subtask is over; return to the previous context and delete m_{21}, m_{20}			
11	Context t_8 without m_{22}, m_{21}, m_{20}	(3, 8)		t_4, t_6, t_{10}, t_{11}
	Subtask is over; we have found all GMRTs whose intents are contained in t_8.			

Table 10. The sets STGOOD and TGOOD

N	STGOOD	TGOOD	N	STGOOD	TGOOD
1	13	$m_1m_4m_{18}m_{19}m_{23}m_{26}$	9	2,7,8	m_bm_{22}
2	2,10	$m_4m_am_{26}$	10	1,5,12	$m_2m_{23}m_{24}$
3	3,10	$m_3m_4m_{13}m_{18}m_{26}$	11	4,7,12	$m_{20}m_{24}m_{26}$
4	8,10	$m_3m_6m_am_{13}m_{20}m_{21}$	12	3,7,12	$m_3m_{24}m_{26}$
5	9,11	$m_{19}m_{20}m_{21}m_{22}m_{26}$	13	7,8,11	$m_3m_{20}m_{22}$
6	3,11	$m_3m_7m_{19}m_{26}$	14	2,3,4,7	$m_4m_{12}m_bm_{24}m_{26}$
7	3,8	$m_3m_7m_{13}m_bm_{19}$	15	4,6,8,11	$m_7m_{20}m_{21}$
8	1,4,7	$m_5m_6m_{24}m_{26}$	16	1,2,12,14	$m_{23}m_{24}m_{26}$

References

1. Chegis, I., Yablonskii, S.: Logical methods of electric circuit control. Trudy Mian SSSR 51, 270–360 (1958), (in Russian)
2. Ferré, S., Ridoux, O.: The use of associative concepts in the incremental building of a logical context. In: Priss, U., Corbett, D., Angelova, G. (eds.) ICCS. Lecture Notes in Computer Science, vol. 2393, pp. 299–313. Springer (2002)
3. Ganter, B., Wille, R.: Formal concept analysis: mathematical foundations. Springer, Berlin (1999)
4. Ganter, B., Kuznetsov, S.O.: Formalizing hypotheses with concepts. In: Proceedings of the Linguistic on Conceptual Structures: Logical Linguistic, and Computational Issues. pp. 342–356. Springer-Verlag (2000)
5. Naidenova, X.A.: DIAGARA: An Incremental Algorithm for Inferring Implicative Rules from Examples. Inf. Theories and Application 12 - 2, 171 – 196 (2005)
6. Naidenova, X.A., Plaksin, M.V., Shagalov, V.L.: Inductive Inferring All Good Classification Tests. In: Valkman, J. (ed.) "Knowledge-Dialog-Solution", Proceedings of International Conference. vol. 1, pp. 79 – 84. Kiev Institute of Applied Informatics, Kiev, Ukraine (1995)
7. Naidenova, X.A., Polegaeva, J.G.: An Algorithm of Finding the Best Diagnostic Tests. In: Mintz, G., Lorents, E. (eds.) The 4-th All Union Conference "Application of Mathematical Logic Methods". pp. 87 – 92 (1986), (in Russian)
8. Naidenova, X., Ermakov, A.: The decomposition of good diagnostic test inferring algorithms. In: Alty, J., Mikulich, L., Zakrevskij, A. (eds.) "Computer-Aided Design of Discrete Devices" (CAD DD2001), Proceedings of the 4-th Inter. Conf., vol. 3, pp. 61 – 68. Minsk (2001)
9. Ore, O.: Galois connections. Trans. Amer. Math. Soc 55, 494–513 (1944)

Removing an incidence from a formal context

Martin Kauer* and Michal Krupka**

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

Abstract. We analyze changes in the structure of a concept lattice corresponding to a context resulting from a given context with a known concept lattice by removing exactly one incidence. We identify the set of concepts affected by the removal and show how they can be used for computing concepts in the new concept lattice. We present algorithms for incremental computation of the new concept lattice, with or without structural information.

1 Introduction

When computing concept lattices of two very similar concepts (i.e., differing only in a small number of incidences), it doesn't seem to be efficient to compute both concept lattices independently. Rather, an incremental method of computing one of the lattices using the other would be more desirable. Also, analyzing structural differences between concept lattices of two similar contexts would be interesting from the theoretical point of view.

This paper presents first results in this direction. Namely, we consider two formal contexts differing in just one incidence and develop a method of computing the concept lattice of the context without the incidence from the other one. In other words, we give a first answer to the question "What happens to the concept lattice, if we remove one cross from the context?".

Our results are the following. We consider contexts $\langle X, Y, I\rangle$ and $\langle X, Y, J\rangle$ such that J results from I by removing exactly one incidence. Further we consider the respective concept lattices $\mathcal{B}(I)$ and $\mathcal{B}(J)$. For these contexts and concept lattices we

1. identify concepts in $\mathcal{B}(I)$, affected by the removal (they form an interval in $\mathcal{B}(I)$),

* The author acknowledges support by IGA of Palacky University, No. PrF 2014 034
** The author acknowledges support by the ESF project No. CZ.1.07/2.3.00/20.0059. The project is co-financed by the European Social Fund and the state budget of the Czech Republic.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

2. show how they transform to concepts in the new concept lattice (they will either vanish entirely, or transform to one or two concepts),
3. derive several further results on the correspondence between the two lattices,
4. propose two basic algorithms for transforming incrementally $\mathcal{B}(I)$ to $\mathcal{B}(J)$.

Several algorithms for incremental computation of concept lattices have been developed in the past [1, 5, 8, 6, 7, 2] (see also [4] for a comparison of some of the algorithms). In general, the algorithms build a concept lattice incrementally by modifying formal contexts by adding or removing objects one by one. Our approach is different as we focus on removing just one incidence.

2 Formal concept analysis

Formal Concept Analysis has been introduced in [9], our basic reference is [3]. A *(formal) context* is a triple $C = \langle X, Y, I\rangle$ where X is a set of objects, Y a set of attributes and $I \subseteq X \times Y$ a binary relation between X and Y. For $\langle x, y\rangle \in I$ it is said "The object x has the attribute y".

For subsets $A \subseteq X$ and $B \subseteq Y$ we set

$$A^{\uparrow_I} = \{y \in Y \mid \text{for each } x \in A \text{ it holds } \langle x, y\rangle \in I\},$$
$$B^{\downarrow_I} = \{x \in X \mid \text{for each } y \in B \text{ it holds } \langle x, y\rangle \in I\}.$$

The pair $\langle {}^{\uparrow_I}, {}^{\downarrow_I}\rangle$ is a Galois connection between sets X and Y, i.e., it satisfies for each $A, A_1, A_2 \subseteq X$, $B, B_1, B_2 \subseteq Y$,

1. If $A_1 \subseteq A_2$, then $A_2^{\uparrow_I} \subseteq A_1^{\uparrow_I}$, if $B_1 \subseteq B_2$, then $B_2^{\downarrow_I} \subseteq B_1^{\downarrow_I}$.
2. $A \subseteq A^{\uparrow_I\downarrow_I}$ and $B \subseteq B^{\downarrow_I\uparrow_I}$.

If $A^{\uparrow_I} = B$ and $B^{\downarrow_I} = A$, then the pair $\langle A, B\rangle$ is called a *formal concept* of $\langle X, Y, I\rangle$. The set A is called the *extent of* $\langle A, B\rangle$, the set B the *intent of* $\langle A, B\rangle$.

A partial order $\leq$ on the set $\mathcal{B}(X, Y, I)$ of all formal concepts of $\langle X, Y, I\rangle$ is defined by $\langle A_1, B_1\rangle \leq \langle A_2, B_2\rangle$ iff $A_1 \subseteq A_2$ (iff $B_2 \subseteq B_1$). $\mathcal{B}(X, Y, I)$ along with $\leq$ is a complete lattice and is called the *concept lattice* of $\langle X, Y, I\rangle$. Infima and suprema in $\mathcal{B}(X, Y, I)$ are given by

$$\bigwedge_{j\in J} \langle A_j, B_j\rangle = \left\langle \bigcap_{j\in J} A_j, \left(\bigcup_{j\in J} B_j\right)^{\downarrow_I\uparrow_I}\right\rangle, \tag{1}$$

$$\bigvee_{j\in J} \langle A_j, B_j\rangle = \left\langle \left(\bigcup_{j\in J} A_j\right)^{\uparrow_I\downarrow_I}, \bigcap_{j\in J} B_j\right\rangle. \tag{2}$$

One of immediate consequences of (1) and (2) is that the intersection of any system of extents (resp. intents) is again an extent (resp. intent).

Mappings $\gamma_I : x \mapsto \langle \{x\}^{\uparrow_I\downarrow_I}, \{x\}^{\uparrow_I}\rangle$ and $\mu_I : y \mapsto \langle \{y\}^{\downarrow_I}, \{y\}^{\downarrow_I\uparrow_I}\rangle$ assign to each object x its *object concept* and to each attribute y its *attribute concept*. We call a subset $K \subseteq L$, where L is a complete lattice, $\bigvee$-dense (resp. $\bigwedge$-dense) if

and only if any element of L can be expressed by suprema (resp. infima) of some elements from K. The set of all object concepts (resp. attribute concepts) is $\bigvee$-dense (resp. $\bigwedge$-dense) in $\mathcal{B}(X,Y,I)$. This can be easily seen from (1) (resp. (2)).

We will also need a notion of an interval in lattice L. We call a subset $K \subseteq L$ an *interval*, if and only if there exist elements $a, b \in L$ such that $K = \{k \in L \mid a \leq k \leq b\}$. We denote K as $[a, b]$.

3 Problem statement and basic notions

Let $\langle X, Y, I\rangle$, $\langle X, Y, J\rangle$ be two contexts over the same sets of objects and attributes such that $\langle x_0, y_0\rangle \notin J$ and $I = J \cup \{\langle x_0, y_0\rangle\}$.

We usually denote concepts of $\langle X, Y, I\rangle$ by c, c_1, $\langle A, B\rangle$, $\langle A_1, B_1\rangle$, etc., and concepts of $\langle X, Y, J\rangle$ by d, d_1, $\langle C, D\rangle$, $\langle C_1, D_1\rangle$, etc. The respective concept lattices will be denoted $\mathcal{B}(I)$ and $\mathcal{B}(J)$.

Our goal is to find an efficient way to compute the concept lattice $\mathcal{B}(J)$ from $\mathcal{B}(I)$. We provide two solutions to this problem. First solution computes just elements of $\mathcal{B}(J)$, the second one adds also information on its structure. In this section we introduce some basic tools and prove simple preliminary results.

The following proposition shows a correspondence between the derivation operators of contexts $\langle X, Y, I\rangle$ and $\langle X, Y, J\rangle$.

Proposition 1. *For each $A \subseteq X$ and $B \subseteq Y$ it holds*

$$A^{\uparrow_J} = \begin{cases} A^{\uparrow_I} & \text{if } x_0 \notin A, \\ A^{\uparrow_I} \setminus \{y_0\} & \text{if } x_0 \in A, \end{cases} \qquad B^{\downarrow_J} = \begin{cases} B^{\downarrow_I} & \text{if } y_0 \notin B, \\ B^{\downarrow_I} \setminus \{x_0\} & \text{if } y_0 \in B. \end{cases}$$

In particular, $A^{\uparrow_J} \subseteq A^{\uparrow_I}$ and $B^{\downarrow_J} \subseteq B^{\downarrow_I}$.

Proof. Immediate. □

Formal concepts from the intersection $\mathcal{B}(I) \cap \mathcal{B}(J)$ are called *stable*. These concepts are not influenced by removing the incidence $\langle x_0, y_0\rangle$ from I. When computing $\mathcal{B}(J)$ from $\mathcal{B}(I)$, stable concepts need not be recomputed.

Proposition 2. *A concept $c \in \mathcal{B}(I)$ is not stable iff $c \in [\gamma_I(x_0), \mu_I(y_0)]$.*

Proof. If $c = \langle A, B\rangle \notin [\gamma_I(x_0), \mu_I(y_0)]$, then either $x_0 \notin A$, or $y_0 \notin B$. If, for instance, $x_0 \notin A$, then by Proposition 1, $B = A^{\uparrow_I} = A^{\uparrow_J}$, showing B is the intent of a $d \in \mathcal{B}(J)$. Now by Proposition 1,

$$B^{\downarrow_J} = \begin{cases} B^{\downarrow_I} = A & \text{if } y_0 \notin B, \\ B^{\downarrow_I} \setminus \{x_0\} = A \setminus \{x_0\} = A & \text{if } y_0 \in B \end{cases}$$

and so $d = c$. The case $y_0 \notin B$ is dual.

To prove the opposite direction it is sufficient to notice that $c \in [\gamma_I(x_0), \mu_I(y_0)]$ is equivalent to $\langle x_0, y_0\rangle \in A \times B$, excluding the case $\langle A, B\rangle \in \mathcal{B}(J)$. □

For concepts $c = \langle A, B\rangle \in \mathcal{B}(I), d = \langle C, D\rangle \in \mathcal{B}(J)$ we set

$$c^{\square} = \langle A^{\square}, B^{\square}\rangle = \langle A^{\uparrow_J\downarrow_J}, A^{\uparrow_J}\rangle, \qquad c_{\square} = \langle A_{\square}, B_{\square}\rangle = \langle B^{\downarrow_J}, B^{\downarrow_J\uparrow_J}\rangle,$$
$$d^{\boxtimes} = \langle C^{\boxtimes}, D^{\boxtimes}\rangle = \langle D^{\downarrow_I}, D^{\downarrow_I\uparrow_I}\rangle, \qquad d_{\boxtimes} = \langle C_{\boxtimes}, D_{\boxtimes}\rangle = \langle C^{\uparrow_I\downarrow_I}, C^{\uparrow_I}\rangle.$$

Evidently, $c^{\square}, c_{\square} \in \mathcal{B}(J)$ and $d^{\boxtimes}, d_{\boxtimes} \in \mathcal{B}(I)$. $c^{\square}$ (resp. $c_{\square}$) is called *the upper* (resp. *lower*) *child of* c. In our setting, $d^{\boxtimes} = d_{\boxtimes}$ (it would not be the case if $I \setminus J$ had more than one element). It is the (unique) concept from $\mathcal{B}(I)$, containing, as a rectangle, the rectangle represented by d.

The following theorem shows basic properties of the pairs $\langle^{\square}, ^{\boxtimes}\rangle$ and $\langle_{\square}, _{\boxtimes}\rangle$.

Proposition 3 (child operators). *The mappings $c \mapsto c^{\square}$, $c \mapsto c_{\square}$, and $d \mapsto d^{\boxtimes}$ are isotone and satisfy*

$$c \le c^{\square\boxtimes}, \qquad d \le d^{\boxtimes\square}, \qquad c^{\square\boxtimes\square} = c^{\square}, \qquad d^{\boxtimes\square\boxtimes} = d^{\boxtimes},$$
$$c \ge c_{\square\boxtimes}, \qquad d \ge d_{\boxtimes\square}, \qquad c_{\square\boxtimes\square} = c_{\square}, \qquad d_{\boxtimes\square\boxtimes} = d_{\boxtimes}.$$

Proof. Isotony follows directly from definition.

Let $c = \langle A, B\rangle$. From Proposition 1 we have $A^{\uparrow_J} \subseteq A^{\uparrow_I}$. Thus, $A = A^{\uparrow_I\downarrow_I} \subseteq A^{\uparrow_J\downarrow_I}$, whence $c \le c^{\square\boxtimes}$. Similarly, for $d = \langle C, D\rangle$, $D^{\downarrow_J} \subseteq D^{\downarrow_I}$, whence $D^{\downarrow_I\uparrow_J} \subseteq D^{\downarrow_J\uparrow_J} = D$.

To prove $c^{\square\boxtimes\square} = c^{\square}$ it suffices to show that for the extent A of c it holds $A^{\uparrow_J\downarrow_I\uparrow_J} = A^{\uparrow_J}$. By Proposition 1, we have two possibilities: either $A^{\uparrow_J} = A^{\uparrow_I}$, or $A^{\uparrow_J} = A^{\uparrow_I} \setminus \{y_0\}$. In the first case $A^{\uparrow_J\downarrow_I\uparrow_J} = A^{\uparrow_J}$ holds trivially, in the second case $A^{\uparrow_J\downarrow_I} = A^{\uparrow_J\downarrow_J}$ (by the same proposition, because $y_0 \notin A^{\uparrow_J}$) and $A^{\uparrow_J\downarrow_I\uparrow_J} = A^{\uparrow_J\downarrow_J\uparrow_J} = A^{\uparrow_J}$. The equality $d^{\boxtimes\square\boxtimes} = d^{\boxtimes}$ can be proved similarly.

The assertions for lower children are dual. □

Corollary 1. *The mappings $c \mapsto c^{\square\boxtimes}$ and $d \mapsto d^{\boxtimes\square}$ are closure operators and the mappings $c \mapsto c_{\square\boxtimes}$ and $d \mapsto d_{\boxtimes\square}$ are interior operators.*

Following two theorems utilize the operators $^{\square}, ^{\boxtimes}, _{\square}, _{\boxtimes}$ to give several equivalent characterizations of stable concepts. First we prove a proposition.

Proposition 4. *The following assertions are equivalent for any $c = \langle A, B\rangle \in \mathcal{B}(I)$.*

1. *c is stable,*
2. *$A^{\uparrow_I} = A^{\uparrow_J}$,*
3. *$B^{\downarrow_I} = B^{\downarrow_J}$.*

Proof. "2 ⇒ 3": by Proposition 1, $A \subseteq A^{\uparrow_J\downarrow_J} = B^{\downarrow_J} \subseteq B^{\downarrow_I} = A$.

"3 ⇒ 2": dual.

The other implications follow by definition, since c is stable iff both 2. and 3. are satisfied. □

Proposition 5 (stable concepts in $\mathcal{B}(I)$). *The following assertions are equivalent for a concept $c \in \mathcal{B}(I)$:*

1. *c is stable,*
2. $c \notin [\gamma_I(x_0), \mu_I(y_0)]$,
3. $c = c^{\Box}$,
4. $c = c_{\Box}$,
5. $c^{\Box} = c_{\Box}$.

Proof. Directly from Proposition 4. □

Proposition 6 (stable concepts in $\mathcal{B}(J)$). *The following assertions are equivalent for a concept $d \in \mathcal{B}(J)$:*

1. *d is stable,*
2. $d = d^{\boxtimes}$,
3. $d^{\boxtimes}$ *is stable.*

Proof. Directly from Proposition 4. □

4 Computing $\mathcal{B}(J)$ without structural information

Proposition 7. *The following holds for $c = \langle A, B\rangle \in \mathcal{B}(I)$ and $d = \langle C, D\rangle \in \mathcal{B}(J)$: If $d = c^{\Box}$, then $B \in \{D, D \cup \{y_0\}\}$ and if $d = c_{\Box}$, then $A \in \{C, C \cup \{x_0\}\}$.*

Proof. By definition of $^{\Box}$, $D = A^{\uparrow_J}$, which is by Proposition 1 either equal to B, or to $B \setminus \{y_0\}$. Similarly for $_{\Box}$. □

Proposition 8. *A non-stable concept $d \in \mathcal{B}(J)$ is a (upper or lower) child of exactly one concept $c \in \mathcal{B}(I)$. This concept is non-stable and satisfies $c = d^{\boxtimes} = d_{\boxtimes}$.*

Proof. Let $d = \langle C, D\rangle$. Since d is non-stable, then either $C^{\uparrow_I} \neq C^{\uparrow_J}$, or $D^{\downarrow_I} \neq D^{\downarrow_J}$. Suppose $C^{\uparrow_I} \neq C^{\uparrow_J}$ and set $A = C$, $B = C^{\uparrow_I}$. By Proposition 1, $x_0 \in C$, $y_0 \notin D$ and $B = D \cup \{y_0\}$. By the same proposition, $A = C = D^{\downarrow_J} = D^{\downarrow_I}$, whence A is an extent of I. Thus, $c = \langle A, B\rangle \in \mathcal{B}(I)$ and it is non-stable because $x_0 \in A$ and $y_0 \in B$ (Proposition 2). Since $D = C^{\uparrow_J} = A^{\uparrow_J}$, $d = c^{\Box}$. $A = C$ yields $c = d_{\boxtimes}$.

We prove uniqueness of c. By Proposition 7, if for $c' = \langle A', B'\rangle \in \mathcal{B}(I)$ we have $d = c'^{\Box}$, then either $B' = D$, or $B' = D \cup \{y_0\}$. The first case is impossible, because it would make D an intent of I and, consequently, d a stable concept. The second case means c' equals c above. There is a third case left: if $d = c'_{\Box}$, then $C = B'^{\downarrow_J}$. Since $x_0 \in C$, we have $y_0 \notin B'$ (Proposition 1). Thus, $C = B'^{\downarrow_I}$ (Proposition 1 again). Consequently, $C^{\uparrow_I} = B'$ and since $y_0 \notin B'$, $B' = C^{\uparrow_J}$ (Proposition 1 for the last time). Thus, $d = c'$, which is a contradiction with non-stability of d.

The case $D^{\downarrow_I} \neq D^{\downarrow_J}$ is proved dually (in this case we obtain $d = c_{\Box}$). □

The meaning of the previous theorem is that for each non-stable concept in $\mathcal{B}(J)$ there exists exactly one non-stable concept in $\mathcal{B}(I)$, such that these two are related via mappings $^{\Box}, ^{\boxtimes}$ or $_{\Box}, _{\boxtimes}$.

The theorem leads the following simple way of constructing $\mathcal{B}(J)$ from $\mathcal{B}(I)$. For each $c \in \mathcal{B}(I)$ the following has to be done:

1. If c is stable, then it has to be added to $\mathcal{B}(J)$.
2. If c is not stable, then each its non-stable child (i.e., each non-stable element of $\{c^{\square}, c_{\square}\}$) has to be added to $\mathcal{B}(J)$.

This method ensures that all proper elements will be added to $\mathcal{B}(J)$ (i.e., no element will be omitted) and each element will be added exactly once.

Stable (resp. non-stable) concepts can be identified by means of Proposition 11. The following proposition shows a simple way of detecting whether a child of a non-stable concept from $\mathcal{B}(I)$ is stable. It also describes the role of fixpoints of operators ${}^{\square\boxtimes}$ and ${}_{\square\boxtimes}$.

Proposition 9. *Let $c \in \mathcal{B}(I)$ be non-stable. Then*

- *$c^{\square}$ is non-stable iff c is a fixpoint of ${}^{\square\boxtimes}$,*
- *$c_{\square}$ is non-stable iff c is a fixpoint of ${}_{\square\boxtimes}$.*

Proof. If $c^{\square}$ is not stable, then $c = (c^{\square})^{\boxtimes}$ by Theorem 8. On the other hand, if $c^{\square}$ is stable, then $c^{\square\boxtimes} = c^{\square}$ by Theorem 6, which rules out $c^{\square\boxtimes} = c$, because in that case c would be equal to $c^{\square}$, which would make it stable by Theorem 5.

The proof for $c_{\square}$ is dual. □

Example 1. In Fig. 1 we can see some examples of contexts with concepts of different types w.r.t. operators ${}^{\square\boxtimes}, {}_{\square\boxtimes}$.

The method is utilized in Algorithm 1.

Algorithm 1 Transforming $\mathcal{B}(I)$ into $\mathcal{B}(J)$ (without structural information).

```
procedure TransformConcepts(B(I))
    B(J) ← B(I);
    for all c = ⟨A, B⟩ ∈ [γ_I(x_0), μ_I(y_0)] do
        B(J) ← B(J) \ {c};
        if c = c_{□⊠} then
            B(J) ← B(J) ∪ {c_□};
        end if
        if c = c^{□⊠} then
            B(J) ← B(J) ∪ {c^□};
        end if
    end for
    return B(J);
end procedure
```

Time complexity of Algorithm 1 is clearly $O(|\mathcal{B}(I)||X||Y|)$ in the worst case scenario. Indeed, the number of non-stable concepts is at most equal to $|\mathcal{B}(I)|$ and the computation of operators ${}^{\square\boxtimes}, {}_{\square\boxtimes}$ can be done in $O(|X| \cdot |Y|)$ time.

5 Computing $\mathcal{B}(J)$ with structural information

To analyze changes in the structure of a concept lattice after removing an incidence, we need to investigate deeper properties of the closure operator ${}^{\square\boxtimes}$ and the interior operator ${}_{\square\boxtimes}$ and the sets of their fixpoints.

	y_0	y_1	y_2
x_0	•	×	×
x_1			
x_2			

(a) The least concept is not stable and is a fixpoint of both operators.

	y_1	y_2	y_3	y_0
x_0	×	×	×	•
x_1			×	×
x_2		×		×
x_3	×			×

(b) Several non-trival non-stable concepts are fixpoints of both operators.

	y_1	y_2	y_0
x_0		×	•
x_1		×	×
x_2		×	

(c) Concept $\langle\{x_0, x_1\}, \{y_0, y_2\}\rangle$ is a fixpoint of $^{\square\boxtimes}$, but not $^{\square\boxtimes}$.

	y_0	y_1	y_2
x_0	•	×	
x_1	×	×	×
x_2			

(d) Concept $\langle\{x_0, x_1\}, \{y_0, y_1\}\rangle$ is a fixpoint of $^{\square\boxtimes}$, but not $^{\square\boxtimes}$.

	y_0	y_1	y_2
x_0	•		
x_1	×	×	
x_2			

(e) Concept $\langle\{x_0, x_1\}, \{y_0\}\rangle$ is not a fixpoint of any operator.

	y_1	y_2	y_3	y_4	y_0
x_0		×		×	•
x_1			×	×	×
x_2				×	
x_3	×	×			×
x_4		×			

(f) Two concepts are not fixpoints of any operator.

Fig. 1: Examples of contexts with concepts of different types w.r.t. operators $^{\square\boxtimes}$, $^{\square\boxtimes}$.

Proposition 10. *Each stable concept is a fixpoint of both* $^{\square\boxtimes}$ *and* $^{\square\boxtimes}$.

Proof. Follows directly from Theorem 5 and Theorem 6. □

Since $^{\square\boxtimes}$ is an interior operator and $^{\square\boxtimes}$ is a closure operator on $\mathcal{B}(I)$, we have for each $c \in \mathcal{B}(I)$, $c^{\square\boxtimes} \leq c \leq c^{\square\boxtimes}$. Thus, we can consider the interval $[c^{\square\boxtimes}, c^{\square\boxtimes}] \subseteq \mathcal{B}(I)$.

Proposition 11. *For any* $c \in \mathcal{B}(I)$*, each concept from* $[c^{\square\boxtimes}, c^{\square\boxtimes}] \setminus \{c\}$ *is stable.*

Proof. First we prove that either $c^{\square\boxtimes}$ equals c, or is its upper neighbor. Let $c = \langle A, B\rangle$. By definition, the intent of $c^{\square\boxtimes}$ is equal to $A^{\uparrow_J\downarrow_I\uparrow_I}$. By Proposition 1, $A^{\uparrow_J} \in \{B, B \setminus \{y_0\}\}$. Thus, $A^{\uparrow_J\downarrow_I\uparrow_I} \in \{B, B \setminus \{y_0\}\}$. If it equals B, then $c^{\square\boxtimes} = c$. Otherwise the intents of c and $c^{\square\boxtimes}$ differ in exactly one attribute, which makes c and $c^{\square\boxtimes}$ neighbors. Also notice that in this case $c^{\square\boxtimes}$ is stable because its intent does not contain y_0 (Proposition 2).

Now let $c' \leq c^{\square\boxtimes}$ be non-stable. If $c = c^{\square\boxtimes}$, then $c' \leq c$. If $c < c^{\square\boxtimes}$, then c is non-stable (Proposition 10) whereas $c^{\square\boxtimes}$ is stable. Non-stable concepts in $\mathcal{B}(I)$

form an interval (Theorem 5). Thus, $c' \vee c$ is non-stable and should be less than $c^{\square\boxtimes}$. Hence, $c' \vee c = c$ (c is a lower neighbor of $c^{\square\boxtimes}$), concluding $c' \leq c$ again.

In a similar way we obtain the inequality $c' \geq c$ for each non-stable $c' \geq c_{\square\boxtimes}$. □

The following proposition shows an important property of the sets of fixpoints w.r.t. the ordering on $\mathcal{B}(I)$: The set of fixpoints of $^{\square\boxtimes}$ is a lower set whereas the set of fixpoints of $_{\square\boxtimes}$ is an upper set.

Proposition 12. *Let $c \in \mathcal{B}(I)$ be a non-stable concept. If c is a fixpoint of $^{\square\boxtimes}$, then each $c' \leq c$ is also a fixpoint of $^{\square\boxtimes}$. If c is a fixpoint of $_{\square\boxtimes}$, then each $c' \geq c$ is also a fixpoint of $_{\square\boxtimes}$.*

Proof. Let $c = c^{\square\boxtimes}$ and $c' \leq c$. If c' is stable, then the assertion follows by Proposition 10. Suppose c' is not stable. By extensivity and isotony of $^{\square\boxtimes}$, $c' \leq {c'}^{\square\boxtimes} \leq c^{\square\boxtimes} = c$. Thus, ${c'}^{\square\boxtimes}$ is not stable (Proposition 2) and ${c'}^{\square\boxtimes} = c'$ by Proposition 11.

The case $c = c_{\square\boxtimes}$ is dual. □

The above results are used in Algorithm 2, which computes the lattice $\mathcal{B}(J)$ together with the information of its ordering. The algorithm is more complicated than the previous one. We provide a short description of the algorithm, together with some examples. Due to space limitations, we will not dwell into details. We will also leave out dual parts of similar cases.

The algorithm processes all non-stable concepts of $\mathcal{B}(I)$ in a bottom-up direction, using an arbitrary linear ordering $\sqsubseteq$ such that if $c_1 \leq c_2$, then $c_1 \sqsubseteq c_2$. Each concept is either modified (by removing x_0 from the extent or y_0 from intent), or disposed of entirely. Sometimes, new concepts are created. All concepts also get updated their lists of upper and lower neighbors.

Let $c = \langle A, B\rangle$ be an arbitrary non-stable concept from $\mathcal{B}(I)$ ($c \in [\gamma_I(x_0), \mu_I(y_0)]$).

- If $c = c^{\square\boxtimes}$, $c = c_{\square\boxtimes}$, then c will "split" into $d_1 \leq d_2$.
  - We set $d_1 = c_{\square}$ and $d_2 = c^{\square}$.
  - The concept d_1 will be a lower neighbor of d_2.
  - If for a lower neighbor c_l of c it holds $c_l = {c_l}^{\square\boxtimes}$, $c_l \neq c_{l\square\boxtimes}$, then it will be a lower neighbor of d_2. It is necessary to check whether d_1 and $c_{l\square\boxtimes}$ will be neighbors. It certainly holds $c_{l\square\boxtimes} \leq d_1$, but there can be a concept k, such that $c_{l\square\boxtimes} \leq k \leq d_1$.
  - Dually for upper neighbors.
  - If for a non-stable neighbor c_n of c it holds $c_n = {c_n}^{\square\boxtimes}$, $c_n = c_{n\square\boxtimes}$, i.e., the same conditions as for c (c_n will split into d_{n_1}, d_{n_2}), then d_1, d_{n_1} and d_2, d_{n_2} will be neighbors.
  - All other upper (resp. lower) neighbors will be neighbors of d_2 (resp. d_1).
- If $c = c^{\square\boxtimes}$ and $c \neq c_{\square\boxtimes}$, then c will lose y_0 from its intent.
  - Denote the transformed c as $d = \langle C, D\rangle = c^{\square} = \langle A, B \setminus \{y_0\}\rangle$.

  - If for an upper neighbor c_u it holds $c_u = c_{u\boxtimes}$, $c_u \neq {c_u}^{\square\boxtimes}$ (c_u will lose x_0 from its extent), then c_u and d will become incomparable. It is necessary to check whether $c_{\square\boxtimes}, c_u$ and $c, {c_u}^{\square\boxtimes}$ should be neighbors (again, there can be a concept between them).
- If $c \neq c^{\square\boxtimes}$ and $c = c_{\square\boxtimes}$, then c will lose x_0 from its extent.
  - Denote transformed c as $d = \langle C, D\rangle = c_{\square} = \langle A \setminus \{x_0\}, B\rangle$.
- If $c \neq c^{\square\boxtimes}$ and $c \neq c_{\square\boxtimes}$, then c will vanish entirely.
  - It is necessary to check whether $c_{\square\boxtimes}$ and $c^{\square\boxtimes}$ should be neighbors (again, a concept can lie between them).
  - Denote by U the set of all upper neighbors of c, except for $c^{\square\boxtimes}$. There is no fixed point of $^{\square\boxtimes}$ among the elements from U.
  - Denote by L the set of all lower neighbors of c, except for $c_{\square\boxtimes}$.
  - Concepts from U and L will not be neighbors.
    Concepts will either become incomparable or one of them or both will vanish. There is also no need for additional checks regarding neighborhood relationship between concepts from U and $c_{\square\boxtimes}$ (resp. L and $c^{\square\boxtimes}$) or their neighbors.
  - It holds $\forall c_l \in L:\ c_l \leq c \leq c^{\square\boxtimes}$, but it is necessary to check if there is a concept between them.
  - Similarly, it holds $\forall c_u \in U : c_{\square\boxtimes} \leq c \leq c_u$, but again it is necessary to check if there is a concept between them.

The number of iterations in TransformConceptLattice is at most $|\mathcal{B}(I)|$, which occurs when each concept in $\mathcal{B}(I)$ is non-stable. In each of the iterations, tests $c = c^{\square\boxtimes}$ and $c = c_{\square\boxtimes}$ are performed and one of the procedures SplitConcept, RelinkReducedIntent, UnlinkVanishedConcept is called. It can be easily seen that the tests can be performed quite efficiently and do not add to the time complexity.

The most time consuming among the above three procedures is SplitConcept. It iterates through all upper (which can be bounded by $|X|$) and lower (which can be bounded by $|Y|$) neighbors of the concept c. For each of the neighbors it might be necessary to check if the interval between the neighbor and certain other concept is empty (and we should make a new edge). This can be done by checking intents/extents of its neighbors.

The above considerations lead to the result that time complexity of Algorithm 2 is in the worst case $O(|\mathcal{B}| \cdot |X|^2 \cdot |Y|)$.

Example 2. In Fig. 2, we can see some examples of transformations of non-stable concepts from $\mathcal{B}(I)$ into concepts of $\mathcal{B}(J)$.

In Algorithm 2 we will assume that following functions are already defined:

- $UpperNeighbors(c)$ - returns upper neighbors of c;
- $LowerNeighbors(c)$ - returns lower neighbors of c;
- $Link(c_1, c_2)$ - introduces neighborhood relationship between c_1 and c_2;
- $Unlink(c_1, c_2)$ - cancels neighborhood relationship between c_1 and c_2.

Algorithm 2 Transforming $\mathcal{B}(I)$ with structural information into $\mathcal{B}(J)$.

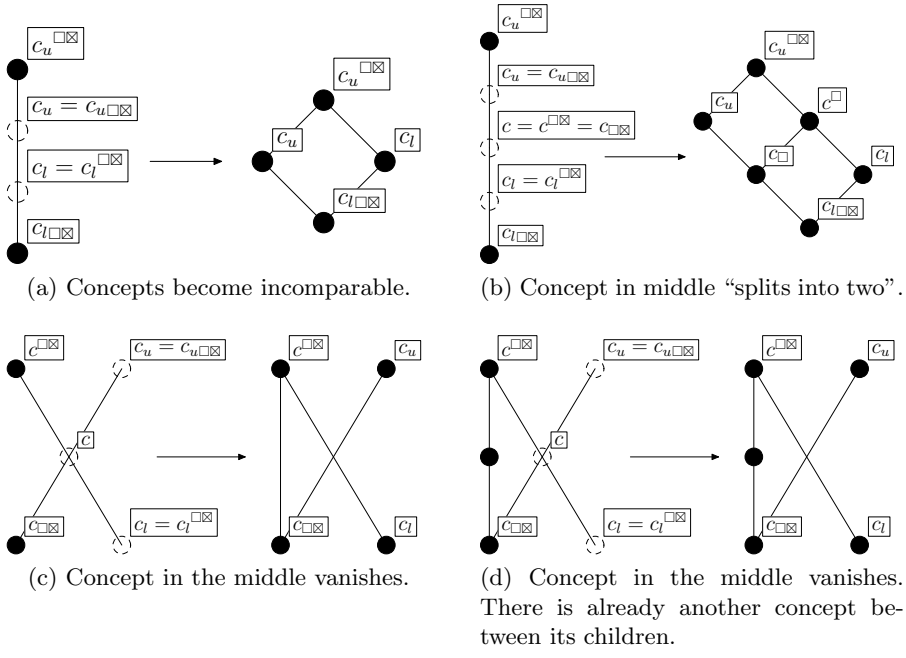
procedure LinkIfNeeded(c_1, c_2)
  if $\nexists k \in \mathcal{B}(I) : c_1 < k < c_2$ **then**
    $Link(c_1, c_2)$;
  end if
end procedure

procedure SplitConcept($c \in [\gamma_I(x_0), \mu_I(y_0)]$)
  $d_1 = c_{\square}$; $d_2 = c^{\square}$;
  $Link(d_1, d_2)$;
  for all $u \in UpperNeighbors(c)$ **do**
    $Unlink(c, u)$; $Link(d_2, u)$;
  end for
  for all $l \in LowerNeighbors(c)$ **do**
    $Unlink(l, c)$; $Link(l, d_1)$;
  end for
  for all $u \in UpperNeighbors(c)$ **do**
    if $u \neq u^{\square\boxtimes}$ **then**
      $Unlink(d_2, u)$; $Link(d_1, u)$; $LinkIfNeeded(d_2, u^{\square\boxtimes})$;
    end if
  end for
  for all $l = \langle C, D\rangle \in LowerNeighbors(c)$ **do**
    if $y_0 \notin D$ **then**
      $Unlink(l, d_1)$; $Link(l, d_2)$; $LinkIfNeeded(l_{\boxtimes\square}, d_1)$;
    end if
  end for
  return d_1, d_2;
end procedure

procedure RelinkReducedIntent($c \in [\gamma_I(x_0), \mu_I(y_0)]$)
  for all $u = \langle C, D\rangle \in UpperNeighbors(c)$ **do**
    if $u \neq u^{\square\boxtimes}$ **then**
      $Unlink(c, u)$;
      $LinkIfNeeded(c_{\square\boxtimes}, u)$; $LinkIfNeeded(c, u^{\square\boxtimes})$;
    end if
  end for
end procedure

procedure UnlinkVanishedConcept($c \in [\gamma_I(x_0), \mu_I(y_0)]$)
  for all $u \in UpperNeighbors(c)$ **do**
    $Unlink(c, u)$; $LinkIfNeeded(c_{\square\boxtimes}, u)$;
  end for
  for all $l \in LowerNeighbors(c)$ **do**
    $Unlink(l, c)$;
  end for
end procedure

procedure TransformConceptLattice($\mathcal{B}(I)$)
  for all $c = \langle A, B\rangle \in [\gamma_I(x_0), \mu_I(y_0)]$ from least to largest w.r.t. $\sqsubseteq$ **do**
    if $c = c^{\square\boxtimes}$ **and** $c = c_{\square\boxtimes}$ **then** ▷ Concept will split.
      $\mathcal{B}(I) \leftarrow \mathcal{B}(I) \setminus \{c\}$;
      $\mathcal{B}(I) \leftarrow \mathcal{B}(I) \cup SplitConcept(c)$;
    else if $c \neq c^{\square\boxtimes}$ **and** $c = c_{\square\boxtimes}$ **then** ▷ Extent will be smaller.
      $A \leftarrow A \setminus \{x_0\}$;
    else if $c = c^{\square\boxtimes}$ **and** $c \neq c_{\square\boxtimes}$ **then** ▷ Intent will be smaller.
      $RelinkReducedIntent(c)$;
      $B \leftarrow B \setminus \{y_0\}$;
    else if $c \neq c^{\square\boxtimes}$ **and** $c \neq c_{\square\boxtimes}$ **then** ▷ Concept will vanish.
      $\mathcal{B}(I) \leftarrow \mathcal{B}(I) \setminus \{c\}$;
      $UnlinkVanishedConcept(c)$;
    end if
  end for
end procedure



(a) Concepts become incomparable.

(b) Concept in middle "splits into two".

(c) Concept in the middle vanishes.

(d) Concept in the middle vanishes. There is already another concept between its children.

Fig. 2: Examples of transformations of non-stable concepts from $\mathcal{B}(I)$ into concepts of $\mathcal{B}(J)$.

6 Conclusion

We analyzed changes of the structure of a concept lattice, caused by removal of exactly one incidence from the associated formal context. We proved some theoretical results and presented two algorithms with time complexities $O(|\mathcal{B}| \cdot |X| \cdot |Y|)$ (Algorithm 1; without structure information) and $O(|\mathcal{B}| \cdot |X|^2 \cdot |Y|)$ (Algorithm 2; with structure information).

There exist several algorithms for incremental computation of concept lattice [1, 5, 8, 6, 7, 2], based on addition and/or removal of objects. Our approach is new in that we recompute a concept lattice based on a removal of just one incidence.

Note that the algorithm proposed by Nourine and Raynaud in [7] has time complexity $O((|Y| + |X|) \cdot |X| \cdot |\mathcal{B}|)$, which is better than complexity of our Algorithm 2. However, experiments presented in [5] indicate that this algorithm sometimes performs slower than some algorithms with time complexity $O(|\mathcal{B}| \cdot |X|^2 \cdot |Y|)$. In the case of our Algorithm 2, some preliminary experiments indicate that the size of the interval of non-stable concepts is usually relatively small, which substantially reduces the overall processing time of the algorithm.

A natural next step would be investigate adding incidences to a formal context, instead of removing. This problem, however, seems to be more difficult than the first one, namely because the set of non-stable concepts in the lattice $\mathcal{B}(J)$ has more complicated structure (it is not an interval) and also because not

all non-stable concepts in $\mathcal{B}(I)$ can be computed via the operator $^{\boxtimes}$. We will try to address this issues in the future. We will also focus on the following:

- experimenting with proposed algorithms on various datasets and comparing them with other known algorithms,
- generalizing the results to allow removing and adding more incidences at the same time.

References

1. Carpineto, C., Romano, G.: Concept Data Analysis: Theory and Applications. John Wiley & Sons (2004)
2. Dowling, C.E.: On the irredundant generation of knowledge spaces. J. Math. Psychol. 37(1), 49–62 (1993)
3. Ganter, B., Wille, R.: Formal Concept Analysis – Mathematical Foundations. Springer (1999)
4. Kuznetsov, S.O., Obiedkov, S.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental and Theoretical Artificial Intelligence 14, 189–216 (2002)
5. Merwe, D., Obiedkov, S., Kourie, D.: Addintent: A new incremental algorithm for constructing concept lattices. In: Eklund, P. (ed.) Concept Lattices, Lecture Notes in Computer Science, vol. 2961, pp. 372–385. Springer Berlin Heidelberg (2004)
6. Norris, E.M.: An algorithm for computing the maximal rectangles in a binary relation. Revue Roumaine de Mathématiques Pures et Appliquées 23(2), 243–250 (1978)
7. Nourine, L., Raynaud, O.: A fast algorithm for building lattices. Inf. Process. Lett. 71(5-6), 199–204 (1999)
8. Outrata, J.: A lattice-free concept lattice update algorithm based on *CbO. In: Ojeda-Aciego, M., Outrata, J. (eds.) CLA. CEUR Workshop Proceedings, vol. 1062, pp. 261–274. CEUR-WS.org (2013)
9. Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Rival, I. (ed.) Ordered Sets, pp. 445–470. Boston (1982)

Formal L-concepts with Rough Intents*

Eduard Bartl and Jan Konecny

Data Analysis and Modeling Lab
Dept. Computer Science, Palacky University, Olomouc
17. listopadu 12, CZ-77146 Olomouc
Czech Republic

Abstract. We provide a new approach to synthesis of Formal Concept Analysis and Rough Set Theory. In this approach, the formal concept is considered to be a collection of objects accompanied with two collections of attributes—those which are shared by all the objects and those which are possessed by at least one of the objects. We define concept-forming operators for these concepts and describe their properties. Furthermore, we deal with reduction of the data by rough approximation by given equivalence. The results are elaborated in a fuzzy setting.

1 Introduction

Formal concept analysis (FCA) [12] is a method of relational data analysis identifying interesting clusters (formal concepts) in a collection of objects and their attributes (formal context), and organizing them into a structure called concept lattice. Numerous generalizations of FCA, which allow to work with graded data, were provided; see [19] and references therein.

In a graded (fuzzy) setting, two main kinds of concept forming-operators—antitone and isotone one—were studied [2, 13, 20, 21], compared [7, 8] and even covered under a unifying framework [4, 18]. We describe concept-forming operators combining both isotone and antitone operators in such a way that each formal (fuzzy) concept is given by two sets of attributes. The first one is a *lower intent approximation*, containing attributes shared by all objects of the concept; the second one is an *upper intent approximation*, containing those attributes which are possessed by at least one object of the concept. Thus, one can consider the two intents to be a lower and upper approximation of attributes possessed by an object.

Several authors dealing with synthesis of FCA and Rough Set Theory have noticed that intents formed by isotone and antitone operators (in both, crisp and fuzzy setting) correspond to upper and lower approximations, respectively (see e.g. [15, 16, 24]). To the best of our knowledge, no one has studied concept-forming operators which would provide both approximations being present in one concept lattice.

In this papers we present such concept-forming operators, structure of their concepts, and reduction of the data by means of rough approximations by equivalences. Due to page limitation we omit proofs of some theorems.

* Supported by grant no. P202/14-11585S of the Czech Science Foundation.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

2 Preliminaries

In this section we summarize the basic notions used in the paper.

Residuated Lattices and Fuzzy Sets We use complete residuated lattices as basic structures of truth-degrees. A complete residuated lattice [1, 14, 23] is a structure $\mathbf{L} = \langle L, \wedge, \vee, \otimes, \rightarrow, 0, 1\rangle$ such that $\langle L, \wedge, \vee, 0, 1\rangle$ is a complete lattice, i.e. a partially ordered set in which arbitrary infima and suprema exist; $\langle L, \otimes, 1\rangle$ is a commutative monoid, i.e. $\otimes$ is a binary operation which is commutative, associative, and $a \otimes 1 = a$ for each $a \in L$; $\otimes$ and $\rightarrow$ satisfy adjointness, i.e. $a \otimes b \leqslant c$ iff $a \leqslant b \rightarrow c$. 0 and 1 denote the least and greatest elements. The partial order of $\mathbf{L}$ is denoted by $\leqslant$. Throughout this work, $\mathbf{L}$ denotes an arbitrary complete residuated lattice.

Elements a of L are called truth degrees. Operations $\otimes$ (multiplication) and $\rightarrow$ (residuum) play the role of (truth functions of) "fuzzy conjunction" and "fuzzy implication". Furthermore, we define the complement of $a \in L$ as $\neg a = a \rightarrow 0$.

An $\mathbf{L}$-set (or fuzzy set) A in a universe set X is a mapping assigning to each $x \in X$ some truth degree $A(x) \in L$. The set of all $\mathbf{L}$-sets in a universe X is denoted $\mathbf{L}^X$, or $\mathbf{L}^X$ if the structure of $\mathbf{L}$ is to be emphasized.

The operations with $\mathbf{L}$-sets are defined componentwise. For instance, the intersection of $\mathbf{L}$-sets $A, B \in \mathbf{L}^X$ is an $\mathbf{L}$-set $A \cap B$ in X such that $(A \cap B)(x) = A(x) \wedge B(x)$ for each $x \in X$. An $\mathbf{L}$-set $A \in \mathbf{L}^X$ is also denoted $\{{}^{A(x)}\!/\!_x \mid x \in X\}$. If for all $y \in X$ distinct from $x_1, \ldots, x_n$ we have $A(y) = 0$, we also write $\{{}^{A(x_1)}\!/\!_{x_1}, \ldots, {}^{A(x_n)}\!/\!_{x_n}\}$.

An $\mathbf{L}$-set $A \in \mathbf{L}^X$ is called normal if there is $x \in X$ such that $A(x) = 1$, and it is called crisp if $A(x) \in \{0, 1\}$ for each $x \in X$. Crisp $\mathbf{L}$-sets can be identified with ordinary sets. For a crisp A, we also write $x \in A$ for $A(x) = 1$ and $x \notin A$ for $A(x) = 0$.

Binary $\mathbf{L}$-relations (binary fuzzy relations) between X and Y can be thought of as $\mathbf{L}$-sets in the universe $X \times Y$. That is, a binary $\mathbf{L}$-relation $I \in \mathbf{L}^{X \times Y}$ between a set X and a set Y is a mapping assigning to each $x \in X$ and each $y \in Y$ a truth degree $I(x, y) \in L$ (a degree to which x and y are related by I). For $\mathbf{L}$-relation $I \in \mathbf{L}^{X \times Y}$ we define its transpose $I^{\mathrm{T}} \in \mathbf{L}^{Y \times X}$ as $I^{\mathrm{T}}(y, x) = I(x, y)$ for all $x \in X$, $y \in Y$.

The composition operators are defined by

$$(I \circ J)(x, z) = \bigvee_{y \in Y} I(x, y) \otimes J(y, z),$$
$$(I \triangleleft J)(x, z) = \bigwedge_{y \in Y} I(x, y) \rightarrow J(y, z),$$
$$(I \triangleright J)(x, z) = \bigwedge_{y \in Y} J(y, z) \rightarrow I(x, y)$$

for every $I \in \mathbf{L}^{X \times Y}$ and $J \in \mathbf{L}^{Y \times Z}$.

A binary **L**-relation E is called an **L**-equivalence if it satisfies $\mathrm{Id}_X \subseteq E$ (reflexivity), $E = E^{\mathrm{T}}$ (symmetry), $E \circ E \subseteq E$ (transitivity).

An **L**-set $B \in \mathbf{L}^Y$ is compatible w.r.t. **L**-equivalence $E \in \mathbf{L}^{Y \times Y}$ if

$$B(y_1) \otimes E(y_1, y_2) \leqslant B(y_2).$$

for any $y_1, y_2 \in Y$.

Formal Concept Analysis in the Fuzzy Setting An **L**-context is a triplet $\langle X, Y, I \rangle$ where X and Y are (ordinary) sets and $I \in \mathbf{L}^{X \times Y}$ is an **L**-relation between X and Y. Elements of X are called objects, elements of Y are called attributes, I is called an incidence relation. $I(x, y) = a$ is read: "The object x has the attribute y to degree a." An **L**-context may be described as a table with the objects corresponding to the rows of the table, the attributes corresponding to the columns of the table and $I(x, y)$ written in cells of the table (for an example see Fig. 1).

	α	β	γ	δ
A	0.5	0	1	0
B	1	0.5	1	0.5
C	0	0.5	0.5	0.5
D	0.5	0.5	1	0.5

Fig. 1. Example of **L**-context with objects A,B,C,D and attributes $\alpha, \beta, \gamma, \delta$.

Consider the following pairs of operators induced by an **L**-context $\langle X, Y, I \rangle$. First, the pair $\langle \uparrow, \downarrow \rangle$ of operators $^{\uparrow} : \mathbf{L}^X \to \mathbf{L}^Y$ and $^{\downarrow} : \mathbf{L}^Y \to \mathbf{L}^X$ is defined by

$$A^{\uparrow}(y) = \bigwedge_{x \in X} A(x) \to I(x, y), \quad B^{\downarrow}(x) = \bigwedge_{y \in Y} B(y) \to I(x, y). \tag{1}$$

Second, the pair $\langle \cap, \cup \rangle$ of operators $^{\cap} : \mathbf{L}^X \to \mathbf{L}^Y$ and $^{\cup} : \mathbf{L}^Y \to \mathbf{L}^X$ is defined by

$$A^{\cap}(y) = \bigvee_{x \in X} A(x) \otimes I(x, y), \quad B^{\cup}(x) = \bigwedge_{y \in Y} I(x, y) \to B(y). \tag{2}$$

To emphasize that the operators are induced by I, we also denote the operators by $\langle \uparrow_I, \downarrow_I \rangle$ and $\langle \cap_I, \cup_I \rangle$. Fixpoints of these operators are called formal concepts. The set of all formal concepts (along with set inclusion) forms a complete lattice, called **L**-concept lattice. We denote the sets of all concepts (as well as the corresponding **L**-concept lattice) by $\mathcal{B}^{\uparrow\downarrow}(X, Y, I)$ and $\mathcal{B}^{\cap\cup}(X, Y, I)$, i.e.

$$\begin{aligned} \mathcal{B}^{\uparrow\downarrow}(X, Y, I) &= \{\langle A, B \rangle \in \mathbf{L}^X \times \mathbf{L}^Y \mid A^{\uparrow} = B,\ B^{\downarrow} = A\}, \\ \mathcal{B}^{\cap\cup}(X, Y, I) &= \{\langle A, B \rangle \in \mathbf{L}^X \times \mathbf{L}^Y \mid A^{\cap} = B,\ B^{\cup} = A\}. \end{aligned} \tag{3}$$

For an **L**-concept lattice $\mathcal{B}(X, Y, I)$, where $\mathcal{B}$ is either $\mathcal{B}^{\uparrow\downarrow}$ or $\mathcal{B}^{\cap\cup}$, denote the corresponding sets of extents and intents by $\mathrm{Ext}(X, Y, I)$ and $\mathrm{Int}(X, Y, I)$. That is,

$$\begin{aligned} \mathrm{Ext}(X, Y, I) &= \{A \in \mathbf{L}^X \mid \langle A, B\rangle \in \mathcal{B}(X, Y, I) \text{ for some } B\}, \\ \mathrm{Int}(X, Y, I) &= \{B \in \mathbf{L}^Y \mid \langle A, B\rangle \in \mathcal{B}(X, Y, I) \text{ for some } A\}. \end{aligned} \tag{4}$$

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

In $\mathcal{B}^{\cap\cup}(X, Y, I)$, for any $x \in X$, $y \in Y$ and formal **L**-concept $\langle A, B\rangle$ we have $A(x) \geqslant a$ and $B(y) \leqslant b$ if and only if there is a formal concept $\langle A_1, B_1\rangle \leqslant \langle A, B\rangle$, labeled by ${}^{a}\!/\!x$ and a formal concept $\langle A_2, B_2\rangle \geqslant \langle A, B\rangle$, labeled by ${}^{b}\!/\!y$ (see examples depicted in Fig. 2).

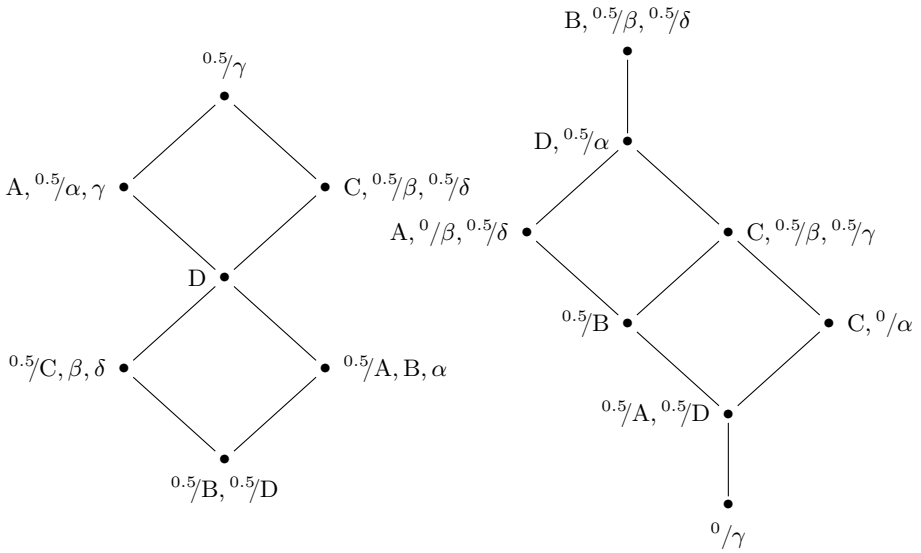


Fig. 2. Concept lattice $\mathcal{B}^{\uparrow\downarrow}(X, Y, I)$ (left) and $\mathcal{B}^{\cap\cup}(X, Y, I)$ (right) of the **L**-context in Fig. 1.

3 L-rough concepts

We consider concept-forming operators induced by **L**-context $\langle X, Y, I \rangle$ defined as follows:

Definition 1. *Let* $\langle X, Y, I \rangle$ *be an* **L***-context. Define* **L**-rough concept-forming operators *as*

$$A^{\triangle} = \langle A^{\uparrow}, A^{\cap} \rangle \quad \textit{and} \quad \langle \underline{B}, \overline{B} \rangle^{\triangledown} = \underline{B}^{\downarrow} \cap \overline{B}^{\cup}$$

for $A \in \mathbf{L}^X, \underline{B}, \overline{B} \in \mathbf{L}^Y$. **L**-rough concept *is then a fixed point of* $\langle \triangle, \triangledown \rangle$*, i.e. a pair* $\langle A, \langle \underline{B}, \overline{B} \rangle \rangle \in \mathbf{L}^X \times (\mathbf{L} \times \mathbf{L})^Y$ *such that* $A^{\triangle} = \langle \underline{B}, \overline{B} \rangle$ *and* $\langle \underline{B}, \overline{B} \rangle^{\triangledown} = A$.[1] $A^{\uparrow}$ *and* $A^{\cap}$ *are called* lower intent approximation *and* upper intent approximation*, respectively.*

That means, $^{\triangle}$ gives intents w.r.t. both $\langle \uparrow, \downarrow \rangle$ and $\langle \cap, \cup \rangle$; $^{\triangledown}$ then gives intersection of extents related to the corresponding intents.

We denote the set of all fixed-points of $\langle \triangle, \triangledown \rangle$, in correspondence with (3), as $\mathcal{B}^{\triangle\triangledown}(X, Y, I)$ and call it **L**-*rough concept lattice*. Below, we present an analogy of the Main theorem on concept lattices for **L**-rough setting.

Theorem 1 (Main theorem on L-rough concept lattices).

(a) **L***-rough concept lattice* $\mathcal{B}^{\triangle\triangledown}(X, Y, I)$ *is a complete lattice with suprema and infima defined as follows*

$$\bigwedge_i \langle A_i, \underline{B}_i, \overline{B}_i \rangle = \langle \bigcap_i A_i, \langle \bigcup_i \underline{B}_i, \bigcap_i \overline{B}_i \rangle^{\triangledown\triangle} \rangle,$$
$$\bigvee_i \langle A_i, \underline{B}_i, \overline{B}_i \rangle = \langle (\bigcup_i A_i)^{\triangle\triangledown}, \bigcap_i \underline{B}_i, \bigcup_i \overline{B}_i \rangle.$$

(b) Moreover, a complete lattice $\mathbf{V} = \langle V, \leqslant \rangle$ *is isomorphic to* $\mathcal{B}^{\triangle\triangledown}(X, Y, I)$ *iff there are mappings*

$$\gamma : X \times L \to V \quad \textit{and} \quad \mu : Y \times L \times L \to V$$

such that $\gamma(X \times L)$ *is supremally dense in* $\mathbf{V}$*,* $\mu(Y \times L \times L)$ *is infimally dense in* $\mathbf{V}$*, and*

$$a \otimes \underline{b} \leqslant I(x, y) \textit{ and } I(x, y) \leqslant a \to \overline{b} \quad \textit{is equivalent to} \quad \gamma(x, a) \leqslant \mu(y, \underline{b}, \overline{b})$$

for all $x \in X, y \in Y, a, \underline{b}, \overline{b} \in L$.

When drawing a concept lattice we label nodes as in $\mathcal{B}^{\uparrow\downarrow}$ for lower intent approximations and $\mathcal{B}^{\cap\cup}$ for upper intent approximations. We write $\frac{a}{}y$ or $\frac{\overline{a}}{}y$ instead of just $\frac{a}{}y$ to distinguish them. Fig. 3 (middle) shows an **L**-rough concept lattice for the **L**-context from Fig. 1.

The following theorem explains that normal extents have natural intent approximations; that is $\underline{B} \subseteq \overline{B}$.

[1] In what follows, we naturally identify $\langle A, \langle \underline{B}, \overline{B} \rangle \rangle$ with $\langle A, \underline{B}, \overline{B} \rangle$.

Theorem 2. *For normal $A \in \mathbf{L}^X$, we have $A^{\uparrow} \subseteq A^{\cap}$, for crisp singleton $A \in \mathbf{L}^X$, we have $A^{\uparrow} = A^{\cap}$.*

Proof. Since A is normal, there is $x_1 \in X$ such that $A(x_1) = 1$. Then we have

$$\begin{aligned} A^{\uparrow}(y) = \bigwedge_{x \in X} A(x) \to I(x,y) \leqslant A(x_1) \to I(x_1,y) = I(x_1,y) = \\ = A(x_1) \otimes I(x_1,y) \leqslant \bigvee_{x \in X} A(x) \otimes I(x,y) = A^{\cap}(y) \end{aligned} \tag{5}$$

for each $y \in Y$.

For A being a crisp singleton, one can show $A^{\uparrow} = A^{\cap}$ by changing all inequalities in (5) to equalities. □

Since $\langle \triangle, \triangledown \rangle$ is defined via $\langle \uparrow, \downarrow \rangle$ and $\langle \cap, \cup \rangle$, one can expect that there is a strong relationship between the associated concept lattices. In the rest of this section, we summarize them.

Theorem 3. *For $\mathcal{S} \subseteq \mathbf{L}^X$, let $[\mathcal{S}]$ denote an $\mathbf{L}$-closure span of $\mathcal{S}$, i.e. the smallest $\mathbf{L}$-closure system containing $\mathcal{S}$. We have*

$$[\mathrm{Ext}^{\uparrow\downarrow}(X,Y,I) \cup \mathrm{Ext}^{\cap\cup}(X,Y,I)] = \mathrm{Ext}^{\triangle\triangledown}(X,Y,I).$$

Proof. "$\subseteq$": Let $A \in \mathrm{Ext}^{\uparrow\downarrow}(X,Y,I)$. Then $A = A \cap X = \langle A^{\uparrow}, Y \rangle^{\triangledown} \in \mathrm{Ext}^{\triangle\triangledown}(X,Y,I)$. Similarly for $A \in \mathrm{Ext}^{\cap\cup}(X,Y,I)$.

"$\supseteq$": Let $A \in \mathrm{Ext}^{\triangle\triangledown}(X,Y,I)$ and let $\langle B_1, B_2 \rangle = A^{\triangle}$. Then we have $A = B^{\downarrow} \cap B^{\cup} \in [\mathrm{Ext}^{\uparrow\downarrow}(X,Y,I) \cup \mathrm{Ext}^{\cap\cup}(X,Y,I)]$ since $B^{\downarrow} \in \mathrm{Ext}^{\uparrow\downarrow}(X,Y,I)$ and $B^{\cup} \in \mathrm{Ext}^{\cap\cup}(X,Y,I)$.

From Theorem 3 one can observe that no extent from $\mathrm{Ext}^{\uparrow\downarrow}(X,Y,I)$ and $\mathrm{Ext}^{\cap\cup}(X,Y,I)$ is lost.

Corollary 1. $\mathrm{Ext}^{\uparrow\downarrow}(X,Y,I) \subseteq \mathrm{Ext}^{\triangle\triangledown}(X,Y,I)$ *and* $\mathrm{Ext}^{\cap\cup}(X,Y,I) \subseteq \mathrm{Ext}^{\triangle\triangledown}(X,Y,I)$.

In addition, no concept is lost.

Corollary 2. *For each $\langle A, \underline{B} \rangle \in \mathcal{B}^{\uparrow\downarrow}(X,Y,I)$ there is $\langle A, \underline{B}, A^{\cap} \rangle \in \mathcal{B}^{\triangle\triangledown}(X,Y,I)$.*

For each $\langle A, \overline{B} \rangle \in \mathcal{B}^{\cap\cup}(X,Y,I)$ there is $\langle A, A^{\uparrow}, \overline{B} \rangle \in \mathcal{B}^{\triangle\triangledown}(X,Y,I)$.

Remark 1. One can observe from Fig. 3 that in $\mathrm{Ext}^{\triangle\triangledown}(X,Y,I)$ there exist extents which are present neither in $\mathrm{Ext}^{\uparrow\downarrow}(X,Y,I)$ nor in $\mathrm{Ext}^{\cap\cup}(X,Y,I)$. On the other hand, lower intent approximations are exactly those from $\mathrm{Int}^{\uparrow\downarrow}(X,Y,I)$ and upper intent approximations are exactly those from $\mathrm{Int}^{\cap\cup}(X,Y,I)$.

With results on mutual reducibility from [8] we can state the following theorem on representation of $\mathcal{B}^{\triangle\triangledown}$ by $\mathcal{B}^{\uparrow\downarrow}$.

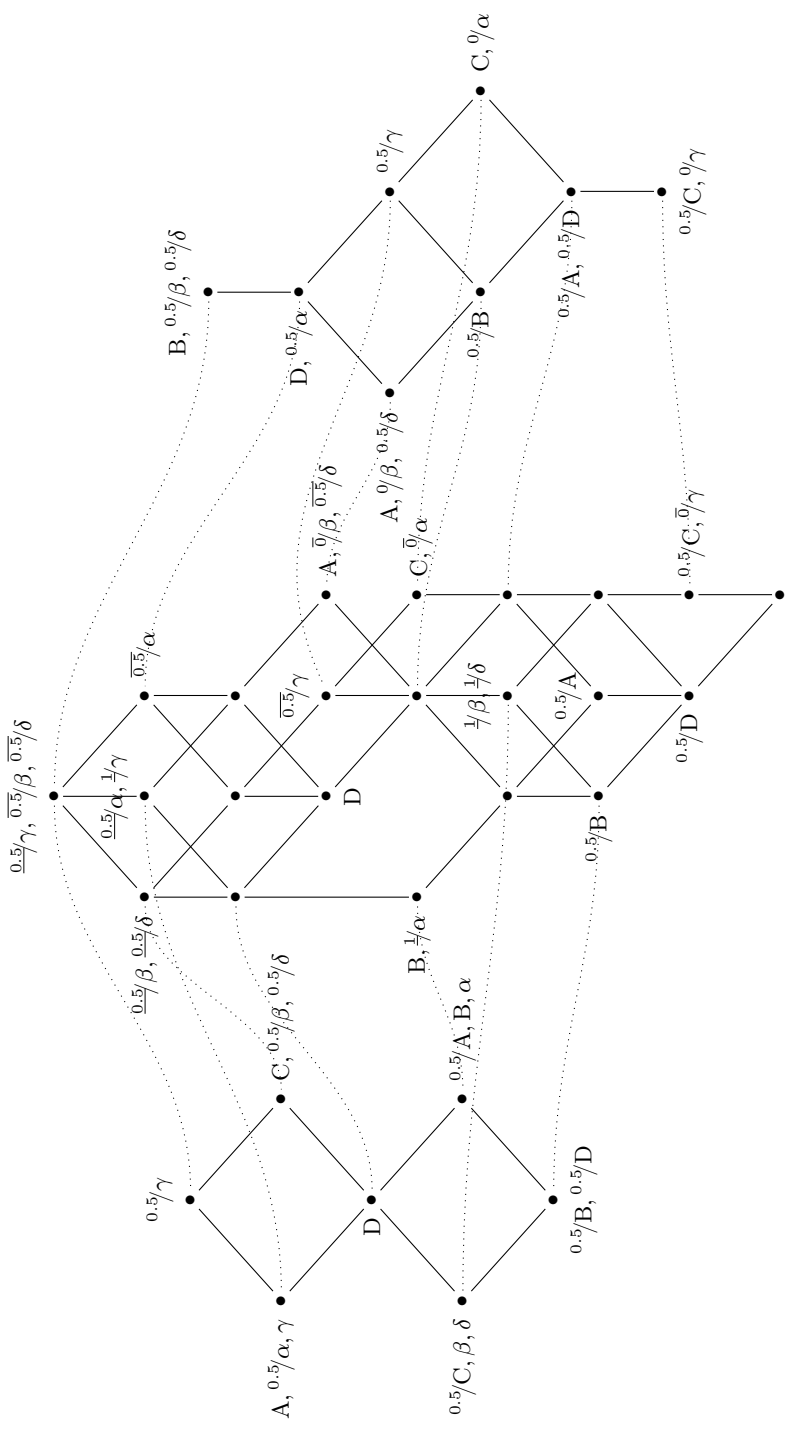


Fig. 3. $\mathcal{B}^{\triangle\triangledown}(X, Y, I)$ (middle) and positions of original concepts in $\mathcal{B}^{\uparrow\downarrow}(X, Y, I)$ (left) and $\mathcal{B}^{\cap\cup}(X, Y, I)$ (right) with $\mathbf{L}$ being a three-element Lukasiewicz chain

Theorem 4. *For a* $\mathbf{L}$*-context* $\langle X, Y, I\rangle$*, consider the* $\mathbf{L}$*-context* $\langle X, Y \times L, J\rangle$ *where* J *is defined by*

$$J(x, \langle y, a\rangle) = \begin{cases} I(x,y) & \text{if } a = 1, \\ I(x,y) \to a & \text{otherwise.} \end{cases}$$

Then we have that $\mathcal{B}^{\uparrow\downarrow}(X, Y \times L, J)$ *is isomorphic to* $\mathcal{B}^{\Delta\nabla}(X, Y, I)$ *as a lattice. In addition,*

$$\mathrm{Ext}^{\uparrow\downarrow}(X, Y \times L, J) = \mathrm{Ext}^{\Delta\nabla}(X, Y, I).$$

Proof (sketch). In [8] we show that for $\mathbf{L}$-contexts $\langle X, Y, I\rangle$ and $\langle X, Y \times L\backslash\{1\}, J\rangle$ such that

$$J(x, \langle y, a\rangle) = I(x, y) \to a$$

it holds that $\mathrm{Ext}^{\cap\cup}(X, Y, I) = \mathrm{Ext}^{\uparrow\downarrow}(X, Y \times L\backslash\{1\}, J)$. Using this fact, one can check that mapping i defined as

$$i(\langle A, \underline{B}, \overline{B}\rangle) \mapsto \langle A, \underline{B}' \cup \overline{B}'\rangle,$$

where $\underline{B}' \in L^{Y\times\{1\}}, \overline{B}' \in L^{Y\times L\backslash\{1\}}$ with

$$\underline{B}'(\langle y, 1\rangle) = \underline{B}(y),$$
$$\overline{B}'(\langle y, a\rangle) = \overline{B}(y) \to a,$$

is the desired isomorphism from $\mathcal{B}^{\Delta\nabla}(X, Y, I)$ to $\mathcal{B}^{\uparrow\downarrow}(X, Y \times L, J)$.

Theorem 4 shows how we can obtain a concept lattice formed by $\langle\uparrow, \downarrow\rangle$ which is isomorphic to $\mathbf{L}$-rough concept lattice of given $\mathbf{L}$-context.

4 Rough approximation of an L-context and L-concept lattice

In [17] Pawlak introduced Rough Set Theory where uncertain elements are approximated with respect to an equivalence relation representing indiscernibility.

Formally, given *Pawlak approximation space* $\langle U, E\rangle$, where U is a non-empty set of *objects* (universe) and E is an equivalence relation on U, the *rough approximation* of a crisp set $A \subseteq U$ by E is the pair $\langle A^{\Downarrow_E}, A^{\Uparrow_E}\rangle$ of sets in U defined by

$$x \in A^{\Downarrow_E} \quad \text{iff} \quad \text{for all } y \in U, \langle x, y\rangle \in E \text{ implies } y \in A,$$
$$x \in A^{\Uparrow_E} \quad \text{iff} \quad \text{there exists } y \in U \text{ such that } \langle x, y\rangle \in E \text{ and } y \in A.$$

$A^{\Downarrow_E}$ and $A^{\Uparrow_E}$ are called *lower and upper approximation* of the set A by E, respectively.

In the fuzzy setting, one can generalize $\langle A^{\Downarrow_E}, A^{\Uparrow_E}\rangle$ as in [10, 11, 22],

$$A^{\Downarrow_E}(x) = \bigwedge_{y\in U}(E(x,y) \to A(y)),$$
$$A^{\Uparrow_E}(x) = \bigvee_{y\in U}(A(y) \otimes E(x,y))$$

for **L**-equivalence $E \in \mathbf{L}^{U\times U}$ and **L**-set $A \in \mathbf{L}^U$.

Considering **L**-context $\langle U, U, E\rangle$, we can easily see that $^{\Downarrow_E}$ is equivalent to $^{\cup_E}$; and $^{\Uparrow_E}$ is equivalent to $^{\cap_{E^{\mathrm{T}}}}$. Since E is symmetric, we can also write

$$\langle \Downarrow_E, \Uparrow_E\rangle = \langle \cup_E, \cap_E\rangle. \tag{6}$$

Note that for **L**-set A, $A^{\Downarrow_E}$ is its largest subset compatible with E and $A^{\Uparrow_E}$ is its smallest superset compatible with E.

Below, we deal with situation where lower and upper intent approximations are further approximated using Pawlak's approach. In other words, instead of lower intent approximation $A^{\uparrow}$ we consider the largest subset of $A^{\uparrow}$ compatible with a given indiscernibility relation E, and similarly, instead of upper intent approximation $A^{\cap}$ we consider its smallest superset compatible with E. In Theorem 5 we show how to express this setting using **L**-rough concept forming operators.

Definition 2. *Let $\langle X, Y, I\rangle$ be an* **L***-context, E be an* **L***-equivalence on Y. Define* **L**-rough concept-forming operators *as follows:*

$$A^{\Delta_E} = \langle A^{\uparrow\Downarrow_E}, A^{\cap\Uparrow_E}\rangle,$$
$$\langle \underline{B}, \overline{B}\rangle^{\nabla_E} = \underline{B}^{\Uparrow_E\downarrow} \cap \overline{B}^{\Downarrow_E\cup}.$$

Directly from (6) and results in [5] we have:

Theorem 5. *Let $\langle X, Y, I\rangle$ be an* **L***-context, E be an* **L***-equivalence on Y. We have*

$$A^{\Delta_E} = \langle A^{\uparrow_{I\triangleright E}}, A^{\cap_{I\circ E}}\rangle \quad \text{and} \quad \langle \underline{B}, \overline{B}\rangle^{\nabla_E} = \underline{B}^{\downarrow_{I\triangleright E}} \cap \overline{B}^{\cup_{I\circ E}}.$$

Again, for normal extents we obtain natural upper and lower intent approximations.

Theorem 6. *For normal $A \in \mathbf{L}^X$ we have $A^{\uparrow_{I\triangleright E}} \subseteq A^{\cap_{I\circ E}}$.*

In correspondence with (3) and (4), we denote set of the set of fixpoints of $\langle \triangle_E, \nabla_E\rangle$ in **L**-context $\langle X, Y, I\rangle$ by $\mathcal{B}^{\Delta_E\nabla_E}(X, Y, I)$ and set of its extents and intents by $\mathrm{Ext}^{\Delta_E\nabla_E}(X, Y, I)$ and $\mathrm{Int}^{\Delta_E\nabla_E}(X, Y, I)$, respectively.

The following theorem shows that a use of a rougher **L**-equivalence relation leads to a reduction of size of the **L**-rough concept lattices. Furthermore, this reduction is natural, i.e. it preserves extents.

Theorem 7. *Let $\langle X, Y, I \rangle$ be an* **L***-context, and E_1, E_2 be* **L***-equivalences on Y, such that $E_1 \subseteq E_2$. Then*

$$\mathrm{Ext}^{\triangle_{E_2}\nabla_{E_2}}(X, Y, I) \subseteq \mathrm{Ext}^{\triangle_{E_1}\nabla_{E_1}}(X, Y, I).$$

Example 1. Fig. 4 shows **L**-rough concept lattice of the **L**-context in Fig. 1 and rough **L**-concept lattice approximated using the following **L**-equivalence relation on Y.

	α	β	γ	δ
α	1	0.5	0	0
β	0.5	1	0	0
γ	0	0	1	0.5
δ	0	0	0.5	1

To demonstrate Theorem 7, the concepts with the same extents in the two lattices are connected.

5 Conclusions and further research

We proposed a novel approach to synthesis of RTS na FCA. It provides a lot of directions to be further explored. Our future research includes:

Study of attribute implications using whose semantics is related to the present setting. That will combine results on fuzzy attribute implications [9] and attribute containment formulas [6].

Generalization of the current setting. Note that the operators $^{\uparrow}$ and $^{\cap}$ which compute the universal and the existential intent, need not be induced by the same relation to keep most of the described properties. Actually, this feature is used in Section 4. In our future research, we want to elaborate more on this. For instance, it can provide interesting solution of problem of missing values in a formal fuzzy context—the idea is to use $^{\uparrow}$ induced by the context with missing values substituted by 0, and $^{\cap}$ induced by the context with missing values substituted by 1.

Reduction of **L***-rough concept lattice via linguistic hedges* As two intents are considered in each **L**-rough concept, the size of concept lattice can grow very large. The RST approach to reduction of data, i.e. use of rougher **L**-relation, directly leads to reduction of **L**-rough concept lattice as we showed in Theorem 7. FFCA provides other ways to reduce the size, one of them is parametrization of concept-forming operators using hedges.

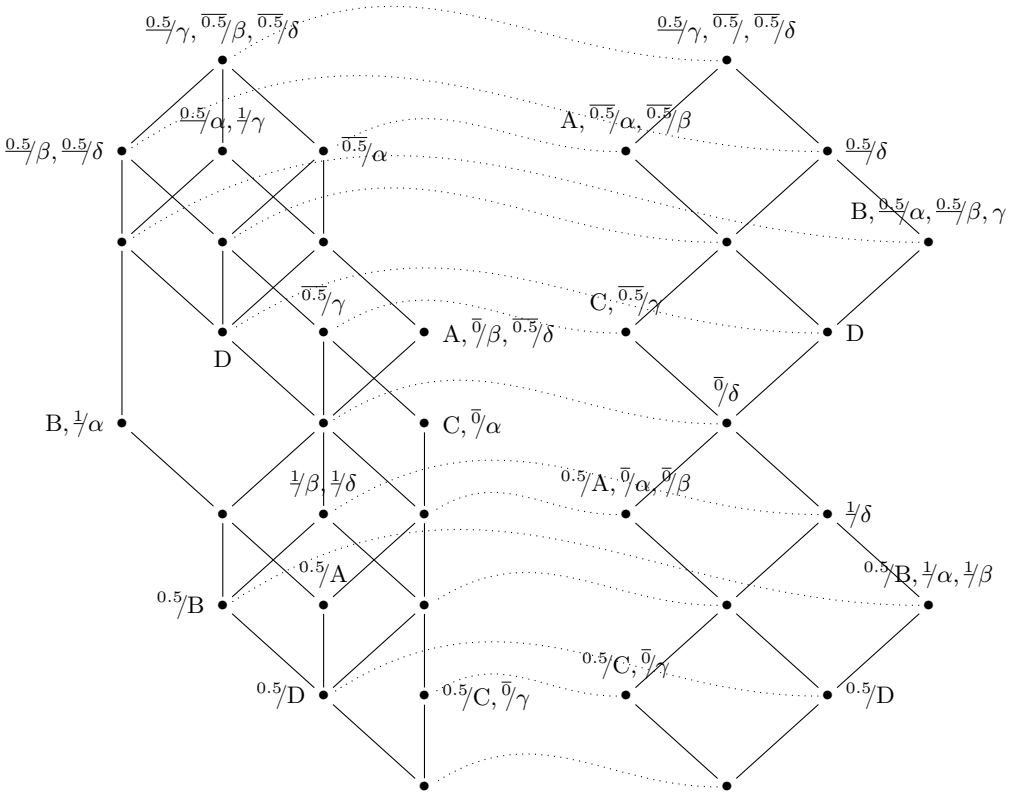


Fig. 4. Rough **L**-concept lattices $\mathcal{B}^{\Delta\nabla}(X, Y, I)$ (left) and $\mathcal{B}^{\Delta E^{\nabla E}}(X, Y, I)$ (right) with **L** being three-element Łukasiewicz chain. The corresponding extents are connected.

References

1. Radim Belohlavek. *Fuzzy Relational Systems: Foundations and Principles*. Kluwer Academic Publishers, Norwell, USA, 2002.
2. Radim Belohlavek. Concept lattices and order in fuzzy logic. *Ann. Pure Appl. Log.*, 128(1-3):277–298, 2004.
3. Radim Belohlavek. Optimal decompositions of matrices with entries from residuated lattices. Submitted to *J. Logic and Computation*, 2009.
4. Radim Belohlavek. Sup-t-norm and inf-residuum are one type of relational product: Unifying framework and consequences. *Fuzzy Sets Syst.*, 197:45–58, June 2012.
5. Radim Belohlavek and Jan Konecny. Operators and spaces associated to matrices with grades and their decompositions. In *NAFIPS 2008*, pages 288–293.
6. Radim Belohlavek and Jan Konecny. A logic of attribute containment, KAM '08 Proceedings of the 2008 International Symposium on Knowledge Acquisition and Modeling, pages 246–251.

7. Radim Belohlavek and Jan Konecny. Closure spaces of isotone galois connections and their morphisms. In *Proceedings of the 24th international conference on Advances in Artificial Intelligence*, AI'11, pages 182–191, Springer-Verlag, Berlin, Heidelberg, 2011.
8. Radim Belohlavek and Jan Konecny. Concept lattices of isotone vs. antitone Galois connections in graded setting: Mutual reducibility revisited. *Information Sciences*, 199: 133 – 137, 2012.
9. Radim Belohlavek and Vilem Vychodil. A logic of graded attributes. *submitted to Artificial Intelligence*.
10. Didier Dubois and Henri Prade. Rough fuzzy sets and fuzzy rough sets. *International Journal of General Systems*, 17(2–3):191–209, 1990.
11. Didier Dubois and Henri Prade. Putting rough sets and fuzzy sets together. In Roman Słowiński, editor, *Intelligent Decision Support*, volume 11 of *Theory and Decision Library*, pages 203–232. Springer Netherlands, 1992.
12. Bernard Ganter and Rudolf Wille. *Formal Concept Analysis – Mathematical Foundations*. Springer, 1999.
13. George Georgescu and Andrei Popescu. Non-dual fuzzy connections. *Arch. Math. Log.*, 43(8):1009–1039, 2004.
14. Petr Hájek. *Metamathematics of Fuzzy Logic (Trends in Logic)*. Springer, November 2001.
15. Robert E. Kent. Rough concept analysis: A synthesis of rough sets and formal concept analysis. *Fundam. Inf.*, 27(2,3):169–181, August 1996.
16. Hongliang Lai and Dexue Zhang. Concept lattices of fuzzy contexts: Formal concept analysis vs. rough set theory. *International Journal of Approximate Reasoning*, 50(5):695 – 707, 2009.
17. Zdzisław Pawlak. Rough sets. *International Journal of Computer & Information Sciences*, 11(5):341–356, 1982.
18. Jesus Medina, Manuel Ojeda-Aciego, and Jorge Ruiz-Calvino. Formal concept analysis via multi-adjoint concept lattices. *Fuzzy Sets and Systems*, 160(2):130–144, January 2009.
19. Jonas Poelmans, Dmitry I. Ignatov, Sergei O. Kuznetsov, and Guido Dedene. Fuzzy and rough formal concept analysis: a survey. *International Journal of General Systems*, 43(2):105–134, 2014.
20. Silke Pollandt. *Fuzzy Begriffe: Formale Begriffsanalyse von unscharfen Daten*. Springer–Verlag, Berlin–Heidelberg, 1997.
21. Andrei Popescu. A general approach to fuzzy concepts. *Mathematical Logic Quarterly*, 50(3):265–280, 2004.
22. Anna M. Radzikowska and Etienne E. Kerre. Fuzzy rough sets based on residuated lattices. In James F. Peters, Andrzej Skowron, Didier Dubois, Jerzy W. Grzymała-Busse, Masahiro Inuiguchi, and Lech Polkowski, editors, *Transactions on Rough Sets II*, volume 3135 of *Lecture Notes in Computer Science*, pages 278–296. Springer Berlin Heidelberg, 2005.
23. Morgan Ward and Robert P. Dilworth. Residuated lattices. *Transactions of the American Mathematical Society*, 45:335–354, 1939.
24. Yiyu Yao. On unifying formal concept analysis and rough set analysis. Xi'an Jiaotong University Press, 2006.

Reduction dimension of bags of visual words with FCA

Ngoc Bich Dao, Karell Bertet, Arnaud Revel

Laboratoire L3i, University of La Rochelle, France

Abstract. In image retrieval involving bag of visual words, reduction dimension is a fundamental task of data preprocessing. In recent years, several methods have been proposed for supervised and unsupervised cases. In the supervised case, the problem has been addressed with encouraging results. However, in the unsupervised case, reduction dimension is still an unavoidable challenge. In this article, we propose an application of a logic reduction dimension method which is based on Formal Concept Analysis for image retrieval. This method is the reduction of a closure system without, theoretically, loss of information. In our context, combining our proposed method with bag of visual words is original. Experimental results on five data sets such as COREL, CALTECH256, VOC2005, VOC2012 and MIR flickr are analyzed to show the influence of the data structures and the parameters on the reduction factor.

1 Introduction

Thanks to the generalization of multimedia devices, huge collections of digital images are available today. As far as mining in multimedia documents is concerned, web search engines usually give poor results. Hence, such results are far from expected regarding the semantics of the documents. Content Based Image Retrieval (CBIR)[1] has been investigated in order to give an answer to this problem for decades. The main idea is to build a description based on the image content, and to find similarities between descriptions. Classically, visual features are extracted from images and then compiled into an index or signature to give a dense description of images. To perform the retrieval, a similarity function is computed to compare the index of the query with those of collection. A ranking of the results according to the calculated similarity is proposed to the users. The detection of visual features can be performed by a SIFT detector[2] or a dense grid which both select an important number of interest points (up to several thousands) from the images. Each of these points is then described thanks to a SIFT-like descriptor. However, to limit the dimension of the description space, a vector quantization (usually k-means) is performed in order to cluster similar interest points into "visual words", and to generate a dictionary of "visual words" (usually up to 1000 words). Then, the signature of the image is composed of the set of all the visual words corresponding to each feature point detected into the image (what formed a "bag of visual words"[3]). The comparison between the images then consists in comparing the bags of visual words of each image

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

in a dataset. The processing cost introduced by these techniques makes them difficult to use with large amounts of images such as a query on the Internet.

On the other hand, supervised data is labeled (the data has ground truth) and classification methods are required to deal with the categorization problem. Data in the case unsupervised is unlabeled, hence clustering methods are used to gather the similar observations in the same cluster. There are many applications for classification and clustering on many domains of computer science such as bioinformatics, numerical analysis, machine learning, data mining, pattern recognition, etc., where data may contain a grand set of features, means the description of the data is high dimension, and therefore it need to be reduced. However, reduction them while preserving the quality of the data is still challenging.

To be able to manage high dimensional description spaces, reduction techniques have been proposed. These techniques are much used as a data preprocessing step in machine learning and pattern recognition. This step can usually increase the accuracy of the results in the next steps such as classification or clustering while the computational cost and time cost of the former step may be significantly decreased. Regarding statistics and machine learning literature, we distinguish two main strategies: feature extraction and feature selection. These methods can be used for supervised case or unsupervised case. The main idea of feature transformation consists in transforming the given set of features into a new one. In case that the size of the new feature set is greater than the original feature set, we called it the feature generation. And when new feature set size is smaller than the original feature set, feature extraction is mentioned. Feature selection methods propose a manipulation of data to select features from the original set. This approach is interesting in some domains when they prefer the existing features in order to maintain their physical properties.

In this article, we propose a logic and unsupervised feature reduction method issued from FCA to address the visual word reduction problem in a CBIR system. In FCA, data are organised into a "context" by a set of observations (called "objects", "samples" or "experimental units" in other fields) and a set of features (also known as "attributes", "parameters", or "variables" in computer science, machine learning and statistic communities) that are associated with each observation.

Context reduction is a simple and polynomial treatment in FCA classically applied on the whole context, thus both reducing observations and features. This treatment is based on a nice result establishing that the concept lattice of the context can be reduced to a minimal one while preserving its graph structure by deleting some redundant observations and features. For example, when two attributes are shared by the same objects, then they belong to the same concepts of the concept lattice, thus they are redundant and one of these two objects can be deleted while preserving the concept lattice structure. In our case, we focus on feature reduction of a context. Our algorithm accepts as input the closure operator of the context on attributes set, and returns the redundant attributes. Thus, this algorithm extends the classical attributes reduction of a context to the

more general case of data described by a closure operator. Moreover, we propose a new application in image analysis for features reduction of visual words.

This paper is organized as follows: In order to introduce our approach, we recall some definitions of formal concepts in the section 2.1. Section 2.2 shows details our proposed method. Section 3 shows some experimental results with real data. Finally, section 4 ends this paper with a conclusion and perspectives.

2 The proposed features selection method

The feature reduction algorithm we propose is a logic and unsupervised method stemming from FCA where a concept lattice, defining from a binary table, represents the description of all object-attribute combinations. When the concept lattice structure is preserved after the deletion of some attributes and objects, then these attributes are "redundant" for the lattice structure and can be deleted from the initial data without affecting the structure of object-attributes combinations. Therefore, from a theoretical point of view, the description of data is equivalently represented by a concept lattice where "redundant" attributes and objects are deleted.

The reduction is a simple and polynomial treatment in FCA, classically decomposed into two steps: attribute and object reduction. In this article, we focus on attributes/features reduction, thus on the detection of redundant attributes for the concept lattice structure reduced to attributes. A nice result establishes that each subset of a concept (A,B) is a closure defined on the objects and attributes set, and the concept lattice reduced to the attributes/objects is denoted a closure lattice.

In the first subsection, we introduce the notions of closure lattice according to a closure operator, reduced closure lattice and redundant attributes. In the second section, we presents the reduction algorithm aiming at removing redundant attributes, with a closure operator as input. This algorithm is thus a generic algorithm that can be applied either on attributes or on objects of a binary table, but also on any closure system.

2.1 Reduced lattice

In FCA, the relationship between a set of attributes I and a set of objects O are described by a formal context $(O, I, (\alpha, \beta))$ where $\alpha(A)$ the set of attributes sharing by a subset A of objects, and $\beta(B)$ the set of objects sharing a subset B of attributes. One can derive two closure systems from a context. The first one is defined on the set of attributes I, with $\beta \circ \alpha$ as closure operator. The second one is defined on the set of objects O with $\alpha \circ \beta$ as closure operator[18]. A *closure system* (φ, S) is defined by a *closure operator* φ on a set S, i.e. a map on $\mathcal{P}(S)$ satisfying the three following properties: φ is *isotone*, *extensive* and *idempotent*. A subset $X \subseteq S$ is called *closed* if $\varphi(X) = X$ (see Table 2). The set system $\mathcal{F}$ of all closed subsets, fitted out with the inclusion relation $\subseteq$, forms a lattice usually called the *closure lattice* (see Fig. 1a). See the survey of Caspard

and Monjardet[19] for more details about closure systems. There are infinitely set systems whose closure lattice are isomorphic. A *reduced closure lattice* is a closure lattice defined on a set S of the smallest size among all isomorphic closure lattices. A nice result[20,18] establishes that a closure system is reduced when, for each $x \in S$, the closure $\varphi(x)$ is a join irreducible (Equation 1).

$$\forall x \in S, \forall Y \subseteq S \text{ so that } x \notin Y, \text{ then } \varphi(x) \neq \varphi(Y) \tag{1}$$

Therefore, a non-reduced closure system contains reducible elements - elements which do not satisfy Equation 1 - each reducible element $x \in S$ is then equivalent to a set $E_x \subseteq S$ of equivalent elements with $x \notin E_x$ and $\varphi(x) = \varphi(E_x)$. Reducible elements can be removed without affecting the structure of the closure lattice. The reduction of a closure system consists then in removing or replacing each reducible element $x \in S$ by its equivalent set E_x.

2.2 Proposed reduction algorithm

The algorithm we propose is a generic reduction algorithm since it only needs a closure operator as input. Thus it can be applied with the same complexity on any closure system, and in particular on a context by considering the attributes - using $\beta \circ \alpha$ as closure operator.

	a	b	c	d	e	f	g	h
1		×					×	
2	×		×		×		×	
3				×	×	×	×	×
4	×			×	×	×	×	×
5	×				×	×	×	×
6		×				×	×	
7	×					×	×	
8	×				×		×	
9	×	×	×	×	×	×	×	×

(a) The context

	a	b	c	d	e	f
1		×				
2	×		×		×	
3				×	×	×
4	×			×	×	×
5	×				×	×
6		×				×
7	×					×
8	×				×	
9	×	×	×	×	×	×

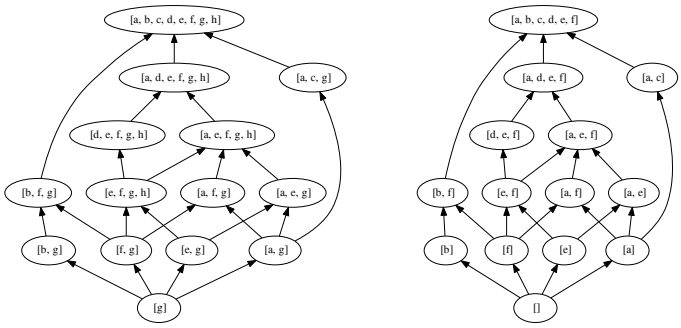
(b) The attribute-reduced context

Table 1: The example of context

x	a	b	c	d	e	f	g	h
$\varphi(x)$	a,g	b,g	a,c,g	d,e,f,g	e,g	f,g	g	e,f,g,h

Table 2: Attributes $x \in S$ and their closure $\varphi(x)$ for the context in Table 1a

A direct application of the definition (see Eq. 1) would imply an exponential cost by checking if any subset $Y \subset S$ is equivalent to each $x \in S$. We use the precedence relation (precedence graph) for a polynomial reduction. The precedence graph is defined on the set S, with an edge between two elements $x, y \in S$



(a) The closure lattice of context in Table 1a

(b) The reduced closure lattice of context in Table 1b

Fig. 1: The example of closure lattices

when $\varphi(x) \subseteq \varphi(y)$. This graph is clearly acyclic for a reduced closure system. We propose a generic algorithm in 3 steps:

Step 1: Standardization. Check if there exists $x, y \in S$ such that $\varphi(x) = \varphi(y)$. When $\varphi(x) = \varphi(y)$, then x and y belong to the same strongly connected components of the graph. Each strongly connected components $X \subseteq S$ include the elements x_i, x_j so that $\varphi(x_i) = \varphi(x_j), \forall x_i \neq x_j \in X$. Thus, we can delete all elements except one representative element $x \in X$ of the component. The obtained precedence graph is then an acyclic graph.

Step 2: Clarification. Check if there exists $x \in S$ such that $\varphi(x) = \varphi(\emptyset)$. When such an x exists, then $\varphi(x)$ is included into $\varphi(y)$ for any $y \in S$, thus x is the only source of the precedence graph. The clarification test has only to be performed for graphs with one source.

Step 3: Reduction. Check, for any $x \in S$, if there exists a set $E_x \subset S$ such that $x \notin E_x$ and $\varphi(x) = \varphi(E_x)$. One can observe that an attribute x with only one immediate predecessor y is not reducible, because it would be equivalent to y, and thus belong to the same strongly connected component already removed in the previous step. If there exists $E_x \subset S$ such that $\varphi(x) = \varphi(E_x)$, then elements of E_x are clearly predecessors of x in the precedence graph since, for $\forall y \in E_x$, $\varphi(x) = \cap\varphi(y)$. Moreover, this test can be reduced to maximal predecessors of x. Therefore, this treatment has only to be performed for elements with more than one immediate predecessors, and the equality has to be checked with the set of immediate predecessors of x.

This algorithm takes into account a closure operator φ on a set S as input. The output of the alforithm is the reducible element set $X \subset S$ and the equivalent elements set E_x for each $x \in X$.

Alg. 1 reduces a closure system in $O(|S|.c_\varphi + |S|^2 \log |S|)$ where c_φ is the cost of a closure generation and —S— is the number of nodes. Indeed, the precedence graph can be initialized in $O(|S|c_\varphi + |S|^2 log|S|)$ by computing the closures in $O(|S|c_\varphi)$, and then comparing two closures in $O(|S|^2 log|S|)$. Then, the SCCs can be computed using Kosaraju's algorithm by two passes of depth first search, thus a complexity in $O(|S| + |A|) \leq O(|S|^2)$, with $|A|$ nb of edges in the graph. Standardization and clarification are clearly in $O(|S|)$ by a simple pass into the graph. Finaly, reduction considers the immediate predecessors of each $x \in S$ in $O(|S|^2)$, and then computes and compare two closures in $O(|S|c_\varphi + |S|^2 log|S|)$. Therefore, Alg. 1 computes the attribute reduced context in $O(|I|^2|O| + |I|^2 log|I|)$. since a closure can be obtained in $O(|I|.|O|)$.

Input: a closure operator φ on a set S
Output: the reducible elements set $X \subset S$, and the equivalent elements set E_x for each $x \in X$
init a set *Res* with $\emptyset$;
init a graph G with S as set of node;
\\ Precedence graph;
foreach $(x, y) \in S \times S$ **do**
  if $\varphi(x) \subseteq \varphi(y)$ **then**
    add the edge (x, y) in G;
  end
end
compute the set CFC of the strongly connected components of G;
let *source* be the sources of the graph G;
\\ Step (1): Standardization;
foreach $C \in CFC$ **do**
  choose $y \in C$;
  foreach $x \in C$ *such that* $x \neq y$ **do**
    add x in *Res* with $E_x = \{y\}$; delete x from the graph G;
  end
end
\\ Step (2): Clarification;
if $|source| = 1$ *and* $\varphi(source) = \varphi(\emptyset)$ **then**
  add *source* in *Res* with $E_{source} = \emptyset$; delete *source* from G;
end
\\ Step (3): Reduction;
foreach $x \in G$ **do**
  let P the set of immediate predecessors x in the graph G;
  if $|P| \neq 1$ *and* $\varphi(x) = \varphi(P)$ **then**
    add x in *Res* with $E_x = P$; delete x from the graph G;
  end
end
return *Res*, $(E_x)_{x \in Res}$;

Algorithm 1: Reduction of a closure system

3 Experimentation

3.1 Datasets

In our experiments, we compare the performance of the method we propose on different image data sets. Each image in a data set is described by a vector composed of the occurrence frequencies of its visual words, where a set of visual words is defined for each data set. Table 3 describes the different data sets we used in our experiments, and the methods applied to generate the whole bag of visual words.

Database	Images nb	Features nb	Detector	Descriptor	Dictionary of visual words
VOC2012[21]	17124	4096	Harris-Laplace	CMI (Colour Moment Invariants)[22]	Random selection of all key points
MIR flickr[23]	24991	4096	Harris-Laplace	CMI[1]	Random selection of all key points
COREL[24]	4998	500	SIFT	SIFT[2]	K-means[25] (OpenCV)
CALTECH 256[26]	30607	500	SIFT	SIFT[2]	K-means (OpenCV)
Dataset 1 (VOC2005)[27]	1354	262	Harris-Laplace and Laplacian[3]	SIFT	K-means (OpenCV)

Table 3: Description of used datasets

3.2 Experimental protocol

As mentioned earlier, the algorithm we propose requires binary values indicating for each object whether it possesses a given attribute or not. Since each image is described by a visual word occurence frequency vector, its values can vary from 0 to a max value depending on the image size and the quantity of visual words in the image. For instance, if an image is black painted, there is only one visual word "black" for the whole image with a big frequency, and the vector

[1] http://koen.me/research/colordescriptors/
[2] http://www.robots.ox.ac.uk/ vgg/research/affine/#software
[3] http://lear.inrialpes.fr/people/dorko/downloads.html

will be sparse. Conversely, an image with a patchwork of colors is described by a frequency vector mainly composed of low but not zero values. To be able to compare several images, it is thus necessary to normalize their frequency vector before binarization.

Normalization As mentioned before, the visual word occurrence frequency can be very important in some images, and insignificant in others. In order to compare the visual words, several strategies can be adopted.

First of all, it is necessary to find out a "max" value in the data set and then divide the visual word frequency by this max value to transform the values in a range 0 to 1. Two manners to define the max value have been considered into this article.

Normalization by line (image) With this type of normalization, a max value is computed for each image as being the maximum frequency value of the corresponding image. The interpretation of this normalization is that we consider as significant the ratio between the different attributes of a given image. This kind of normalization does not depend on the database size and on the image size. However, the normalized values do not account for the ratio measurement of the same attribute between the images in the database.

Normalization by column (feature) Normalization by column finds out the maximum values of the frequency for each attribute in the database. With this approach, the correspondence between the images in the database is taken into account. The drawback is that each time a new image is inserted into the database, the normalized values must be recomputed. Besides, the image size must also be taken into account. Table 4 gives an illustrated example.

	f_1	f_2	f_3	f_4
img_1	1	0	50	5
img_2	10	9	1	8
img_3	0	0	0	99

(a) Initial data

	f_1	f_2	f_3	f_4
img_1	0.02	0	1	0.1
img_2	1	0.9	0.1	0.8
img_3	0	0	0	1

(b) After normalization by line

	f_1	f_2	f_3	f_4
img_1	0.1	0	1	0.05
img_2	1	1	0.02	0.08
img_3	0	0	0	1

(c) After normalization by column

Table 4: Illustration for normalization types

Binarization After the normalization, we simply binarize the normalized values by comparing these values with a threshold varying from 0 to 0.9. At the highest threshold one, in the *normalization by line* case, it is possible that most of the attributes in an image should be below the threshold. To avoid removing all the visual words from an image, the highest threshold has been assigned to 0.9.

Reduction The next phase in the algorithm is to apply our reduction method which is itself composed of three steps (clarification, standardisation, reduction). Indeed, before applying the proposed method to bag of visual words, we must remove all the visual words that appear (resp. do not appear) in each (resp. any) image. This step corresponds to the *clarification*. The *standardization* step reduces the feature that the vector of images of a given feature equivalent to the vector of images of another feature. At last, in the *reduction* step, all the features which are the combination of other features are removed.

3.3 Results

In this section, we detail the results obtained with our reduction method for 5 data sets, described in section 2.2. To analyze the behavior of our method, and the contribution of each step of the algorithm, we introduce the ratio of removed features for each step of the reduction algorithm as follows:

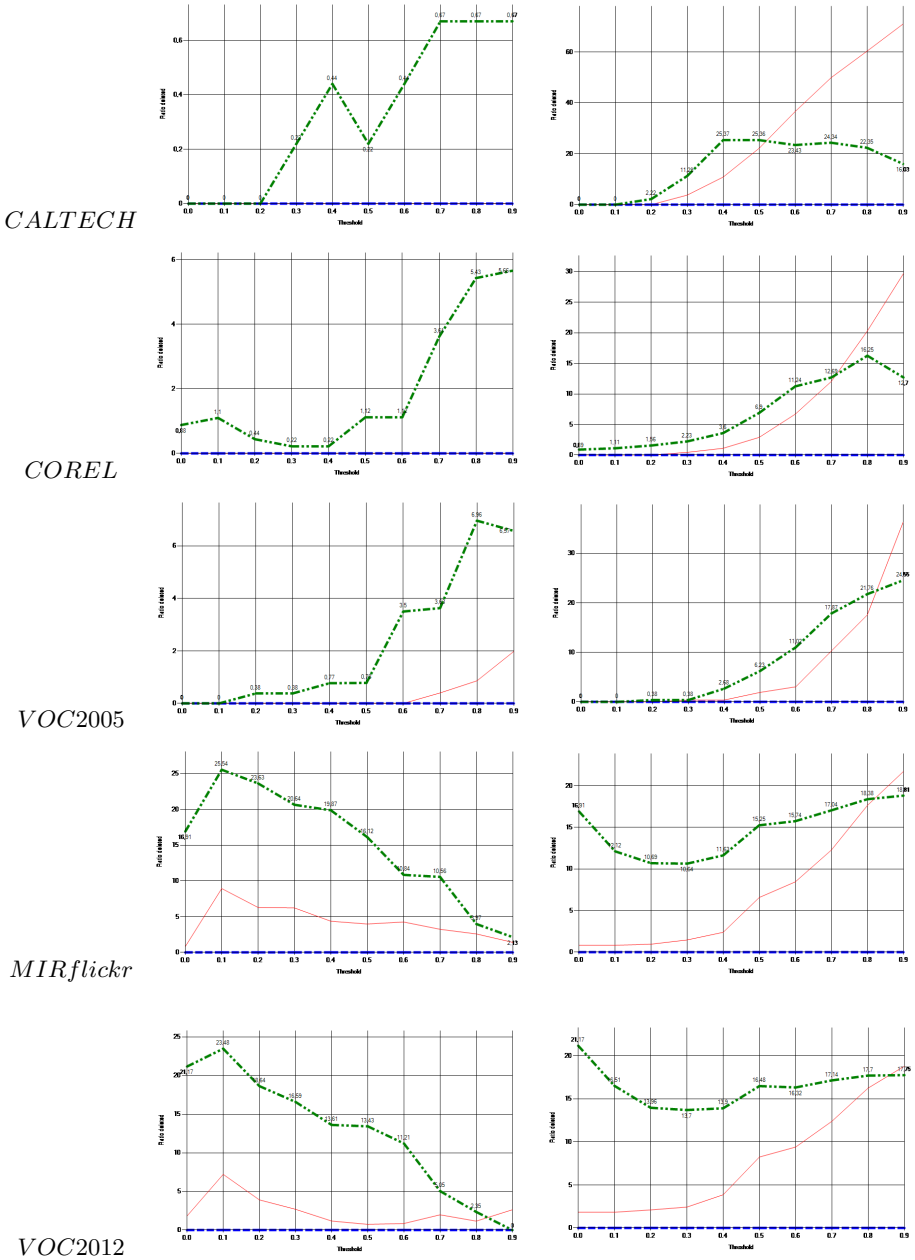
$$\Delta_1 = \frac{a}{N_{att}},\ \Delta_2 = \frac{b}{N_{att}-a},\ \Delta_3 = \frac{c}{N_{att}-a-b}$$

Where a (resp. b and c) is the number of removed attributes in the *standardization* (resp. *clarification* and *reduction*) step; N_{att} is the attribute number in total. Figure 2 shows the evolution of Δ_1, Δ_2, Δ_3 with regard to the threshold level, for both normalization types: line and column.

The maximum ratio of removed attributes of the data sets (CALTECH, COREL, VOC2005, MIRflickr, VOC2012) are approximately equal to 0.67%, 2.6%, 22.5%, 95%, 96% respectively. The impact of the reduction is more interesting in the last three datasets. This phenomenon can be explained by the bag of visual words generation since the two data sets MIR flickr and VOC2012 are composed of randomly selected visual words stemming from the keypoints set. Conversely, the data sets CALTECH, COREL and VOC2005, are composed of bags of visual words defined by the SIFT detector and descriptor, and by a K-means clustering. Thus, the randomly selected visual words are less consistent.

We can also observe that the percentage of removed attributes increases while the binarization threshold increases. With an increasing threshold, only the most frequent words are kept, thus more attributes are potentially equivalent and removed.

At last, there is no attribute reduction in the step 1 (Δ_1 value) with a normalization by column because this kind of normalization can not generate empty columns. Morover, a normalization by line keeps the most frequent attributes in each image whereas a normalization by column keeps the most frequent images for each attribute. To summarize, the number of removed attributes depends both on the visual words generation, on the chosen threshold of binarization and on the normalization process (by line or column). However, care should be taken, that the greater the binarization threshold is, the smaller the number of images remaining. Except in the case normalization by line.



(a) Normalization by line (b) Normalization by column

Fig. 2: The ratio of removed attributes according to the initial attributes corresponding to three cases of proposed method where red line is Δ_1, blue dash is Δ_2 and green dash dot dot is Δ_3.

4 Conclusion and perspective

In this article, we present a logic feature selection method of bags of visual words. This method, stemming from Formal Concept Analysis, is a closure system reduction without, theoretically, loss of information. That means that the data description lattice is preserved by the reduction treatment. In our context, combining our proposed method with a bag of visuals words is original. The experimentations show that the number of deleted features can be interesting, depending on the data set and the binarization treatment. Moreover, it is possible to perform both an object and an attribute reduction.

A finer analysis should be obtained in the supervised case, by comparing classification performance before and after reduction. Moroever, the number of potentially deleted objects could also be usefull to autmatically define a good binarization thresold in the supervised case: while suppression of objects belonging to the same class is to promote, we must avoid removing objects of different classes. Objects reduction can easily be performed by applying our reduction algorithm on the objects set.

At last, we plan to study the number of deleted attributes and deleted objects (of the same class / of different class) to evaluate the complexity of a data set, and the quality of its visuals words.

Acknowledgment: We would like to thank Thierry URRUTY, Nhu Van NGUYEN and Dounia AWAD who extracted the bag of visual words we used in this paper.

References

1. Smeulders, A., Worring, M., Santini, S., Gupta, A., Jain, R.: Content-based image retrieval at the end of the early years. IEEE Transactions on Pattern Analysis and Machine Intelligence **22** (2000) 1349–1380
2. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the International Conference on Computer Vision, Kerkyra (1999) 1150–1157
3. Bosch, A., Zisserman, A., Munoz, X.: Scene Classification Via pLSA. In Leonardis, A., Bischof, H., Pinz, A., eds.: 9th European Conference on Computer Vision. Volume 3954 of Lecture Notes in Computer Science., Graz, Austria, Springer Berlin Heidelberg (2006) 517–530
4. Tufféry, S.: Data mining et statistique décisionnelle: L'intelligence des données. Technip edn. Volume 2010. (2010)
5. Belohlavek, R., Kruse, R., Vychodil, V.: Discovery of optimal factors in binary data via a novel method of matrix decomposition. Journal of Computer and System Sciences **76** (2010) 3–20
6. Fisher, R.A.: The use of multiple measurements in taxonomic problems. The Annals of Eugenics **7** (1936) 179–188
7. Hotelling, H.: Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology **24** (1933) 417–441
8. Yu, L., Liu, H.: Feature selection for high-dimensional data: A fast correlation-based filter solution. In: Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003), Washington DC (2003) 856–863

9. Hall, M.A.: Correlation-based feature subset selection for machine learning. Doctor of philosophy, University of Waikato, Hamilton, NewZealand (1999)
10. Battiti, R.: Using mutual information for selecting features in supervised neural net learning. IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council **5** (1994) 537–550
11. Rakotomalala, R., Lallich, S.: Construction d'arbres de decision par optimisation. Revue Extraction des Connaissances et Apprentissage **16** (2002) 685–703
12. Kononenko, I.: Estimating attributes: Analysis and extensions of RELIEF. In Bergadano, F., Raedt, L., eds.: Machine Learning: ECML-94. Volume 784 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, Berlin, Heidelberg (1994) 171–182
13. He, X., Cai, D., Niyogi, P.: Laplacian score for feature selection. In: Neural Information Processing Systems Foundation, MIT Press (2005)
14. Devaney, M., Ram, A.: Efcient Feature Selection in Conceptual Clustering. In: Machine Learning: Proceedings of the Fourteenth International Conference, Nashville, TN (1997)
15. Dy, J.G., Brodley, C.E.: Feature Selection for Unsupervised Learning. Journal of Machine Learning Research **5** (2004) 845–889
16. Wolf, L., Shashua, A.: Feature Selection for Unsupervised and Supervised Inference: The Emergence of Sparsity in a Weight-Based Approach. The Journal of Machine Learning Research **6** (2005) 1855–1887
17. Elghazel, H., Aussem, A.: Unsupervised feature selection with ensemble learning. Machine Learning (2013)
18. Barbut, M., Monjardet, B.: Ordre et classification: algèbre et combinatoire. Hachette, Paris (1970)
19. Caspard, N., Monjardet, B.: The lattices of closure systems, closure operators, and implicational systems on a finite set: a survey. Discrete Applied Mathematics **127** (2003) 241–269
20. Birkhoff, G.: Lattice Theory. 1st edn. American Mathematical Society (1940)
21. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes (VOC) Challenge (2012)
22. Mindru, F., Tuytelaars, T., Gool, L.V., Moons, T.: Moment invariants for recognition under changing viewpoint and illumination. Computer Vision and Image Understanding **94** (2004) 3–27
23. Huiskes, M.J., Lew, M.S.: The MIR flickr retrieval evaluation. In: Proceeding of the 1st ACM international conference on Multimedia information retrieval - MIR '08, New York, USA, ACM Press (2008) 39–43
24. Carneiro, G., Chan, A.B., Moreno, P.J., Vasconcelos, N.: Supervised Learning of Semantic Classes for Image Annotation and Retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence **29** (2007) 394–410
25. Macqueen, J.B.: Some Methods for classification and Analysis of Multivariate Observations. In: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. (1967) 281–297
26. Griffin, G., Holub, A.D., Perona, P.: Caltech-256 Object Category Dataset. Technical report (2007)
27. Everingham, M., Zisserman, A., Williams, C.K.I., Van Gool, L., Al., A.: The 2005 PASCAL Visual Object Classes Challenge. In: First PASCAL Machine Learning Challenges Workshop, MLCW 2005. Volume 3944 of Lecture Notes in Computer Science., Berlin, Heidelberg, Springer Berlin Heidelberg (2005) 117–176

A One-Pass Triclustering Approach: Is There any Room for Big Data?

Dmitry V. Gnatyshak[1], Dmitry I. Ignatov[1], Sergei O. Kuznetsov[1], and Lhouari Nourine[2]

[1] National Research University Higher School of Economics, Russian Federation
dmitry.gnatyshak@gmail.com
http://www.hse.ru
[2] Blaise Pascal University, LIMOS, CNRS, France
http://www.univ-bpclermont.fr/

Abstract. An efficient one-pass online algorithm for triclustering of binary data (triadic formal contexts) is proposed. This algorithm is a modified version of the basic algorithm for OAC-triclustering approach, but it has linear time and memory complexities with respect to the cardinality of the underlying ternary relation and can be easily parallelized in order to be applied for the analysis of big datasets. The results of computer experiments show the efficiency of the proposed algorithm.

Keywords: Formal Concept Analysis, triclustering, triadic data, data mining, big data

1 Introduction

Cluster analysis of multimodal data and specifically of dyadic and triadic relations is a natural extension of the idea of normal clustering. In dyadic case biclustering methods (the term bicluster was coined by B. Mirkin [17]) are used to simultaneously find subsets of the sets of objects and attributes that form homogeneous patterns of the input object-attribute data. One of the most popular applications of biclustering is gene expression analysis in Bionformatics [16,3]. Triclustering methods operate in triadic case in which for each object-attribute pair one assigns a set of some conditions [18,8,5]. Both biclustering and triclustering algorithms are widely used in such areas as the analysis of gene expression [21,15,13], recommender systems [19,10,9], social networks analysis [6], etc. The processing of numeric multimodal data is also possible by modifications of existing approaches for mining binary relations [12].

Though there are methods that can enumerate all triclusters satisfying certain constraints [1] (in most cases they ensure that triclusters are dense), their time complexity is rather high, as in the worst case the maximal number of triclusters usually is exponential (e.g. in case of formal triconcepts), showing that these methods are hardly scalable. To process big data algorithms need to have at most linear time complexity and be easily parallelizable. Also, in most cases, it is necessary that such algorithms output the results in one pass.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

In order to create an algorithm satisfying these requirements we adapted a triclustering method based on prime operators (prime OAC-triclustering method) [5]. As the result we developed an online version of prime OAC-triclustering method, which is linear, one-pass and easily parallelizable.

The rest of the paper is organized as follows: in Section 2 we recall the method and the basic version of the algorithm of prime OAC-triclustering. In Section 3 we describe the online setting for the problem and the corresponding online version of the basic algorithm with some optimizations. Finally, in Section 4 we show the results of some experiments which demonstrate the efficiency of the online version of the algorithm.

2 Prime object-attribute-condition triclustering method

Prime object-attribute-condition triclustering method based on the framework of Formal Concept Analysis [20,4,2] is an extension for the triadic case of object-attribute biclustering method [7]. Triclusters generated by this method have the same structure as the corresponding biclusters, namely the cross-like structure of triples inside the iput data cuboid (i.e. formal tricontext).

Let $\mathbb{K} = (G, M, B, I)$ be a triadic context, where G, M, B are respectively the sets of objects, attributes, and conditions, and $I \subseteq G \times M \times B$ is a triadic incidence relation. Each prime OAC-tricluster is generated by applying the following prime operators to each pair of components of some triple:

$$\begin{gathered}(X,Y)' = \{b \in B \,|\, (g,m,b) \in I \text{ for all } g \in X, m \in Y\},\\ (X,Z)' = \{m \in M \,|\, (g,m,b) \in I \text{ for all } g \in X, b \in Z\},\\ (Y,Z)' = \{g \in G \,|\, (g,m,b) \in I \text{ for all } m \in Y, b \in Z\}\end{gathered} \tag{1}$$

Then the triple $T = ((m,b)', (g,b)', (g,m)')$ is called *prime OAC-tricluster* based on triple $(g,m,b) \in I$. The components of tricluster are called, respectively, *extent*, *intent*, and *modus*. The triple (g,m,b) is called a *generating triple* of the tricluster T. Figure 2 shows the structure of an OAC-tricluster (X,Y,Z) based on triple $(\widetilde{g}, \widetilde{m}, \widetilde{b})$, triples corresponding to the gray cells are contained in the context, other triples may be contained in the tricluster (cuboid) as well.

The basic algorithm for prime OAC-triclustering method is rather simple (Alg. 1). First of all, for each combination of elements from each two sets of $\mathbb{K}$ we compute the results of applying the corresponding prime operator (we will call the resulting sets *prime sets*). After that we enumerate all triples from I and on each step we must generate a tricluster based on the corresponding triple, check whether this tricluster is already presented in the tricluster set (by using hashing) and also check conditions.

The total time complexity of the algorithm depends on whether there is a non-zero minimal density threshold or not and on the complexity of the hashing algorithm used. In case we use some basic hashing algorithm processing the tricluster's extent, intent and modus and have a minimal density threshold equal to 0, the total time complexity of the main loop is $O(|I|(|G|+|M|+|B|))$, and of

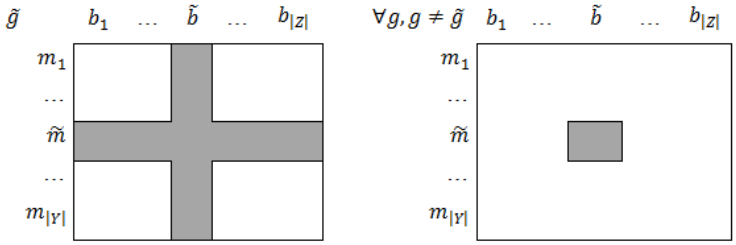


Fig. 1. Structure of prime OAC-triclusters

Algorithm 1 Algorithm for prime OAC-triclustering.

Input: $\mathbb{K} = (G, M, B, I)$ — tricontext;
ρ_{min} — minimal density threshold
Output: $\mathcal{T} = \{T = (X, Y, Z)\}$

$\mathcal{T} := \emptyset$
for all (g, m): $g \in G$,$m \in M$ **do**
  $PrimesOA[g, m] = (g, m)'$
end for
for all (g, b): $g \in G$,$b \in B$ **do**
  $PrimesOC[g, b] = (g, b)'$
end for
for all (m, b): $m \in M$,$b \in B$ **do**
  $PrimesAC[m, b] = (m, b)'$
end for
for all $(g, m, b) \in I$ **do**
  $T = (PrimesAC[m, b], PrimesOC[g, b], PrimesOA[g, m])$
  $Tkey = hash(T)$
  if $Tkey \notin \mathcal{T}.keys \wedge \rho(T) \geq \rho_{min}$ **then**
    $\mathcal{T}[Tkey] := T$
  end if
end for

the whole algorithm is $O(|G||M||B|+|I|(|G|+|M|+|B|))$. If we have a non-zero minimal density threshold, the time complexity of the main loop, as well as the time complexity of the algorithm, is $O(|I||G||M||B|)$.

The memory complexity is $O(|I|(|G|+|M|+|B|))$, as we need to keep the dictionaries with the prime sets in memory.

3 Online version of the OAC-triclustering algorithm

At first, let us describe the online problem of finding the set of prime OAC-triclusters. Let $\mathbb{K} = (G, M, B, I)$ be a triadic context. The user has no a priori knowledge of the elements and even cardinalities of G, M, B, and I. At each iteration we receive some set of triples from I: $J \subseteq I$. After that we must process J and get the current version of the set of all triclusters. It is important in this setting to consider every pair of triclusters different if they have different generating triples, event if their extents, intents, and modi are equal, because any other triple can change only one of them, thus making them different. The picture 2 shows the example of such situation (dark gray cells are the generating triples, light gray — prime sets).

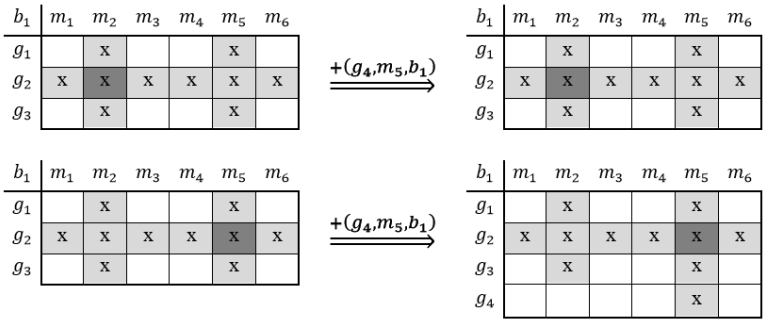


Fig. 2. Example of modification of triclusters by adding a triple

Also the algorithm requires that the dictionaries containing the prime sets are implemented as hash-tables. Because of this data structure the algorithm can efficiently access prime sets for their processing.

The algorithm itself is also quite simple (Alg. 2). It takes some set of triples (J) and current versions of the tricluster set ($\mathcal{T}$) and the dictionaries containing prime sets ($PrimesOA$, $PrimesOC$, $PrimesAC$) as input and outputs the modified versions of the tricluster set and dictionaries. The algorithm processes each triple (g, m, b) of J sequentially (line 1). On each iteration the algorithm modifies the corresponding prime sets:

- adds b to $(g, m)'$ (line 2)
- adds m to $(g, b)'$ (line 3)
- adds g to $(m, b)'$ (line 4)

Finally, it adds a new tricluster to the tricluster set. It is important to note that this tricluster contains pointers to the corresponding prime sets (in the corresponding dictionaries) instead of the copies of the prime sets (line 5).

In effect this algorithm is the same as the basic one but with some optimizations. First of all, instead of computing prime sets at the beginning, we modify them on spot, as adding an additional triple to the relation modifies only three prime sets by one element. Secondly, we remove the main loop by using pointers for the triclusters' extents, intents, and modi, as we can generate triclusters at the same step as we modify the prime sets. And the third important optimization is the use of only one pass through the triples of the ternary relation I, instead of enumeration of different pairwise combinations of objects, attributes, and conditions.

Algorithm 2 Add function for the online algorithm for prime OAC-triclustering.

Input: J — set of triples;
$\mathcal{T} = \{T = (*X, *Y, *Z)\}$ — current set of triclusters;
$PrimesOA$, $PrimesOC$, $PrimesAC$;
Output: $\mathcal{T} = \{T = (*X, *Y, *Z)\}$;
$PrimesOA$, $PrimesOC$, $PrimesAC$;
1: **for all** $(g, m, b) \in J$ **do**
2: $PrimesOA[g, m] := PrimesOA[g, m] \cup b$
3: $PrimesOC[g, b] := PrimesOC[g, b] \cup m$
4: $PrimesAC[m, b] := PrimesAC[m, b] \cup g$
5: $\mathcal{T} := \mathcal{T} \cup (\&PrimesAC[m, b], \&PrimesOC[g, b], \&PrimesOA[g, m])$
6: **end for**

Let us estimate the complexities of this algorithm. Each step requires the constant time: we need to modify three sets and add one tricluster to the set of triclusters. The total number of steps is equal to $|I|$. Thus the time complexity is linear $O(|I|)$. Beside that the algorithms is one-pass.

The memory complexity is the same: for each of $|I|$ steps the size of each dictionary containing prime sets is increased either by one element (if the required prime set is already present), or by one key-value pair (if not). Still, each of these dictionary requires $O(|I|)$ memory. Thus, the memory complexity is also linear $O(|I|)$.

Another important step used as an addition to this algorithm is post-processing. In addition to the user-specific post-processing there are some common useful steps. First of all, in the fixed moment of time we may want to remove additional triclusters with the same extent, intent, and modus from the output. Also some simple conditions like minimal support condition can be processed during

this step without increasing the original complexity. It should be done only during the post-processing step, as the addition of a triple in the main algorithm can drastically change the set of triclusters, and, respectively, the values used to check the conditions. Finally, if we need to check more difficult conditions like minimal density condition the time complexity of the post-processing will be higher than the time complexity of the original algorithm, but it can be also efficiently implemented.

To remove the same triclusters we need to use an efficient hashing procedure that can be improved by implementing it in the main algorithm. For this for all prime sets we need to keep their hash-values with them in the memory. And finally, when using hash-functions other than LSH function (Locality-Sensitive Hashing) [14] we can calculate hash-values of prime sets as some function of their elements (for example, exclusive disjunction or sum). Then when we modify prime sets we just need to get the result of this function and the new element. In this case, the hash-value of the tricluster can be calculated as the same function of the hash-values of its extent, intent, and modus.

Then it would be enough to implement the tricluster set as a hash-set in order to efficiently remove the additional entries of the same tricluster.

Pseudo-code for the basic post-processing (Alg. 3).

Algorithm 3 Post-processing for the online algorithm for prime OAC-triclustering.

Input: $\mathcal{T} = \{T = (*X, *Y, *Z)\}$ — full set of triclusters;
Output: $\overline{\mathcal{T}} = \{T = (*X, *Y, *Z)\}$ — processed hash-set of triclusters;

```
for all T ∈ 𝒯 do
    Calculate hash(T)
    if hash(T) ∉ 𝒯̄ then
        𝒯̄ := 𝒯̄ ∪ T
    end if
end for
```

If the names of the objects, attributes, and conditions are small enough (so that we can consider the time complexity of computing their hash values as $O(1)$), the time complexity of the post-processing is $O(|I|)$ if we do not need to calculate densities, and $O(|I||G||M||B|)$ otherwise. Also, the basic version of the post-processing does not require any additional memory, so its memory complexity is $O(1)$.

Finally, the algorithm can be easily paralleled by splitting the subset of triples J into several subsets, processing each of them independently, and merging the resulting sets afterwards.

4 Experiments

Two series of experiments were conducted in order to verify the time complexities and efficiency of the online algorithm: first one was conducted on the first set of synthetic contexts and on real world datasets, the second one — on the second set of synthetic contexts with large number of triples in each. In each experiment for the first set both versions of the OAC-triclustering algorithm were used to extract triclusters from a given context. Only the online version of the algorithm was applied to the second set of contexts as the computation time of the basic version of the algorithm was too high. To evaluate the time more precisely, for each context there were 5 runs of the algorithms with the average result recorded.

4.1 Datasets

Synthetic datasets. As it was mentioned, two sets of synthetic contexts were generated.

First five contexts have the same size, but different average densities. The sets of objects, attributes, and conditions of these contexts consist of 50 elements each (thus, the maximal number of triples for them is equal to 125,000). To form the relation I a pseudo-random number generator was used. It added each triple to the context with the given probability that was different for each context. These probabilities were: 0.02, 0.04, 0.06, 0.08, and 0.1.

The second set of uniform synthetic contexts consists of 10 contexts with the same probability for each triple to be included (0.001), but with different sizes of the sets of objects, attributes, and conditions. These sizes were 100, 200, 300, ..., 1000.

IMDB. This dataset consists of Top-250 list of the Internet Movie Database (250 best movies based on user reviews). For the analysis the following triadic context was extracted: the set of objects consists of movie names, the set of attributes — of tags, the set of conditions — of genres, and a triple of the ternary relation means that the given movie has the given genre and is assigned the given tag.

Bibsonomy. Finally, a sample of the data of bibsonomy.org was used. This website allows users to share bookmarks and lists of literature and tag them. For the research the following triadic context was extracted: the set of objects consists of users, the set of attributes (tags), the set of conditions (bookmarks), and a triple of the ternary relation means that the given user has assigned the given tag to the given bookmark.

The table 1 contains the summary of the contexts.

4.2 Results

The experiments were conducted on the computer running under Windows 8, using Intel Core i7-3517U 2.40 GHz processor, having 8 GB RAM. The algorithms

Table 1. Contexts for the experiments

Context	$\|G\|$	$\|M\|$	$\|B\|$	# triples	Density
Synthetic[1], 0.02	50	50	50	2530	0.02024
Synthetic[1], 0.04	50	50	50	5001	0.04001
Synthetic[1], 0.06	50	50	50	7454	0.05963
Synthetic[1], 0.08	50	50	50	10046	0.08037
Synthetic[1], 0.1	50	50	50	12462	0.09970
Synthetic[2], 100	100	100	100	996	0.001
Synthetic[2], 200	200	200	200	7995	0.001
Synthetic[2], 300	300	300	300	27161	0.001
Synthetic[2], 400	400	400	400	63921	0.001
Synthetic[2], 500	500	500	500	125104	0.001
Synthetic[2], 600	600	600	600	216021	0.001
Synthetic[2], 700	700	700	700	343157	0.001
Synthetic[2], 800	800	800	800	512097	0.001
Synthetic[2], 900	900	900	900	729395	0.001
Synthetic[2], 1000	1000	1000	1000	1000589	0.001
IMDB	250	795	22	3818	0.00087
BibSonomy	51	924	2844	3000	0.000022

were implemented in C# under .NET Framework 4.5. Jenkins' hash-function [11] was used to generate hash-values.

Figure 3 shows the time performance of both versions of the algorithms for different values of minimal density threshold. Figure 4 shows the computation time for the online version of the algorithm on the second set of synthetic contexts. "Basic" graph refers to the average time required by the basic algorithm, "Online, algorithm" — to average time required by the main algorithm part of the online algorithm (addition of new triples), "Online, total" — to the average time required by both the main algorithm and post-processing. Table 4.2 contains summary of the results for the case of zero minimal threshold.

As it can be clearly seen from all the graphs, online version of the algorithm significantly outperforms the basic version. However, post-processing in case of non-zero minimal density threshold can minimize the difference, especially in cases with small sets of objects, attributes, and conditions and large ternary relation.

In the case of several contexts of the fixed size, but increasing density, total computation time converges to the same value for the both algorithms, with the time for the online one being slightly smaller. For the non-zero minimal density threshold this convergence takes place for almost any average density value. In this case there is a rather large number of triclusters of big size, with many intersections, thus it takes much time to calculate all the triclusters' densities. This situation is close to the worst case, where time complexity is $O(|G||M||B|)$ for the main algorithm (because $|I|$ converges to $|G||M||B|$) and $O(|I||G||M||B|)$ for the post-processing. Also, in the case where the context's

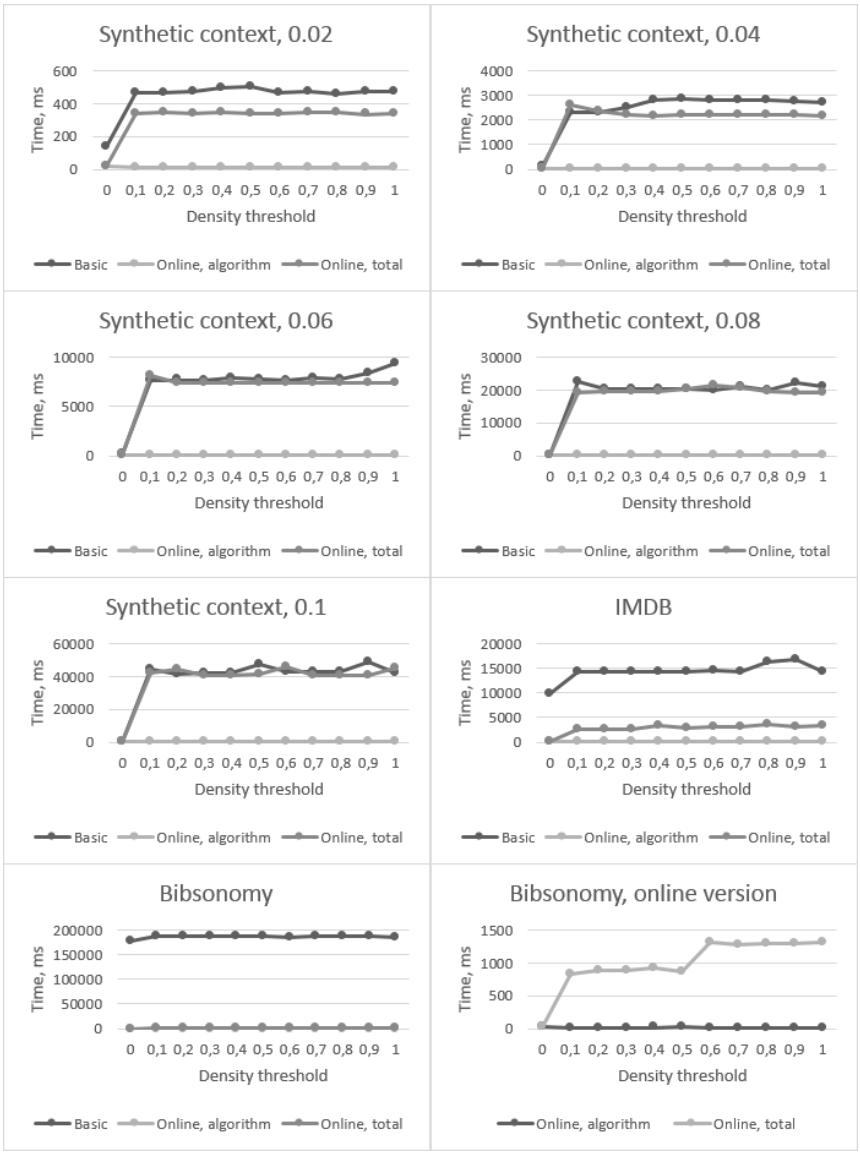


Fig. 3. Results of the experiments for both versions of OAC-triclustering algorithm

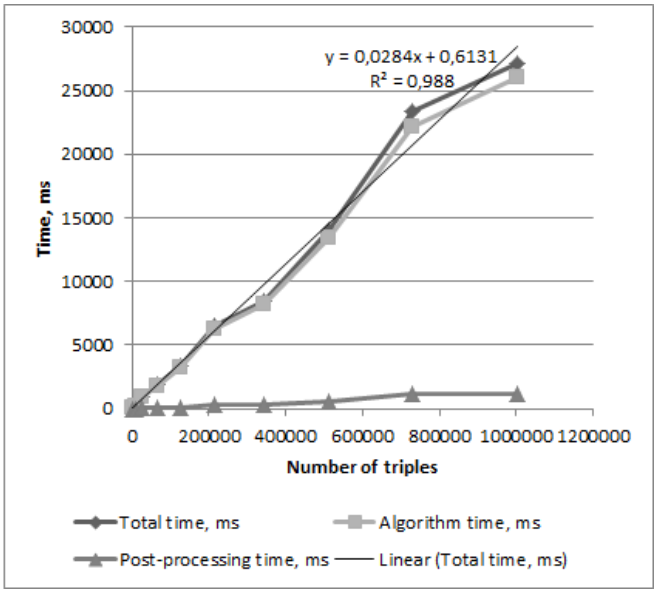


Fig. 4. Computation time for the online algorithm for various numbers of triples

density getting closer to 1, total time for both algorithms should be almost the same even in the case of zero minimal density threshold, as in the worst case for dense contexts $|I|$ is equal to $|G||M||B|$ (though it is an extremely rare case for real datasets).

The results for the second set of synthetic contexts confirm that the algorithm is indeed linear with respect to the number of triples. It also shows that the significant number of triples does not affect the performance as long as the context fits in the memory.

As for the other datasets with large sets of objects, attributes, and conditions and small ternary relation, the online algorithm significantly outperforms the basic one. The basic version spends much time on enumeration the large number of combinations of the elements of different sets of the context, while the online one just passes through the existing triples. Time to compute densities is quite small for these datasets since due to their sparseness they contain small number of rather small triclusters.

Finally, as it can be seen, for non-dense contexts the average density of triclusters is rather high even in the case of zero minimal density threshold. Because of that, it can be advised in most of the cases to use the online version of the algorithm without any hard conditions, like minimal density condition, as the results will still be good, but the performance will be significantly improved.

Table 2. Tricluster sets summary

Context	Number of triclusters	Average density
Synthetic[1], 0.02	2456	0.700
Synthetic[1], 0.04	4999	0.426
Synthetic[1], 0.06	7453	0.286
Synthetic[1], 0.08	10046	0.218
Synthetic[1], 0.1	12462	0.193
Synthetic[2], 100	897	0.993
Synthetic[2], 200	6972	0.972
Synthetic[2], 300	23645	0.941
Synthetic[2], 400	56584	0.909
Synthetic[2], 500	113041	0.871
Synthetic[2], 600	199210	0.834
Synthetic[2], 700	322447	0.796
Synthetic[2], 800	487982	0.759
Synthetic[2], 900	703374	0.722
Synthetic[2], 1000	973797	0.686
IMDB	1276	0.539
BibSonomy	1290	0.946

5 Conclusion

In this paper we have presented an online version of OAC-triclustering algorithm. We have shown that the algorithm is efficient from both theoretical and practical points of view. Its linear time complexity and performance in one pass (with an additional pass for the required post-processing) allows us to use it for big data problems. Moreover, the online algorithm as well as the basic one can be easily parallelized to attain even larger efficiency.

Acknowledgements. The study was implemented in the framework of the Basic Research Program at the National Research University Higher School of Economics in 2013-2014, in the Laboratory of Intelligent Systems and Structural Analysis (Russian Federation), and in the LIMOS (Laboratoire d'Informatique, de Modelisation et d'Optimisation des Systemes) (France). The first three authors were partially supported by Russian Foundation for Basic Research, grant no. 13-07-00504.

References

1. Cerf, L., Besson, J., Nguyen, K.N., Boulicaut, J.F.: Closed and noise-tolerant patterns in n-ary relations. Data Min. Knowl. Discov. 26(3), 574–619 (2013)
2. Davey, B.A., Priestley, H.A.: Introduction to Lattices and Order. Cambridge University Press, 2 edn. (2002)
3. Eren, K., Deveci, M., Kucuktunc, O., Catalyurek, Umit V.: A comparative analysis of biclustering algorithms for gene expression data. Briefings in Bioinform. (2012)

4. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edn. (1999)
5. Gnatyshak, D.V., Ignatov, D.I., Kuznetsov, S.O.: From triadic FCA to triclustering: Experimental comparison of some triclustering algorithms. In: Ojeda-Aciego, M., Outrata, J. (eds.) CLA. CEUR Workshop Proceedings, vol. 1062, pp. 249–260. CEUR-WS.org (2013)
6. Gnatyshak, D.V., Ignatov, D.I., Semenov, A.V., Poelmans, J.: Gaining insight in social networks with biclustering and triclustering. In: BIR. Lecture Notes in Business Information Processing, vol. 128, pp. 162–171. Springer (2012)
7. Ignatov, D.I., Kuznetsov, S.O., Poelmans, J.: Concept-based biclustering for internet advertisement. In: ICDM Workshops. pp. 123–130. IEEE Computer Society (2012)
8. Ignatov, D.I., Kuznetsov, S.O., Poelmans, J., Zhukov, L.E.: Can triconcepts become triclusters? International Journal of General Systems 42(6), 572–593 (2013)
9. Ignatov, D.I., Nenova, E., Konstantinova, N., Konstantinov, A.V.: Boolean Matrix Factorisation for Collaborative Filtering: An FCA-Based Approach. In: Agre, G., et al. (eds.) AIMSA 2014, Varna, Bulgaria, September 11-13, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8722, pp. 47–58. Springer (2014)
10. Jelassi, M.N., Yahia, S.B., Nguifo, E.M.: A personalized recommender system based on users' information in folksonomies. In: Carr, L., et al. (eds.) WWW (Companion Volume). pp. 1215–1224. ACM (2013)
11. Jenkins, B.: A hash function for hash table lookup (2006), `http://www.burtleburtle.net/bob/hash/doobs.html`
12. Kaytoue, M., Kuznetsov, S.O., Macko, J., Napoli, A.: Biclustering meets triadic concept analysis. Ann. Math. Artif. Intell. 70(1-2), 55–79 (2014)
13. Kaytoue, M., Kuznetsov, S.O., Napoli, A., Duplessis, S.: Mining gene expression data with pattern structures in formal concept analysis. Inf. Sci. 181(10), 1989–2001 (2011), `http://dx.doi.org/10.1016/j.ins.2010.07.007`
14. Leskovec, J., Rajaraman, A., Ullman, J.: Mining of Massive Datasets, chap. Finding Similar Items, pp. 71–128. Cambridge University Press, England, Cambridge (2010)
15. Li, A., Tuck, D.: An effective tri-clustering algorithm combining expression data with gene regulation information. Gene regulation and systems biology 3, 49–64 (2009), `http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2758278/`
16. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: A survey. IEEE/ACM Trans. Comput. Biology Bioinform. 1(1), 24–45 (2004)
17. Mirkin, B.: Mathematical Classification and Clustering. Kluwer, Dordrecht (1996)
18. Mirkin, B.G., Kramarenko, A.V.: Approximate bicluster and tricluster boxes in the analysis of binary data. In: Kuznetsov, S.O., et al. (eds.) RSFDGrC 2011. Lecture Notes in Computer Science, vol. 6743, pp. 248–256. Springer (2011)
19. Nanopoulos, A., Rafailidis, D., Symeonidis, P., Manolopoulos, Y.: Musicbox: Personalized music recommendation based on cubic analysis of social tags. IEEE Transactions on Audio, Speech & Language Processing 18(2), 407–412 (2010)
20. Wille, R.: Restructuring lattice theory: An approach based on hierarchies of concepts. In: Rival, I. (ed.) Ordered Sets, NATO Advanced Study Institutes Series, vol. 83, pp. 445–470. Springer Netherlands (1982)
21. Zhao, L., Zaki, M.J.: Tricluster: An effective algorithm for mining coherent clusters in 3d microarray data. In: Özcan, F. (ed.) SIGMOD Conference. pp. 694–705. ACM (2005)

Three Related FCA Methods for Mining Biclusters of Similar Values on Columns

Mehdi Kaytoue[1], Victor Codocedo[2], Jaume Baixieres[3], and Amedeo Napoli[2]

[1] Université de Lyon. CNRS, INSA-Lyon, LIRIS. UMR5205, F-69621, France.
[2] LORIA (CNRS - Inria Nancy Grand Est - Université de Lorraine), B.P. 239, F-54506, Vandœuvre-lès-Nancy.
[3] Universitat Politècnica de Catalunya. 08032, Barcelona. Catalonia.

Corresponding author: mehdi.kaytoue@insa-lyon.fr

Abstract. Biclustering numerical data tables consists in detecting particular and strong associations between both subsets of objects and attributes. Such biclusters are interesting since they model the data as local patterns. Whereas there exists several definitions of biclusters, depending on the constraints they should respect, we focus in this paper on *biclusters of similar values on columns*. There are several *ad hoc* methods for mining such biclusters in the literature. We focus here on two aspects: *genericity* and *efficiency*. We show that Formal Concept Analysis provides a mathematical framework to characterize them in several ways, but also to compute them with existing and efficient algorithms. The proposed methods, which rely on pattern structures and triadic concept analysis, are experimented and compared on two different datasets.

Keywords: biclustering, triadic concept analysis, pattern structure

1 Introduction

Biclustering has attracted a lot of attention for many years now, as it was used in an extensive way for mining biological data [7]. Given a data-table with objects as rows and attributes as columns, the goal is to find "sub-tables", or pairs of both subsets of objects and attributes, such that the values in the subtables respect well-defined constraints or maximize a given measure [17].

There exist several types of biclusters depending on the relation the values should respect. For example, *constant biclusters* are subtables with equal values [12, 6, 17]. *Biclusters with similar values on columns* (BSVC) are subtables where all values are pairwise similar for each column [4, 17]. The latter can also be generalized to *biclusters of similar values* (BSV): any two values in the subtable are similar [2, 3, 12, 21]. Dozens of algorithms, mostly *ad hoc*, have been proposed for computing the different types of biclusters. In this paper, we are interested in possible extensions of the Formal Concept Analysis (FCA) formalism for achieving the problem of biclustering. This comes with two goals: (i) formalizing and understanding biclusters formation and structure, and (ii) reusing existing algorithms for genericity purposes.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

Actually, the present paper is in continuation with the work of the authors on the use of pattern structures –an extension of FCA for mining complex data [8, 12]– for discovering functional dependencies in a crisp and a fuzzy settings [1], and as well on the adaptation of pattern structures to a specific biclustering task: the discovery of biclusters of type BSV [6, 11]. Moreover, the biclustering task is usually considered as a "'two-dimensional" (2D) process where biclusters are rectangles in a table verifying some prior constraints. It was one main idea of [11] to transpose the problem in a "three-dimensional" setting by using and adapting triadic concept analysis [16] to the biclustering task.

Here we follow the same line and we propose a new approach for discovering biclusters in a numerical dataset where biclusters have "similar values" w.r.t. their columns (type BSVC). This works is a new attempt to extend the capabilities of FCA and of pattern structures, in dealing with the important problem of biclustering. Actually, biclustering can be also considered in a (pure) numerical setting, where it is sometimes called coclustering [18] and where kernel or spectral methods are often used for achieving the task. Here we keep the discrete setting and more precisely an FCA-based setting.

The rest of this paper is organized as follows. In Section 2 we formally introduce the biclustering problem. Then, we recall in Section 3 the FCA basics that are necessary for developing our three methods in Section 4. We experiment with these methods and compare them by processing two real-world datasets in Section 5 before concluding.

2 Problem Definition

We introduce the problem of mining biclusters of similar values on columns, or simply biclusters when no confusion can be made. A numerical dataset is defined as a many-valued context in which biclusters are denoted as pairs of object and attribute subsets for which a particular similarity constraint holds.

Definition 1 (Many-valued context and numerical dataset). *A many-valued context consists in a quadruple (G, M, W, I) where G is a set of objects, M a set of attributes, W a set of attribute values, and $I \subseteq G \times M \times W$ a ternary relation. An element $(g, m, w) \in I$, also written $m(g) = w$ or $g(m) = w$, can be interpreted as: w is the value taken by the attribute m for the object g. The relation I is such that $g(m) = w$ and $g(m) = v$ implies $w = v$.*

In the present work, W is a set of numbers and $\mathcal{K}_{num} = (G, M, W, I)$ denotes a numerical dataset, i.e. a many-valued context where W is a set of numbers.

Example. A tabular representation of a numerical dataset is given in Table 1: objects $G = \{g_1, g_2, g_3, g_4, g_5\}$ are represented by rows while attributes $M = \{m_1, m_2, m_3, m_4\}$ are represented by columns. $W = \{0, 1, 2, 6, 7, 8, 9\}$ and we have for example $g_2(m_4) = 9$.

	m_1	m_2	m_3	m_4
g_1	1	2	2	8
g_2	2	1	2	9
g_3	2	1	1	2
g_4	1	0	7	6
g_5	6	6	6	7

Fig. 1. A numerical dataset

Definition 2 (Biclusters with similar values on columns). *Given a numerical dataset (G, M, W, I), a pair (A, B) (where $A \subseteq G, B \subseteq M$) is called a bicluster of similar values on columns when the following statement holds:*

$$\forall g, h \in A, \forall m \in B, m(g) \simeq_\theta m(h)$$

where $\simeq_\theta$ is a similarity relation: $\forall w_1, w_2 \in W, \theta \in [0, max(W) - min(W)]$, $w_1 \simeq_\theta w_2 \iff |w_1 - w_2| \leq \theta$. A bicluster (A, B) is maximal if $\nexists g \in G \backslash A$ such that $(A \cup \{g\}, B)$ is a bicluster, and $\nexists m \in M \backslash B$ such that $(A, B \cup \{m\})$ is a bicluster.

Example. In Table 1, with $\theta = 1$, we have that $(A, B) = (\{g_1, g_2\}, \{m_1, m_2, m_3\})$ is a bicluster. Indeed, consider each attribute of B separately: the values taken by the objects A are pairwise similar. However, (A, B) is not maximal, since we have that both $(A \cup \{g_3\}, B)$ and $(A, B \cup \{m_4\})$ are also biclusters. Then, $(\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\})$ and $(\{g_1, g_2\}, \{m_1, m_2, m_3, m_4\})$ are both maximal.

Problem (Biclustering). Given a numerical dataset (G, M, W, I) and a similarity parameter θ, the goal of biclustering is to extract the set of all maximal biclusters (A, B) respecting the similarity constraint.

Remark. It should be noticed that in the formal definition, the similarity parameter is the same for all attributes. It is possible however to use a different parameter for each attribute without changing neither the problem definition or its resolution. For real-world datasets, one can choose different similarity parameters θ_m ($\forall m \in M$), but also can normalize/scale the attribute domains and use a single similarity parameter θ.

3 Basics on Formal Concept Analysis

In this paper, we show how our biclustering problem can be formalized and answered in FCA in different ways: (i) using standard FCA [9], (ii) using pattern structures [8], and (iii) using triadic concept analysis [16]. We recall below the basics of each approach.

Dyadic Concept Analysis. Let G be a set of objects, M a set of attributes and $I \subseteq G \times M$ be a binary relation. The fact $(g, m) \in I$ is interpreted as "g has attribute m". The two following derivation operators $(\cdot)'$ are defined:

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

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.

Triadic Concept Analysis. A triadic context is given by (G, M, B, Y) where G, M, and B are respectively called sets of objects, attributes and conditions, and $Y \subseteq G \times M \times B$. The fact $(g, m, b) \in Y$ is interpreted as the statement "Object g has the attribute m under condition b". A (triadic) concept of (G, M, B, Y) is a triple (A_1, A_2, A_3) with $A_1 \subseteq G$, $A_2 \subseteq M$ and $A_3 \subseteq B$ satisfying the two following statements: (i) $A_1 \times A_2 \times A_3 \subseteq Y$, $X_1 \times X_2 \times X_3 \subseteq Y$ and (ii) $A_1 \subseteq X_1$, $A_2 \subseteq X_2$ and $A_3 \subseteq X_3$ implies $A_1 = X_1$, $A_2 = X_2$ and $A_3 = X_3$. If (G, M, B, Y) is represented by a three dimensional table, (i) means that a concept stands for a 3-dimensional rectangle full of crosses while (ii) characterizes component-wise maximality of concepts. For a triadic concept (A_1, A_2, A_3), A_1 is called the extent, A_2 the intent and A_3 the modus. To derive triadic concepts, two pairs of derivation operators are defined. The reader can refer to [16] for their definitions which are not necessary for the understanding of the present work.

Pattern Structures. Let G be 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 $(G, (D, \sqcap), \delta)$ is called a *pattern structure*. Elements of D are called *patterns* and are ordered by a subsumption relation $\sqsubseteq$ such that given $c, d \in D$ one has $c \sqsubseteq d \Longleftrightarrow c \sqcap d = c$. Within the pattern structure $(G, (D, \sqcap), \delta)$ we can define the following derivation operators $(\cdot)^{\square}$, given $A \subseteq G$ and a description $d \in (D, \sqcap)$:

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

These operators form a Galois connection between $(\wp(G), \subseteq)$ and $(D, \sqsubseteq)$. *(Pattern) concepts* of $(G, (D, \sqcap), \delta)$ are pairs of the form (A, d), $A \subseteq G$, $d \in (D, \sqcap)$, such that $A^{\square} = d$ and $A = d^{\square}$. For a pattern concept (A, d), d is called a *pattern intent* and is the common description of all objects in A, called *pattern extent*. When partially ordered by $(A_1, d_1) \leq (A_2, d_2) \Leftrightarrow A_1 \subseteq A_2$ $(\Leftrightarrow d_2 \sqsubseteq d_1)$, the set of all concepts forms a complete lattice called a *(pattern) concept lattice*.

Computing Concepts and Concept Lattices. Processing a formal context in order to generate its set of concepts can be achieved by various algorithms (see [15] for a survey and a comparison, see also itemset mining [19]). For processing pattern structures, such algorithms generally need minor adaptations. Basically, one needs to override the code for (i) computing the intersection of any two arbitrary descriptions, and (ii) test the ordering between two descriptions. Processing a triadic context is however not so direct and can be done with nested FCA algorithms [10] or dedicated data-mining algorithm [5].

Similarity relations in FCA. The notion of similarity can be formalized by a tolerance relation: a symmetric, reflexive but not necessarily transitive relation. The similarity relation $\simeq_\theta$ used for defining biclusters of similar values is a tolerance. Given W a set of numbers, any maximal subset of pairwise similar values is called a block of tolerance.

Definition 3. *A binary relation $T \subseteq W \times W$ is called a* tolerance *relation if:*
(i) $\forall x \in W$ xTx (reflexivity)
(ii) $\forall x, y \in W$ $xTy \rightarrow yTx$ (symmetry)

Definition 4. *Given a set W, a subset $K \subseteq W$, and a tolerance relation T on W, K is a* block of tolerance *if:*
(i) $\forall x, y \in K\ xTy$ *(pairwise similarity)*
(ii) $\forall z \notin K, \exists u \in K\ \neg(zTu)$ *(maximality)*

It is shown that tolerance blocks can be obtained from the formal context of a tolerance relation [14]. In the context $(W, W, \simeq_\theta)$, one can characterize all blocks of tolerance K (and only them) as formal concepts (K, K).

4 Mining biclusters of similar values on columns in FCA

The basic notions of FCA of the previous section allow us now to answer our biclustering problem in various ways with: (i) an original method using interval pattern structure, (ii) a recently introduced method using partition pattern structures [6], and (iii) an original method relying on triadic concept analysis. We emphasize the genericity of FCA to answer a data mining problem.

4.1 Interval Pattern Structure Approach

For a dataset $\mathcal{K}_{num} = (G, M, W, I)$, an interval pattern structure $(G, (D, \sqcap), \delta)$ is defined as follows [13]: the objects from G are described by vectors of intervals, where each dimension gives a range of values for an attribute $m \in M$ (following a canonical ordering of the dimensions, i.e. dimension i corresponds to attribute $m_i \in M$). Then, for $m \in M$, the semi-lattice of intervals $(D_m, \sqcap_m)$ is given by:

$$\begin{aligned} D_m &= \{[w_1, w_2] \mid \exists g, h \in G \text{ s.t. } m(g) = w_1 \text{ and } m(h) = w_2\} \\ [a,b] \sqcap_m [c,d] &= [min(a,c), max(b,d)] \\ c \sqcap_m d = c &\iff c \sqsubseteq_m d \\ [a,b] \sqsubseteq_m [c,d] &\iff [c,d] \supseteq [a,b] \end{aligned}$$

The description space $(D, \sqcap)$ of the interval pattern structure is a product of meet-semi-lattices $(D, \sqcap) = \times_{m \in M}(D_m, \sqcap_m)$ which is a semi-lattice.

Examples. In Table 1, $(\{g_1, g_2, g_3\}, \langle [1,2], [1,2], [1,2], [2,9] \rangle)$ is a pattern concept:

$$\begin{aligned} \delta(g_1) &= \langle [1,1], [2,2], [2,2], [8,8] \rangle \\ \{g_1, g_2, g_3\}^\square = \delta(g_1) \sqcap \delta(g_2) \sqcap \delta(g_3) &= \langle [1,2], [1,2], [1,2], [2,9] \rangle \\ \langle [1,2], [1,2], [1,2], [8,9] \rangle \sqsubseteq \langle [1,2], [1,2], [1,2], [2,9] \rangle & \\ \{g_1, g_2, g_3\}^{\square\square} &= \{g_1, g_2, g_3\} \end{aligned}$$

We now give the intuitive idea on how the interval pattern concept lattice can be used to characterize the biclusters. Consider first the concept $(A_1, d_1) = (\{g_1, g_2\}, \langle [1,2], [1,2], [1,2], [8,9] \rangle)$. Consider also a function $attr : D \to M$ which returns for an interval pattern the set of attributes whose interval is not larger than the θ parameter, for $d = \langle [a_i, b_i] \rangle$, $i \in [1, |M|]$: $attr(d) = \{m_i \in M | a_i \simeq_\theta b_i\}$. $(A_1, attr(d_1)) = (\{g_1, g_2\}, \{m_1, m_2, m_3, m_4\})$ is a maximal bicluster. Consider the interval pattern concept $(A_2, d_2) = (\{g_1, g_2, g_3\}, \langle [1,2], [1,2], [1,2], [2,9] \rangle)$: $(A_2, attr(d_2)) = (\{g_1, g_2, g_3\}, \{m_1, m_2, m_3\})$ is a maximal bicluster (with $\theta = 1$). This means that biclusters can be characterized thanks to pattern concepts.

Proposition 1. *Consider a numerical dataset (G, M, W, I) as an interval pattern structure $(G, (D, \sqcap), \delta)$. For any maximal bicluster (A, B), there exists a pattern concept (A, d) such that $(A, B) = (A, attr(d))$.*

Proof. To ease reading, the proof is given in an appendix. □

4.2 Partition pattern structure approach

A partition pattern structure is a pattern structure instance where the description space is given by a semi-lattice of partitions over a set X [2]. Formally, we have $(G, (D, \sqcap), \delta)$ where: $D = Part(X)$ and $d_1 \sqcap d_2 = \bigcup p_i \cap p_j$ where $p_i, p_j \subseteq X$, $p_i \in d_1$, $p_j \in d_2$. The semi-lattice is actually a complete lattice of set partitions in which the bottom element is not considered. In [1], we showed that the definition of $\sqcap$, and equivalently $\sqsubseteq$, needs a slight modification when $D = 2^{2^K}$, i.e. a description $d \in D$ is a set of subsets of X, and they do cover X (possibly with overlapping). In that case, we have that $d_1 \sqcap d_2 = max(\bigcup p_i \cap p_j)$ where $p_i, p_j \subseteq X$, $p_i \in d_1$, $p_j \in d_2$ and $max(.)$ returns the maximal sets w.r.t. inclusion.

Now we show that such a pattern structure can be constructed from a numerical dataset, and that the corresponding concepts allow to generate all maximal biclusters. From a numerical dataset (G, M, W, I), we build the structure $(M, (D, \sqcap), \delta)$ where $D = 2^{2^G}$. The description of an object[4] $m \in M$ is given by: $\delta(m) = \{p_1, p_2, ...\}$ where $p_1, p_2, .. \subseteq G$ and:

$$
\begin{aligned}
m(g_1) \simeq_\theta m(g_2), \forall g_1, g_2 \in p_i \quad &\text{(similarity)} \\
\nexists g_3 \in G \backslash p_i \text{ with } m(g_3) \simeq_\theta m(g_k), \forall g_k \in p_i \quad &\text{(maximality)} \\
\bigcup_i p_i = G \quad &\text{(covering)}
\end{aligned}
$$

In other words, each original attribute $m \in M$ is described by a family of subsets of G, where each one corresponds to a block of tolerance w.r.t. the values of attribute m. Let $(A, d = \{p_i\})$ be a partition pattern concept, it is easy to see how the pairs $bic_i = (p_i, A)$ are biclusters with rows $g \in p_i$ and columns $m \in A$[5]. While any $bic_i = (p_i, A)$ is a bicluster, it is not necessarily a maximal bicluster. Nevertheless, maximal biclusters can be identified using the concept lattice.

Proposition 2. *Consider a pattern concept $(A, d = \{p_i\})$. The bicluster $bic_i = (p_i, A)$ is maximal if there is no pattern concept $(C, \{p_i, ...\})$ with $A \subseteq C$.*

Proof. The proof to this proposition is very intuitive. Recall from Section 2 that the bicluster (p_i, A) is maximal if two conditions are met, namely $\nexists g \in G \backslash p_i$ such that $(p_i \cup \{g\}, A)$ is a bicluster and $\nexists m \in M \backslash A$ such that $(p_i, A \cup \{m\})$ is

[4] Object in the pattern structure; attribute in the numerical dataset.

[5] In order to keep consistency with the previous notation, biclusters are written inversely as partition pattern concepts.

a bicluster, The first condition holds for bic_i given the maximality condition of the tolerance block p_i; The second follows from the proposition declaration. □

Example. The numerical dataset (G, M, W, I) given in Table 1 can be turned into a pattern structure as follows with $\theta = 1$:

$$\delta(m_1) = \{\{g_1, g_2, g_3, g_4\}\{g_5\}\} \quad \delta(m_2) = \{\{g_2, g_3, g_4\}\{g_1, g_2, g_3\}\{g_5\}\}$$
$$\delta(m_3) = \{\{g_1, g_2, g_3\}\{g_4, g_5\}\} \quad \delta(m_4) = \{\{g_4, g_5\}\{g_1, g_5\}\{g_1, g_2\}\{g_3\}\}$$

Indeed, each component of a description is a maximal set of objects having pairwise similar values for a given attribute. The pattern concept lattice is given in Figure 2. We remark that (i) any concept corresponds to a bicluster, (ii) some of them correspond to a maximal bicluster, and most importantly, (iii) any maximal bicluster can be found as a concept. For example, from the concept $(A_1, d_1) = (\{m_3, m_4\}, \{\{g_1, g_2\}, \{g_4, g_5\}, \{g_3\}\})$ we obtain the following biclusters: $bic_1 = (\{g_1, g_2\}, \{m_3, m_4\})$ and $bic_2 = (\{g_4, g_5\}, \{m_3, m_4\})$. Whereas bic_2 is a maximal bicluster bic_1 is not since we have that $(A_2, d_2) = (\{m_1, m_2, m_3, m_4\}, \{\{g_1, g_2\}, \{g_3\}, \{g_4\}, \{g_5\}\})$ with $(A_2, d_2) \leq (A_1, d_1)$. In turn, $bic_3 = (\{g_1, g_2\}, \{m_1, m_2, m_3, m_4\})$ is a maximal bicluster.

Remark. It is noticeable that an equivalent formal context can be built. By equivalent, we mean that the concept lattices produced by both structures are isomorphic. To obtain this formal context, we use a slight modification of the data transformation of [9] (pp. 92): $(M, \mathbb{B}_2(G), I)$ st. $(m, (g, h)) \in I \iff m(g) \simeq_\theta m(h)$. The concept lattice is equivalent to the pattern concept lattice [2], and thus it can be used in the same way to get maximal biclusters. In our running example, such context is given in Table 1, and its associated concept lattice is given in Figure 2 (right), a lattice isomorphic to the one raised from the pattern structure (left). The proof can be done in a similar manner as it is done in [2].

	(g_1, g_2)	(g_1, g_3)	(g_1, g_4)	(g_1, g_5)	(g_2, g_3)	(g_2, g_4)	(g_2, g_5)	(g_3, g_4)	(g_3, g_5)	(g_4, g_5)
m_1	×	×	×		×	×		×		
m_2	×	×			×	×		×		
m_3	×	×			×					×
m_4	×									×

Table 1. Formal context

4.3 Triadic Concept Analysis Approach

We present another original result: any maximal bicluster of similar values is characterized as a triadic concept. The triadic context is derived from the numerical dataset by encoding the tolerance relation between the values.

Proposition 3. *Given a numerical dataset (G, M, W, I), consider the derived triadic context given by (M, G, G, Y) s.t. $(m, g_1, g_2) \in Y \iff m(g_1) \simeq_\theta m(g_2)$.*

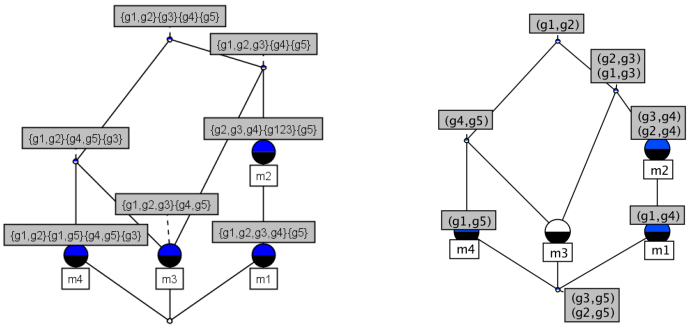


Fig. 2. Pattern concept lattice on the left side, concept lattice of the right side.

There is a one-to-one correspondence between the set of all maximal biclusters (A, B)*, the set of all triadic concepts* (B, A, A) *of the derived context.*

Proof. Consider a maximal bicluster (A, B). We have that $\forall g, h \in A : m(g) \simeq_\theta m(h) \iff m \in B$, if and only if (by the definition of Y) $(B, A, A) \subseteq Y$. We now take $(B', A', A') \subseteq Y$ such that $B \subseteq B'$ and $A \subseteq A'$. Since (A, B) is a maximal bicluster, we have that for any pair of objects $g, h \in A'$ and $m \in B'$ such that $g(m) \simeq_\theta h(m)$, implies that $g, h \in A$ and $m \in B$. Let (B, A, A) be a triadic concept. We have that for any pair of objects $g, h \in A$ and $m \in B$ we have that $g(m) \simeq_\theta h(m)$, this is, that $\forall g, h \in A : g(m) \simeq_\theta h(m) \iff m \in B$, which is the alternative definition of maximal bicluster. □

Example. Taking again $\theta = 1$, the triadic context derived from the numerical dataset from Table 1 is given in Table 2. An example of triadic concept is: $(\{m_3, m_2, m_1\}, \{g_1, g_3, g_2\}, \{g_1, g_2, g_3\})$ which is in turn the maximal bicluster $(\{g_1, g_3, g_2\}, \{m_3, m_2, m_1\})$.

5 Experiments

We experiment with the different FCA methods introduced in the previous section. We report preliminary results in two aspects: efficiency (running time) and compactness (number of concepts) to discuss the strengths and weaknesses of the different methods.

m_1	g_1	g_2	g_3	g_4	g_5
g_1	×	×	×	×	
g_2	×	×	×	×	
g_3	×	×	×	×	
g_4	×	×	×	×	
g_5					×

m_2	g_1	g_2	g_3	g_4	g_5
g_1	×	×	×		
g_2	×	×	×	×	
g_3	×	×	×	×	
g_4		×	×	×	
g_5					×

m_3	g_1	g_2	g_3	g_4	g_5
g_1	×	×	×		
g_2	×	×	×		
g_3	×	×	×		
g_4				×	×
g_5				×	×

m_4	g_1	g_2	g_3	g_4	g_5
g_1	×	×			×
g_2	×	×			
g_3			×		
g_4				×	×
g_5	×			×	×

Table 2. Triadic context derived from Table 1 thanks to $\simeq_1$.

Data and experimental settings. The first dataset, "Diagnosis"[6], contains 120 objects with 8 attributes. The first attribute provides temperature information of a given patient with a range $[35.5, 41.5]$ (numerical). For this attribute we used $\theta = 0.1$ and then $\theta = 0.3$. The other 7 attributes are binary ($\theta = 0$). The second dataset, "dataSample_1.txt", is provided with the BiCat software[7]. It contains 420 objects and 70 numerical attributes with range $[-5.9, 6.7]$. We used $\theta = 0.05$ for all attributes. We provide results in Table 3 for the three different FCA methods discussed in this article, namely interval pattern structure (IPS), tolerance blocks/partition pattern structures (TBPS) and triadic concept analysis (TCA). We also report on the use of standard FCA using the discretization technique discussed at the end of Section 4.2 (FCA). We also discuss the computing of clarified contexts, given that it can dramatically reduce the size of the context while keeping the same concept lattice (FCA-CL). A context is clarified when there exists neither two objects with the same description, or two attributes shared by the same set of objects.

For the methods based on FCA and pattern structures (IPS, TBPS), we used a C++ version of the AddIntent algorithm [20][8]. No restrictions were imposed over the size of the biclusters. The TCA method was implemented using DataPeeler [5]. All the experiments were performed using a Linux machine with Intel Xeon E7 running at 2.67GHz with 1TB of RAM.

Discussion. Results in Table 3 show that for the Diagnosis dataset, the clarified context using standard FCA (FCA-CL) is the best of the five methods w.r.t. execution time while for the BicAt sample 1, the best is TCA. Times are expressed as the sum of the time required to create the input representation of the dataset for the corresponding technique and its execution. In the case of FCA and FCA-CL, the pre-processing can be as high as the time required for applying the AddIntent algorithm. However, for large datasets such as the BicAt example, this times can be ignored. It is also worth noticing that the pre-processing depends on the chosen θ value, hence for each different θ configuration, a new pre-processing task has to be executed. This is not the case for interval and partition pattern structures the pre-processing of which is linear

[6] http://archive.ics.uci.edu/ml/datasets/Contraceptive+Method+Choice
[7] http://www.tik.ee.ethz.ch/sop/bicat/
[8] https://code.google.com/p/sephirot/

	Diagnosis				BicAt sample 1	
	$\theta = 0.3$		$\theta = 0.1$		$\theta = 0.05$	
Technique	Time [s] Preproc + Exec.	#Concepts	Exec. Time [s] Preproc + Exec.	#Concepts	Exec. Time [s] Preproc + Exec.	#Concepts
FCA	0.11 + 0.335	**98**	0.11 + 0.291	**88**	2.3 + 2,220	**476,950**
FCA-CL	**0.11 + 0.02**	**98**	**0.11 + 0.011**	**88**	2.3 + 2,220	**476,950**
TCA	0.04 + 33.3	3,322	0.04 + 31.34	2,127	**3.17 + 360**	741,421
IPS	0.011 + 0.303	928	0.001 + 0.178	301	0.02 + 2,340	722,442
TBPS	0.011 + 1.76	**98**	0.001 + 0.411	**88**	0.02 + 5,340	**476,950**

Table 3. Number of concepts and execution times (pre-processing + addIntent run)

w.r.t. the number of objects (it is actually, just a change of format). We can also appreciate a more compact representation of the biclusters by the use of partition pattern structures (TBPS) and its formal context versions (FCA and FCA-CL). While TBPS is the slowest of the five methods, it is also the cheapest one in terms of the use of machine resources, more specifically RAM. TCA is the more expensive method in terms of machine resources and data representation, however this yields results faster. Interval pattern structures are in the middle as a good trade-off of compactness and execution time.

For this initial experimentation we have not reported the number of maximal biclusters nor the bicluster extraction algorithms that can be implemented for each different technique, but only in the FCA techniques themselves. Regarding the number of maximal biclusters, this is the same for each technique since all of them are bicluster enumeration techniques, i.e. all possible biclusters are extracted. Hence, the difference among techniques is not given by the number of maximal biclusters extracted, but by the number of formal concepts found and their post-processing complexity to extract the maximal biclusters from them. In general, it is easy to observe from Propositions 1, 2 and 3 that the post-processing of TCA is linear w.r.t. the number of triadic concepts found, while for TPS is linear w.r.t. the number of interval pattern concepts times the number of columns of the numerical dataset squared and for TBPS is linear w.r.t. the number of *super-sub* concept relations in the tolerance block pattern concept lattice. Nevertheless, different strategies for bicluster extraction can be implemented for each technique rendering the comparison unfair. For example, in [6] an optimization is proposed regarding biclustering using partition pattern structures (which can be easily adapted to TBPS) which cuts in half its execution time by breaking the structure of the lattice. Similar strategies for IPS and TCA could also be implemented but are still a matter of research.

6 Conclusion

Biclustering is an important data analysis task that is used in several applications such as transcriptome analysis in biology and for the design of recommender systems. Biclustering methods produce a collection of local patterns that are easier to interpret than a global model. There are several types of biclusters and corresponding algorithms, *ad hoc* most of the time. In this paper, our main contribution shows how the *biclusters of similar values on columns* can be characterized or generated from formal concepts, pattern concepts and triadic concepts. Bringing back this problem of biclustering into formal concept analysis settings allows the usage of existing and efficient algorithms without any modifications. However, and this is among the perspectives of research, several optimizations can be made. For example, with the triadic method, one should not generate both concepts (A, B, C) and (A, C, B): they are redundant since only concepts with $B = C$ correspond to maximal biclusters.

References

1. J. Baixeries, M. Kaytoue, and A. Napoli. Computing similarity dependencies with pattern structures. In M. Ojeda-Aciego and J. Outrata, editors, *CLA*, volume 1062 of *CEUR Workshop Proceedings*, pages 33–44. CEUR-WS.org, 2013.
2. J. Baixeries, M. Kaytoue, and A. Napoli. Characterizing Functional Dependencies in Formal Concept Analysis with Pattern Structures. *Annals of Mathematics and Artificial Intelligence*, pages 1–21, Jan. 2014.
3. J. Besson, C. Robardet, L. D. Raedt, and J.-F. Boulicaut. Mining bi-sets in numerical data. In S. Dzeroski and J. Struyf, editors, *KDID*, volume 4747 of *Lecture Notes in Computer Science*, pages 11–23. Springer, 2007.
4. A. Califano, G. Stolovitzky, and Y. Tu. Analysis of gene expression microarrays for phenotype classification. In P. E. Bourne, M. Gribskov, R. B. Altman, N. Jensen, D. A. Hope, T. Lengauer, J. C. Mitchell, E. D. Scheeff, C. Smith, S. Strande, and H. Weissig, editors, *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, August 19-23, 2000, La Jolla / San Diego, CA, USA*, pages 75–85. AAAI, 2000.
5. L. Cerf, J. Besson, C. Robardet, and J.-F. Boulicaut. Closed patterns meet n-ary relations. *TKDD*, 3(1), 2009.
6. V. Codocedo and A. Napoli. Lattice-based biclustering using Partition Pattern Structures. In *21st European Conference on Artificial Intelligence (ECAI)*, 2014.
7. A. V. Freitas, W. Ayadi, M. Elloumi, J. Oliveira, J. Oliveira, and J.-K. Hao. *Biological Knowledge Discovery Handbook: Preprocessing, Mining, and Postprocessing of Biological Data*, chapter Survey on Biclustering of Gene Expression Data. John Wiley & Sons, Inc., 2013.
8. B. Ganter and S. O. Kuznetsov. Pattern structures and their projections. In *ICCS '01: Proceedings of the 9th International Conference on Conceptual Structures*, pages 129–142. Vol. 2120, Springer-Verlag, 2001.
9. B. Ganter and R. Wille. *Formal Concept Analysis*. Springer, 1999.
10. R. Jäschke, A. Hotho, C. Schmitz, B. Ganter, and G. Stumme. Trias - an algorithm for mining iceberg tri-lattices. In *ICDM*, pages 907–911, 2006.
11. M. Kaytoue, S. O. Kuznetsov, J. Macko, and A. Napoli. Biclustering meets triadic concept analysis. *Annals of Mathematics and Artificial Intelligence*, 70(1-2), 2014.
12. M. Kaytoue, S. O. Kuznetsov, and A. Napoli. Biclustering numerical data in formal concept analysis. In P. Valtchev and R. Jäschke, editors, *ICFCA*, volume 6628 of *LNCS*, pages 135–150. Springer, 2011.
13. M. Kaytoue, S. O. Kuznetsov, A. Napoli, and S. Duplessis. Mining gene expression data with pattern structures in formal concept analysis. *Information Science*, 181(10):1989–2001, 2011.
14. S. O. Kuznetsov. Galois connections in data analysis: Contributions from the soviet era and modern russian research. In B. Ganter, G. Stumme, and R. Wille, editors, *Formal Concept Analysis*, volume 3626 of *Lecture Notes in Computer Science*, pages 196–225. Springer, 2005.
15. S. O. Kuznetsov and S. A. Obiedkov. Comparing performance of algorithms for generating concept lattices. *J. Exp. Theor. Artif. Intell.*, 14(2-3):189–216, 2002.
16. F. Lehmann and R. Wille. A triadic approach to formal concept analysis. In *ICCS*, volume 954 of *LNCS*, pages 32–43. Springer, 1995.
17. S. Madeira and A. Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

18. N. Rogovschi, L. Labiod, and M. Nadif. A spectral algorithm for topographical co-clustering. In *IJCNN*, pages 1–6. IEEE, 2012.
19. T. Uno, M. Kiyomi, and H. Arimura. Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In R. J. B. Jr., B. Goethals, and M. J. Zaki, editors, *FIMI*, volume 126 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2004.
20. D. van der Merwe, S. Obiedkov, and D. Kourie. AddIntent: A New Incremental Algorithm for Constructing Concept Lattices. In P. Eklund, editor, *Concept Lattices*, volume 2961 of *LNCS*, pages 205–206. Springer, Berlin/Heidelberg, 2004.
21. R. Veroneze, A. Banerjee, and F. J. V. Zuben. Enumerating all maximal biclusters in real-valued datasets. *CoRR*, abs/1403.3562, 2014.

7 Appendix: Proof of proposition 1

We introduce notations, before to recall and prove Proposition 1 that relates maximal biclusters to interval pattern concepts of a pattern structure. The intuition lies in the relation between the set of attributes M of $(G, M, W, I))$ in an interval pattern structure $(G, (D, \sqcap), \delta)$. Let $d = \langle [a_1, b_1], [a_2, b_2], \ldots, [a_n, b_n] \rangle \in D$ be a pattern interval in an interval pattern structure $(G, (D, \sqcap), \delta)$, where $|M| = n$. For any $m_i \in M$, we define: $d(m_i) = [a_i, b_i]$. and $|d(m_i)| = |a_i - b_i|$.

Definition 5. *Let d be a pattern in an interval pattern structure $(G, (D, \sqcap), \delta)$. The function* $\text{attr} : D \mapsto M$ *is defined as:* $\text{attr}(d) = \{m \in M \mid |d(m)| \leq \theta\}$.

Definition 6. *Let $A \subseteq G$ be a set of objects and $m \in M$ an attribute. We define: $A(m) = \{g(m) \mid g \in B\}$. For instance, in Table 1, if $A = \{g_1, g_2, g_3\}$, then, $A(m_4) = \{2, 8, 9\}$.*

Proposition 4. *For $A \subseteq G$, we have that, for all $m_i \in M$:*

$$A^{\square} = \langle [min(A(m_1)), max(A(m_1))], \ldots, [min(A(m_n)), max(A(m_n))] \rangle$$

Proof. Since the operation $\sqcap$ is associative and commutative, we have that

$$A^{\square} = \prod_{g_i \in A} g_i = \langle [min(A(m_1)), max(A(m_1))], \ldots, [min(A(m_n)), max(A(m_n))] \rangle$$

□

Now we reformulate and prove the Proposition 1.

Proposition 5. *Consider a numerical dataset (G, M, W, I) as an interval pattern structure $(G, (D, \sqcap), \delta)$. For any maximal bicluster (A, B), we define: $d = A^{\square}$. Then: 1. $B = \text{attr}(d)$ and 2. (A, D) is a pattern concept in $(G, (D, \sqcap), \delta)$.*

Proof. 1. $B = \text{attr}(d)$. We prove that $m \in \text{attr}(b) \leftrightarrow m \in B$. Since $B = A^{\square}$, then, by the definition of maximal bicluster we have that $\forall m \in M : m \in B \leftrightarrow |A(m)| \leq \theta$, if and only if $|min(A(m)) - max(A(m))| \leq \theta$ if and only if (by the definition of d) $m \in \text{attr}(d)$. □

2. We need to prove that $A = d^{\square}$ and that $A^{\square} = d$. $A^{\square} = d$ holds by the definition of d. As for $A = d^{\square}$, we take $g \in d^{\square}$, which means that $\forall m \in M : g(m) \in d(m)$, also if $m \in B$, which implies that $g \in A$ by definition of maximal bicluster.

Defining Views with Formal Concept Analysis for Understanding SPARQL Query Results

Mehwish Alam[2,3] and Amedeo Napoli[1,2]

[1] CNRS, LORIA, UMR 7503, Vandoeuvre-lès-Nancy, F-54506, France
[2] Inria, Villers-lès-Nancy, F-54600, France
[3] Université de Lorraine, LORIA, UMR 7503, Vandoeuvre-lès-Nancy, F-54506, France
{mehwish.alam,amedeo.napoli@loria.fr}

Abstract. SPARQL queries over semantic web data usually produce list of tuples as answers that may be hard to understand and interpret. Accordingly, this paper focuses on Lattice-Based View Access (LBVA), a framework based on FCA. This framework provides a classification of the answers of SPARQL queries based on a concept lattice, that can be navigated for retrieving or mining specific patterns in query results. In this way, the concept lattice can be considered as a materialized view of the data resulting from a SPARQL query.

Keywords: Formal Concept Analysis, SPARQL Query Views, Lattice-Based Views, SPARQL, Classification.

1 Introduction

At present, Web has become a potentially large repository of knowledge, which is becoming main stream for querying and extracting useful information. In particular, Linked Open Data (LOD) [2] provides a method for publishing structured data in the form of RDF resources. These RDF resources are interlinked with each other to form a cloud. SPARQL queries are used in order to make these resources usable, i.e., queried. In some cases, queries in natural language against standard search engines can be simple to use but sometimes they are complex and may require integration of data sources. Then the standard search engines will not be able to easily answer these queries, e.g., *Currencies of all G8 countries*. Such a complex query can be formalized as a SPARQL query over data sources present in LOD cloud through SPARQL endpoints for retrieving answers. Moreover, users may sometimes execute queries which generate huge amount of results giving rise to the problem of information overload [5]. A typical example is given by the answers retrieved by search engines, which mix between several meanings of one keyword. In case of huge results, user will have to go through a lot of results to find the interesting ones, which can be overwhelming without any specific navigation tool. Same is the case with the answers obtained by SPARQL queries, which are huge in number and it may be harder to extract the most interesting patterns. This problem of information overload raises new

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

challenges for data access, information retrieval and knowledge discovery w.r.t web querying.

Accordingly, this paper proposes a new approach based on Formal Concept Analysis (FCA [7])s. It describes a lattice-based classification of the results obtained by SPARQL queries by introducing a new clause `VIEW BY` in SPARQL query. This framework, called Lattice-Based View Access (LBVA), allows the classification of SPARQL query results into a concept lattice, referred to as a *view*, for data analysis, navigation, knowledge discovery and information retrieval purposes. This new clause `VIEW BY` which enhances the functionality of already existing `GROUP BY` clause in SPARQL query by adding sophisticated classification and Knowledge Discovery aspects. Here after, we describe how a lattice-based view can be designed from a SPARQL query. Afterwards, a view is accessed for analysis and interpretation purposes which are totally supported by the concept lattice. In case of large data only a part of the lattice [10] can be considered for the analysis. In this way, this paper investigates also the capabilities of FCA to deal with semantic web data.

The intuition of classifying results obtained by SPARQL queries is inspired by web clustering engines [3] such as Carrot2[4]. The general idea behind web clustering engines is to group the results obtained by query posed by the user based on the different meanings of the terms related to a query. Such systems deal with unstructured textual data on web. By contrast, there are some studies conducted to deal with structured RDF data. In [5], the authors introduce a clause `Categorize By` to target the problem of managing large amounts of results obtained by conjunctive queries with the help of subsumption hierarchy present in the knowledge base. By contrast, the `VIEW BY` clause generates lattice-based views which provide a mathematically well-founded classification based on formal concepts and an associated concept lattice. Moreover, it also paves way for navigation or information retrieval by traversing the concept lattice and for data analysis by allowing the extraction of association rules from the lattice. Such data analysis operations allow discovery of new knowledge. Additionally, unlike `Categorize By`, `VIEW BY` can deal with data that has no schema (which is often the case with linked data). Moreover, `VIEW BY` has been evaluated over very large set of answers (roughly 100,000 results) obtained over real datasets. In case of larger number of answers, `Categorize By` does not provide any pruning mechanism while this paper describes how the views can be pruned using iceberg lattices.

The paper is structured as follows: Section 2 introduces a motivating example. Section 3 gives a brief introduction of the state of the art while Section 4 defines LBVA and gives the overall architecture of the framework. Section 5 discusses some experiments conducted using LBVA. Finally, Section 6 concludes the paper.

[4] http://project.carrot2.org/index.html

2 Motivation

In this section we introduce a motivating example focusing on why LOD should be queried and why the SPARQL query results need classification. This scenario will continue in the rest of the paper. Let us consider that a query `Q` searching for *museums where the exhibition of some famous artists is taking place along with the location of the museum*. Here, we do not discuss the interface aspects and we will assume that SPARQL queries are provided. A standard query engine is not adequate for answering such kind of questions and a direct query over LOD will give better results. One of the ways to obtain such an information is to query LOD through its SPARQL endpoint. This query will generate a huge amount of results, which will need further manual work to group the interesting links.

3 Background

3.1 Linked Open Data

Linked Open Data (LOD) [2] is the way of publishing structured data in the form of RDF graphs. Given a set of URIs $\mathbf{U}$, blank nodes $\mathbf{B}$ and literals $\mathbf{L}$, an RDF triple is represented as $t = (s, p, o) \in (\mathbf{U} \cup \mathbf{B}) \times \mathbf{U} \times (\mathbf{U} \cup \mathbf{B} \cup \mathbf{L})$, where s is a subject, p is a predicate and o is an object. A finite set of RDF triples is called as RDF Graph $\mathcal{G}$ such that $\mathcal{G} = (V, E)$, where V is a set of vertices and E is a set of labeled edges and $\mathcal{G} \in \mathbf{G}$, such that $\mathbf{G} = (\mathbf{U} \cup \mathbf{B}) \times \mathbf{U} \times (\mathbf{U} \cup \mathbf{B} \cup \mathbf{L})$. Each pair of vertices connected through a labeled edge keeps the information of a statement. Each statement is represented as $\langle$*subject*, *predicate*, *object*$\rangle$ referred to as an RDF Triple. V includes *subject* and *object* while E includes the *predicate*.

SPARQL[5] is the standard query language for RDF. In the current work we will focus on the queries containing `SELECT` clause. Let us assume that there exists a set of variables $\mathbf{V}$ disjoint from $\mathbf{U}$ in the above definition of RDF, then $(\mathbf{U} \cup \mathbf{V}) \times (\mathbf{U} \cup \mathbf{V}) \times (\mathbf{U} \cup \mathbf{V})$ is a graph pattern called a triple pattern. If a variable $?X \in V$ and $?X = c$ then $c \in U$. Given U, V and a triple pattern t a mapping $\mu(t)$ would be the triple obtained by replacing variables in t with U. $[\![.]\!]_G$ takes an expression of patterns and returns a set of mappings. Given a mapping $\mu : V \to U$ and a set of variables $W \subseteq V$, μ is represented as $\mu_{|W}$, which is described as a mapping such that $dom(\mu_{|W}) = dom(\mu) \cap W$ and $\mu_{|W}(?X) = \mu(?X)$ for every $?X \in dom(\mu) \cap W$. Finally, the SPARQL SELECT query is defined as follows:

Definition 1. *A SPARQL SELECT query is a tuple* (W, P)*, where* P *is a graph pattern and* W *is a set of variables such that* $W \subseteq var(P)$*. The answer of* (W, P) *over an RDF graph* G*, denoted by* $[\![(W, P)]\!]_G$ *, is the set of mappings:*

$$[\![(W, P)]\!]_G = \{\mu_{|W} | \mu \in [\![P]\!]_G\}$$

[5] http://www.w3.org/TR/rdf-sparql-query/

In Definition 1, $var(P)$ is the set of variables in pattern P and W is the set of variables in SELECT clause. Here, P includes the triple patterns containing variables. This triple pattern is then evaluated against the RDF Graph G given as $[\![P]\!]_G$. It returns a set of mappings with respect to the variables in $var(P)$. Finally a projection over μ is done w.r.t. the variables in W. The projected set of mappings obtained as represented as $\mu_{|W}$. Further details on the formalization and foundations of RDF databases are discussed in [1].

Example 1. Continuing the scenario in section 2, following is the SPARQL query:

```
1 SELECT ?museum ?country ?artist WHERE {
   2 ?museum rdf:type dbpedia-owl:Museum .
   3 ?museum dbpedia-owl:location ?city .
   4 ?city dbpedia-owl:country ?country .
   5 ?painting dbpedia-owl:museum ?museum .
   6 ?painting dbpprop:artist ?artist}
7 GROUP BY ?country ?artist
```

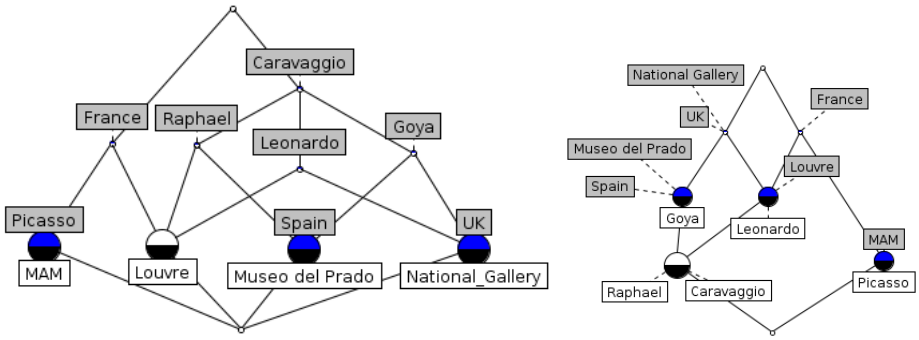
This query retrieves the list of museums along with the artists whose work is exhibited in a museum along with the location of a museum. `Lines 5 and 6` retrieve information about the artists whose work is displayed in some museum. More precisely, the page containing the information on a museum (`?museum`) is connected to the page of the artists (`?artist`) through a page on the work of artist (`?painting`) displayed in the museum. In order to integrate these three resources, two predicates were used `dbpedia-owl:museum` and `dbpprop:artist`. An excerpt of the answers obtained by `Group by` clause is shown below:

```
Pablo_Picasso     Musee_d'Art_Moderne France
Leonardo_Da_Vinci Musee_du_Louvre     France
Raphael           Museo_del_Prado     Spain
```

The problem encountered while browsing such an answer is that there are too many statements to navigate through. Even after using the `GROUP BY` clause the answers are not organized in any ordered structure. By contrast, the clause `VIEW BY` activates the LBVA framework, where the user will obtain a classification of the statements as a concept lattice where statements are partially ordered (see Figure 1a). To obtain the museums in `UK` displaying the work of `Goya`, all the museums displaying the work of `Goya` can be retrieved and then the specific concept containing `Goya and UK` is obtained by navigation. The answer obtained is `National Gallery` in the example.

3.2 Formal Concept Analysis (FCA)

As the basics of Formal Concept Analysis (FCA) [7] are well known, we only introduce some of the concepts which are necessary to understand this paper. FCA is a mathematical framework used for a number of purposes, among which classification and data analysis, information retrieval and knowledge discovery [4]. In some cases we obtain a huge number of concepts. In order to restrict the



(a) Classes of Museums w.r.t Artists and Countries, e.g., the concept on the top left corner with the attribute `France` contains all the French Museums, i.e., Musee du Louvre (Louvre) and Musee d'Art Moderne (MAM). (`VIEW BY ?museum`)

(b) Classes of Artists w.r.t Museums and Countries. (`VIEW BY ?artist`)

Fig. 1: Lattice-Based Views w.r.t Museum's and Artist's Perspective .

number of concepts, iceberg concept lattices can be used [10]. Iceberg concept lattices contain only the top most part of the lattice. Along with iceberg lattices a stability index [9] is also used for filtering the concepts. The stability index shows how much the concept intent depends on particular objects of the extent.

FCA also allows knowledge discovery using association rules. An implication over the attribute set M in a formal context is of the form $B_1 \rightarrow B_2$, where $B_1, B_2 \subseteq M$. The implication holds iff every object in the context with an attribute in B_1 also has all the attributes in B_2. For example, when $(A_1, B_1) \leq (A_2, B_2)$ in the lattice, we have that $B_1 \rightarrow B_2$. Duquenne-Guigues ($\mathcal{DG}$) basis for implications [8] is the minimal set of implications equivalent to the set of all valid implications for a formal context $\mathcal{K} = (G, M, I)$. Actually, the $\mathcal{DG}$-basis contains all information lying in the concept lattice.

4 Lattice-Based View Access

4.1 SPARQL Queries with Classification Capabilities

The idea of introducing a `VIEW BY` clause is to provide classification of the results and add a knowledge discovery aspect to the results w.r.t the variables appearing in `VIEW BY` clause. Let `Q` be a SPARQL query of the form `Q = SELECT ?X ?Y ?Z WHERE {pattern P} VIEW BY ?X` then the set of variables $V = \{?X, ?Y, ?Z\}$ [6]. According to the definition 1 the answer of the tuple (V, P) is represented as $[\![(\{?X, ?Y, ?Z\}, P)]\!] = \mu_i$ where $i \in \{1, \ldots, k\}$ and k is the number of mappings obtained for the query `Q`. For the sake of simplicity, $\mu_{|W}$ is given as μ. Here, $dom(\mu_i) = \{?X, ?Y, ?Z\}$ which means that $\mu(?X) = X_i$,

[6] As W represents set of attribute values in the definition of a many-valued formal context, we represent the variables in select clause as V to avoid confusion.

$\mu(?Y) = Y_i$ and $\mu(?Z) = Z_i$. Finally, a complete set of mappings can be given as $\{\{?X \rightarrow X_i, ?Y \rightarrow Y_i, ?Z \rightarrow Z_i\}\}$.

The variable appearing in the `VIEW BY` clause is referred to as object variable[7] and is denoted as Ov such that $Ov \in V$. In the current scenario $Ov = \{?X\}$. The remaining variables are referred to as attribute variables and are denoted as Av where $Av \in V$ such that $Ov \cup Av = V$ and $Ov \cap Av = \emptyset$, so, $Av = \{?Y, ?Z\}$.

Example 2. Following the example in section 2, an alternate query with the `VIEW BY` clause can be given as:

```
SELECT ?museum ?artist ?country WHERE {
   ?museum rdf:type dbpedia-owl:Museum .
   ?museum dbpedia-owl:location ?city .
   ?city dbpedia-owl:country ?country .
   ?painting dbpedia-owl:museum ?museum .
   ?painting dbpprop:artist ?artist}
VIEW BY ?museum
```

	?museum	?artist	?country
μ_1	Musee_d'Art_Moderne	Pablo_Picasso	France
μ_2	Museo_del_Prado	Raphael	Spain
⋮	⋮	⋮	⋮

Table 1: Generated Mappings for SPARQL Query `Q`

Here, V=`{?museum, ?artist, ?country}` and P is the conjunction of patterns in the `WHERE` clause then the evaluation of $[\![(\{?museum, ?artist, ?country\}, P)]\!]$ will generate the mappings shown in Table 1. Accordingly, $dom(\mu_i) = \{?museum, ?artist, ?country\}$. Here, $\mu_1(?museum) = Musee\_\ d'Art\_Moderne$, $\mu_1(?artist) = Pablo\_Picasso$ and $\mu_1(?country) = France$. We have Ov = {?`museum`} because it appears in the `VIEW BY` clause and Av = {?`artist`, ?`country`}. Figure 1a shows the generated view when Ov = {?`museum`} and in Figure 1b, we have; Ov = {?`artist`} and Av = {?`museum`, ?`country`}.

4.2 Designing a Formal Context of Answer Tuples

The results obtained by the query are in the form of set of tuples, which are then organized as a many-valued context.

Obtaining a Many-Valued Context (G, M, W, I)*:* As described previously, we have $Ov = \{?X\}$ then $\mu(?X) = \{X_i\}_{i \in \{1,\ldots,k\}}$, where X_i denote the values obtained for the object variable and the corresponding mapping is given as $\{\{?X \rightarrow X_i\}\}$. Finally, $G = \mu(?X) = \{X_i\}_{i \in \{1,\ldots,k\}}$. Let $Av = \{?Y, ?Z\}$ then $M = Av$ and the attribute values $W = \{\mu(?Y), \mu(?Z)\} = \{\{Y_i\}, \{Z_i\}\}_{i \in \{1,\ldots,k\}}$. The corresponding mapping for attribute variables are $\{\{?Y \rightarrow Y_i, ?Z \rightarrow Z_i\}\}$.

[7] The object here refers to the object in FCA.

In order to obtain a ternary relation, let us consider an object value $g_i \in G$ and an attribute value $w_i \in W$ then we have $(g_i, \text{“}?Y'', w_i) \in I$ iff $?Y(g_i) = w_i$, i.e., the value of g_i for attribute $?Y$ is w_i, $i \in \{1, \ldots, k\}$ as we have k values for $?Y$.

Obtaining Binary Context (G, M, I)*:* Afterwards, a conceptual scaling used for binarizing the many-valued context, in the form of (G, M, I). Finally, we have $G = \{X_i\}_{i \in \{1,\ldots,k\}}$, $M = \{Y_i\} \cup \{Z_i\}$ where $i \in \{1, \ldots, k\}$ for object variable $Ov = \{?X\}$. The binary context obtained after applying the above transformations to the SPARQL query answers w.r.t to object variable is called the *formal context of answer tuples* and is denoted by $\mathcal{K}_{tuple}$.

Example 3. In the example $Ov = \{?\texttt{museum}\}$, $Av = \{?\texttt{artist}, ?\texttt{country}\}$. The answers obtained by this query are organized into a many-valued context as follows: the distinct values of the object variable `?museum` are kept as a set of objects, so $G = \{MuseeduLouvre,\ MuseodelPrado, \ldots\}$, attribute variables provide $M = \{artist, country\}$, $W_1 = \{Raphael,\ LeonardoDaVinci, \ldots\}$ and $W_2 = \{France, Spain, UK, \ldots\}$ in a many-valued context. The obtained many-valued context is shown in Table 2. Finally, the obtained many-valued context is conceptually scaled to obtain a binary context shown in Table 3.

Museum	*Artist*	*Country*
Musee du Louvre	{Raphael, Leonardo Da Vinci, Caravaggio}	{France}
Musee d'Art Moderne	{Pablo Picasso}	{France}
Museo del Prado	{Raphael, Caravaggio, Francisco Goya}	{Spain}
National Gallery	{Leonardo Da Vinci, Caravaggio, Francisco Goya}	{UK}

Table 2: Many-Valued Context (Museum).

	Artist					*Country*		
Museum	Raphael	Da Vinci	Picasso	Caravaggio	Goya	France	Spain	UK
Musee du Louvre	×	×		×		×		
Musee d'Art Moderne			×			×		
Museo del Prado	×			×	×		×	
National Gallery		×		×	×			×

Table 3: Formal Context $\mathcal{K}_{tuple}$ w.r.t $?museum$.

The organization of the concept lattice is depending on the choice of object variable and the attribute variables. Then, to group the artists w.r.t the museums where their work is displayed and the location of the museums, the object variable would be `?artist` and the attribute variables will be `?museum` and `?country`. Then, the scaling can be performed for obtaining a formal context. In order to complete the set of attribute, domain knowledge can also be taken into account, such as the the ontology related to the type of artists or museums. This domain knowledge can be added with the help of pattern structures, an approach linked to FCA, on top of many-valued context without having to perform scaling. For the sake of simplicity, we do not discuss it in this paper.

4.3 Building a Concept Lattice

Once the context is designed, the concept lattice can be built using an FCA algorithm.There are some very efficient algorithms that can be used [7, 11]. However,

in the current implementation we use AddIntent [11] which is an incremental concept lattice construction algorithm. In case of large data iceberg lattices can be considered [10]. The use of `VIEW BY` clause activates the process of LBVA, which transforms the SPARQL query answers (tuples) to a formal context $\mathcal{K}_{tuples}$ through which a concept lattice is obtained which is referred to as a *Lattice-Based View*. A view on SPARQL query in section 2, i.e, a concept lattice corresponding to Table 3 is shown in Figure 1a.

4.4 Interpretation Operations over Lattice-Based Views

A formal context effectively takes into account the relations by keeping the inherent structure of the relationships present in LOD as object-attribute relation. When we build a concept lattice, each concept keeps a group of terms sharing some attribute (i.e., the relationship with other terms). This concept lattice can be navigated for searching and accessing particular LOD elements through the corresponding concepts within the lattice. It can be drilled down from general to specific concepts or rolled up to obtain the general ones which can be further interpreted by the domain experts. For example, in order to search for the museums where there is an exhibition of the paintings of `Caravaggio`, the concept lattice in Figure 1(a) is explored levelwise. It can be seen that the paintings of `Caravaggio` are displayed in `Musee du Louvre, Museo del Prado and National Gallery`. Now it can be further filtered by country, i.e., look for `French` museums displaying `Caravaggio`. The same lattice can be drilled down and `Musee du Louvre` as an answer can be retrieved. Next, to check the museums located in `France` and `Spain`, the roll up operation from the French Museums to the general concept containing all the museums with Caravaggio's painting can be applied and then the drill down operation to Museums in `France` or `Spain` displaying `Caravaggio` can be performed. The answer obtained will be `Musee du Louvre` and `Museo del Prado`.

A different perspective on the same set of answers can also be retrieved, meaning that the group of artists w.r.t museums and country. For selecting French museums according to the artists they display, the object variable will be $Ov = \{?artist\}$ and attribute variables will be $Av = \{?museum, ?country\}$. The lattice obtained in this case will be from Artist's perspective (see Figure 1b). Now, it is possible to retrieve `Musee du Louvre` and `Musee d'Art Moderne`, which are the French museums and to obtain a specific French museum displaying the work of `Leonardo Da Vinci` a specific concept can be selected which gives the answer `Musee du Louvre`.

FCA provides a powerful means for data analysis and knowledge discovery. `VIEW BY` can be seen as a clause that engulfs the original SPARQL query and enhances it's capabilities by providing *views* which can be reduced using iceberg concept lattices. Iceberg lattices provide the top most part of the lattice filtering out only general concepts. The concept lattice is still explored levelwise depending on a given threshold. Then, only concepts whose extent is sufficiently large are explored, i.e., the support of a concept corresponds to the cardinal of the extent. If further specific concepts are required the support threshold of the

iceberg lattices can be lowered and the resulting concept lattice can be explored levelwise.

Knowledge Discovery: Among the means provided by FCA for knowledge discovery, the Duquenne-Guigues basis of implications takes into account a minimal set of implications which represent all the implications (i.e., association rules with confidence 1) that can be obtained by accessing the *view* i.e., a concept lattice. For example, implications according to Figure 1(a) state that all the museums in the current context which display `Leonardo Da Vinci` also display `Caravaggio` (rule: `Leonardo Da Vinci` $\rightarrow$ `Caravaggio`). It also says that only the museums which display the work of `Caravaggio` display the work of `Leonardo Da Vinci` Such a rule can be interesting if the museums which display the work of both `Leonardo Da Vinci` and `Caravaggio` are to be retrieved. The rule `Goya, Raphael, Caravaggio` $\rightarrow$ `Spain` suggests that there exists a museum which have works of `Goya, Raphael, Caravaggio` only in Spain, more precisely `Museo Del Prado`. (These rules are generated from only the part of SPARQL query answers shown as a context in Table 3).

5 Experimentation

The experiments were conducted on real dataset. Our algorithm is implemented in Java using Jena[8] platform and the experiments were conducted on a laptop with 2.60 GHz Intel core i5 processor, 3.7 GB RAM running Ubuntu 12.04. We extracted the information about the movie with their genre and location using SPARQL query enhanced with `VIEW BY` clause. The experiment shows that even though the background knowledge (ontological information) was not extracted the views reveal the hidden hierarchical information contained in the SPARQL query answers and can be navigated accordingly. Moreover, it also shows that useful knowledge is extracted from the answers through the the views using $\mathcal{DG}-$Basis of implications. We also performed quantitative analysis where we discussed about the sparsity of the semantic web data. We also tested how our method scales with growing number of results. The number of answers obtained by YAGO were 100,000. The resulting view kept the classes of movies with respect to genre and location.

5.1 YAGO

The construction of YAGO ontology is based on the extraction of instances and hierarchical information from Wikipedia and Wordnet. In the current experiment, we sent a query to YAGO with the `VIEW BY` clause.

```
PREFIX rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
PREFIX yago: http://yago-knowledge.org/resource/
SELECT ?movie ?genre ?location WHERE {
```

[8] https://jena.apache.org/

```
   ?movie rdf:type yago:wordnet_movie_106613686 .
   ?movie yago:isLocatedIn ?location .
   ?movie rdf:type ?genre . }
VIEW BY ?movie
```

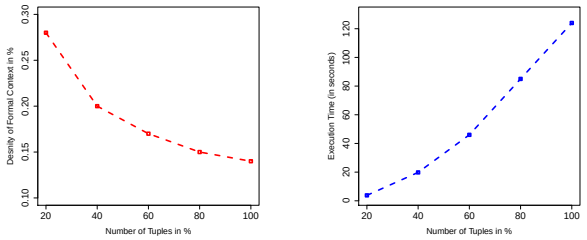
While querying YAGO it was observed that the genre and location information was also given in the ontology. The first level of the obtained view over the SPARQL query results over YAGO kept the groups of movies with respect to their languages. e.g., the movies with genre `Spanish Language Films`. However, as we further drill down in the concept lattice we get more specific categories which include the values from the location variable such as `Spain, Argentina` and `Mexico`. There were separate classes obtained for movies based on novels which were then further specialized by the introduction of the country attribute as we drill down the concept lattice. Finally with the help of lattice-based views, it can be concluded that the answers obtained by querying YAGO provides a clean categorization of movies by making use of the partially ordered relation between the concepts present in the concept lattice.

$\mathcal{DG}$-Basis of Implications: $\mathcal{DG}$-Basis of Implications for YAGO were calculated. The implications were filtered in three ways. Firstly, pruning was performed naively with respect to support threshold. Around 200 rules were extracted on support threshold of 0.2%. In order, to make the rules observable, the second type of filtering based on number of elements in the body of the rules was applied. All the implications which contained one item set in the body were selected. However, if there still are large number of implications to be observed then a third type of pruning can be applied which involved the selection of implications with different attribute type in head and body, e.g., in rule#1 head contains United States which is of type country and body contains the wikicategory. Such kind of pruning helps in finding attribute-attribute relations.

Table 4 contains some of the implications. Calculating $\mathcal{DG} - Basis$ of implications is actually useful in finding regularities in the SPARQL query answers which can not be discovered from the raw tuples obtained. For example, rule#1 states that `RKO picture films` is an American film production and distribution company as all the movies produced and distributed by them are from United States. Moreover, rule#2 says that all the movies in `Oriya language` are from `India`. This actually points to the fact that Oriya is one of many languages that is spoken in India. This rule also tells that Oriya language is only spoken in India. Rule#3 shows a link between a category from Wikipedia and Wordnet, which clearly says that the `wikicategory` is more specific than the `wordnet` category as `remake` is more general than `Film remakes`.

Impl. ID	Supp.	Implication
1.	96	`wikicategory RKO Pictures films` → `United States`
2.	46	`wikicategory Oriya language films` → `India`
3.	64	`wikicategory Film remakes` → `wordnet remake`

Table 4: Some implications from $\mathcal{DG}$-Basis of Implication (YAGO)



(a) Density of $\mathcal{K}_{YAGO}$ (b) Runtime for Building $\mathcal{L}_{YAGO}$

Fig. 2: Experimental Results.

5.2 Evaluation

Besides the qualitative evaluation of LBVA, we performed an empirical evaluation. The characteristics of the dataset are shown in Table 5. These concepts were pruned with the help of iceberg lattices and stability for qualitative analysis.

The plots for the experimentation are shown in Figure 2. Figure 2(a) shows a comparison between the number of tuples obtained and the density of the formal context. The density of the formal context is the proportion of pairs in I w.r.t the size $G \times M$. It has very low range for both the experiments, i.e., it ranges from 0.14% to 0.28%. This means in particular that the semantic web data is very sparse when considered in a formal context and deviates from the datasets usually considered for FCA (as they are dense). Here we can see that as the number of tuples increases the density of the formal context is decreasing which means that sparsity of the data also increases.

We also tested how our method scales with growing number of results. The number of answers obtained by YAGO were 100,000. Figure 2(b) illustrate the execution time for building the concept lattice w.r.t the number of tuples obtained. The execution time ranges from 20 to 100 seconds, it means that the the concept lattices were built in an efficient way and large data can be considered for these kinds of experiments. Usually the computation time for building concept lattices depends on the density of the formal context but in the case of semantic web data, as the density is not more than 1%, the computation completely depends on the number of objects obtained which definitely increase with the increase in the number of tuples (see Table 5).

No. of Tuples	$\|G\|$	$\|M\|$	**No. of Concepts**
20%	3657	2198	7885
40%	6783	3328	19019
60%	9830	4012	31264
80%	12960	4533	43510
100%	15272	4895	55357

Table 5: Characteristics of Datasets (YAGO)

6 Conclusion and Discussion

In LBVA, we introduce a classification framework based on FCA for the set of tuples obtained as a result of SPARQL queries over LOD. In this way, a view

is organized as a concept lattice built through the use of `VIEW BY` clause that can be navigated where information retrieval and knowledge discovery can be performed. Several experiments show that LBVA is rather tractable and can be applied to large data.

For future work, we are interested in extending the `VIEW BY` clause by including the available background knowledge of the resources using the formalism of pattern structures [6]. Moreover, we intend to use implications for completing the background knowledge. We also intend to use pattern structures with a graph description for each considered object, where the graph is the set of all triples accessible w.r.t reference object.

References

1. Marcelo Arenas, Claudio Gutierrez, and Jorge Pérez. Foundations of rdf databases. In Sergio Tessaris, Enrico Franconi, Thomas Eiter, Claudio Gutierrez, Siegfried Handschuh, Marie-Christine Rousset, and Renate A. Schmidt, editors, *Reasoning Web*, volume 5689 of *Lecture Notes in Computer Science*, pages 158–204. Springer, 2009.
2. Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.
3. Claudio Carpineto, Stanislaw Osiński, Giovanni Romano, and Dawid Weiss. A survey of web clustering engines. *ACM Comput. Surv.*, 41(3):17:1–17:38, 2009.
4. Claudio Carpineto and Giovanni Romano. *Concept data analysis - theory and applications.* Wiley, 2005.
5. Claudia d'Amato, Nicola Fanizzi, and Agnieszka Lawrynowicz. Categorize by: Deductive aggregation of semantic web query results. In Lora Aroyo, Grigoris Antoniou, Eero Hyvönen, Annette ten Teije, Heiner Stuckenschmidt, Liliana Cabral, and Tania Tudorache, editors, *ESWC (1)*, volume 6088 of *Lecture Notes in Computer Science*, pages 91–105. Springer, 2010.
6. Bernhard Ganter and Sergei O. Kuznetsov. Pattern structures and their projections. In Harry S. Delugach and Gerd Stumme, editors, *ICCS*, volume 2120 of *Lecture Notes in Computer Science*, pages 129–142. Springer, 2001.
7. Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis: Mathematical Foundations.* Springer, Berlin/Heidelberg, 1999.
8. J.-L. Guigues and V. Duquenne. Familles minimales d'implications informatives résultant d'un tableau de données binaires. *Mathématiques et Sciences Humaines*, 95:5–18, 1986.
9. Sergei O. Kuznetsov. On stability of a Formal Concept. *Ann. Math. Artif. Intell.*, 49(1-4):101–115, 2007.
10. Gerd Stumme, Rafik Taouil, Yves Bastide, and Lotfi Lakhal. Conceptual clustering with iceberg concept lattices. In R. Klinkenberg, S. Rüping, A. Fick, N. Henze, C. Herzog, R. Molitor, and O. Schröder, editors, *Proc. GI-Fachgruppentreffen Maschinelles Lernen (FGML'01)*, Universität Dortmund 763, October 2001.
11. Dean van der Merwe, Sergei A. Obiedkov, and Derrick G. Kourie. Addintent: A new incremental algorithm for constructing concept lattices. In Peter W. Eklund, editor, *ICFCA*, Lecture Notes in Computer Science, pages 372–385. Springer, 2004.

A generalized framework to consider positive and negative attributes in formal concept analysis.

J. M. Rodriguez-Jimenez, P. Cordero, M. Enciso and A. Mora

Universidad de Málaga, Andalucía Tech, Spain.
{pcordero,enciso}@uma.es
{amora,jmrodriguez}@ctima.uma.es

Abstract. In Formal Concept Analysis the classical formal context is analized taking into account only the positive information, i.e. the presence of a property in an object. Nevertheless, the non presence of a property in an object also provides a significant knowledge which can only be partially considered with the classical approach. In this work we have modified the derivation operators to allow the treatment of both, positive and negative attributes which come from respectively, the presence and absence of the properties. In this work we define the new operators and we prove that they are a Galois connection. Finally, we have also studied the correspondence between the formal context in the new framework and the extended concept lattice, providing new interesting properties.

1 Introduction

Data analysis of information is a well established discipline with tools and techniques well developed to challenge the identification of hide patterns in the data. Data mining, and general Knowledge Discovering, helps in the decision making process using pattern recognition, clustering, association and classification methods. One of the popular approaches used to extract knowledge is mining the patterns of the data expressed as implications (functional dependencies in database community) or association rules.

Traditionally, implications and similar notions have been built using the *positive* information, i.e. information induced by the presence of attributes in objects. In Manilla et al. [6] an extended framework for enriched rules was introduced, considering negation, conjunction and disjunction. Rules with negated attributes were also considered in [1]: "if we buy caviar, then we do not buy canned tuna".

In the framework of formal concept analysis, some authors have proposed the mining of implications with positive and negative attributes from the apposition of the context and its negation $(\mathbb{K}|\overline{\mathbb{K}})$ [2, 4]. Working with $(\mathbb{K}|\overline{\mathbb{K}})$ conduits to a huge exponential problem and also as R. Missaoui et.al. shown in [9] real applications use to have sparse data in the context $\mathbb{K}$ whereas dense data in $\overline{\mathbb{K}}$ (or viceversa), and therefore *"generate a huge set of candidate itemsets and a tremendous set of uninteresting rules"*.

ISBN 978–80–8152–159–1, Institute of Computer Science, Pavol Jozef Šafárik University in Košice, 2014.

R. Missaoui et al. [7, 8] propose the mining from a formal context $\mathbb{K}$ of a subset of all *mixed* implications, i.e. implication with positive and negative attributes, representing the presence and absence of properties. As far as we know, the approach of these authors uses, for first time in this problem, a set of inference rules to manage negative attributes.

In [11] we followed the line proposed by Missaoui and presented an algorithm, based on the NextClosure algorithm, that allows to obtain mixed implications. The proposed algorithm returns a feasible and complete basis of mixed implications by performing a reduced number of requests to the formal context. Beyond the benefits provided by the inclusion of negative attributes in terms of expressiveness, Revenko and Kuznetsov [10] use negative attributes to tackle the problem of finding some types of errors in new object intents is introduced. Their approach is based on finding implications from an implication basis of the context that are not respected by a new object. Their work illustrates the great benefit that a general framework for negative and positive attributes would provide.

In this work we propose a deeper study of the algebraic framework for Formal Concept Analysis taking into account positive and negative information. The first step is to consider an extension of the classical derivation operators, proving to be Galois connection. As in the classical framework, this fact will allows to built the two usual dual concept lattices, but in this case, as we shall see, the correspondence among concept lattices and formal contexts reveal several characteristics which induce interesting properties. The main aim of this work is to establish a formal full framework which allows to develop in the future new methods and techniques dealing with positive and negative information.

In Section 2 we present the background of this work: the notions related with formal concept analysis and negative attributes. Section 3 introduces the main results which constitute the contribution of this paper.

2 Preliminaries

2.1 Formal Concept Analysis

In this section, the basic notions related with Formal Concept Analysis (FCA) [12] and attribute implications are briefly presented. See [3] for a more detailed explanation. A *formal context* is a triple $\mathbb{K} = \langle G, M, I\rangle$ where G and M are finite non-empty sets and $I \subseteq G \times M$ is a binary relation. The elements in G are named objects, the elements in M attributes and $\langle g, m\rangle \in I$ means that the object g has the attribute m. From this triple, two mappings $\uparrow\colon 2^G \to 2^M$ and $\downarrow\colon 2^M \to 2^G$, named derivation operators, are defined as follows: for any $X \subseteq G$ and $Y \subseteq M$,

$$X^{\uparrow} = \{m \in M \mid \langle g, m\rangle \in I \text{ for all } g \in X\} \tag{1}$$

$$Y^{\downarrow} = \{g \in G \mid \langle g, m\rangle \in I \text{ for all } m \in Y\} \tag{2}$$

$X^{\uparrow}$ is the subset of all attributes shared by all the objects in X and $Y^{\downarrow}$ is the subset of all objects that have the attributes in Y. The pair $(\uparrow, \downarrow)$ constitutes

a Galois connection between 2^G and 2^M and, therefore, both compositions are closure operators.

A pair of subsets $\langle X, Y\rangle$ with $X \subseteq G$ and $Y \subseteq M$ such $X^{\uparrow} = Y$ and $Y^{\downarrow} = X$ is named a *formal concept*. X is named the *extent* and Y the *intent* of the concept. These extents and intents coincide with closed sets wrt the closure operators because $X^{\uparrow\downarrow} = X$ and $Y^{\downarrow\uparrow} = Y$. Thus, the set of all formal concepts is a lattice, named *concept lattice*, with the relation

$$\langle X_1, Y_1\rangle \leq \langle X_2, Y_2\rangle \text{ if and only if } X_1 \subseteq X_2 \text{ (or equivalently, } Y_2 \subseteq Y_1) \quad (3)$$

This concept lattice will be denoted by $\mathfrak{B}(G, M, I)$.

The concept lattice can be characterized in terms of *attribute implications* being expressions $A \to B$ where $A, B \subseteq M$. An implication $A \to B$ holds in a context $\mathbb{K}$ if $A^{\downarrow} \subseteq B^{\downarrow}$. That is, any object that has all the attributes in A has also all the attributes in B. It is well known that the sets of attribute implications that are valid in a context satisfies the Armstrong's Axioms:

`[Ref]` Reflexivity: If $B \subseteq A$ then $\vdash A \to B$.
`[Augm]` Augmentation: $A \to B \vdash A \cup C \to B \cup C$.
`[Trans]` Transitivity: $A \to B, B \to C \vdash A \to C$.

A set of implications Σ is considered an *implicational system for* $\mathbb{K}$ if: an implication holds in $\mathbb{K}$ if and only if it can be inferred, by using Armstrong's Axioms, from Σ.

Armstrong's axioms allow us to define the closure of attribute sets wrt an implicational system (the closure of a set A is usually denoted as A^{+}) and it is well-known that closed sets coincide with intents. On the other hand, several kind of implicational systems has been defined in the literature being the most used the so-called Duquenne-Guigues (or stem) basis [5]. This basis satisfies that its cardinality is minimum among all the implicational systems and can be obtained from a context by using the renowned NextClosure Algorithm [3].

2.2 Negatives attributes

As we have mentioned in the introduction, classical FCA only discover knowledge limited to positive attributes in the context, but it does not consider information relative to the absence of properties (attributes). Thus, the Duquenne-Guigues basis obtained from Table 1 is $\{e \to bc, d \to c, bc \to e, a \to b\}$. Moreover, the implications $b \to c$ and $b \to d$ do not hold in Table 1 and therefore they can not be derived from the basis by using the inference system. Nevertheless, both implications correspond with different situations. In the first case, some objects have attributes b and c (e.g. objects o_1 and o_3) whereas another objects (e.g. o_2) have the attribute b and do not have c. On the other side, in the second case, any object that has the attribute b does not have the attribute d.

A more general framework is necessary to deal with this kind of information. In [11], we have tackled this issue focusing on the problem of mining implication with positive and negative attributes from formal contexts. As a conclusion of

I	a	b	c	d	e
o_1		×	×		×
o_2	×	×			
o_3		×	×		×
o_4			×	×	

Table 1. A formal context

that work we emphasized the necessity of a full development of an algebraic framework.

First, we begin with the introduction of an extended notation that allows us to consider the negation of attributes. From now on, the set of attributes is denoted by M, and its elements by the letter m, possibly with subindexes. That is, the lowercase character m is reserved for positive attributes. We use $\overline{m}$ to denote the negation of the attribute m and $\overline{M}$ to denote the set $\{\overline{m} \mid m \in M\}$ whose elements will be named negative attributes.

Arbitrary elements in $M \cup \overline{M}$ are going to be denoted by the first letters in the alphabet: a, b, c, etc. and $\overline{a}$ denotes the opposite of a. That is, the symbol a could represent a positive or a negative attribute and, if $a = m \in M$ then $\overline{a} = \overline{m}$ and if $a = \overline{m} \in \overline{M}$ then $\overline{a} = m$.

Capital letters A, B, C,... denote subsets of $M \cup \overline{M}$. If $A \subseteq M \cup \overline{M}$, then $\overline{A}$ denotes the set of the opposite of attributes $\{\overline{a} \mid a \in A\}$ and the following sets are defined:

- $\mathrm{Pos}(A) = \{m \in M \mid m \in A\}$
- $\mathrm{Neg}(A) = \{m \in M \mid \overline{m} \in A\}$
- $\mathrm{Tot}(A) = \mathrm{Pos}(A) \cup \mathrm{Neg}(A)$

Note that $\mathrm{Pos}(A), \mathrm{Neg}(A), \mathrm{Tot}(A) \subseteq M$.

Once we have introduced the notation, we are going to summarize some results concerning the mining of knowledge from contexts in terms of implications with negative and positive attributes [11]. A trivial approach could be obtained by adding new columns to the context with the opposite of the attributes [4]. That is, given a context $\mathbb{K} = \langle G, M, I\rangle$, a new context $(\mathbb{K}|\overline{\mathbb{K}}) = \langle G, M \cup \overline{M}, I \cup \overline{I}\rangle$ is considered, where $\overline{I} = \{\langle g, \overline{m}\rangle \mid g \in G, m \in M, \langle g, m\rangle \notin I\}$. For example, if $\mathbb{K}$ is the context depicted in Table 1, the context $(\mathbb{K}|\overline{\mathbb{K}})$ is those presented in Table 2. Obviously, the classical framework and its corresponding machinery can be used to manage the new context and, in this (direct) way, negative attributes are considered. However, this rough approach induces a non trivial growth of the formal context and, consequently, algorithms have a worse performance.

In our opinion, a deeper study was done by R. Missaoui et al. in [7] where an evolved approach has been provided. For first time –as far as we know– inference rules for the management of positive and negative attributes are introduced [8]. The authors also developed new methods to mine mixed attribute implications by means of the key notion [9].

$I \cup \overline{I}$	a	b	c	d	e	$\overline{a}$	$\overline{b}$	$\overline{c}$	$\overline{d}$	$\overline{e}$
o_1		×	×		×	×			×	
o_2	×	×						×	×	×
o_3		×	×		×	×			×	
o_4			×	×		×	×			×

Table 2. The formal context $(\mathbb{K}|\overline{\mathbb{K}})$

In [11], we have developed a method to mine mixed implications whose main goal has been to avoid the management of the large $(\mathbb{K}|\overline{\mathbb{K}})$ contexts, so that the performance of the corresponding method has a controlled cost.

First, we extend the definitions of derivation operators, formal concept and attribute implication.

Definition 1. *Let $\mathbb{K} = \langle G, M, I\rangle$ be a formal context. We define the operators $\Uparrow\colon 2^G \to 2^{M\cup\overline{M}}$ and $\Downarrow\colon 2^{M\cup\overline{M}} \to 2^G$ as follows: for $X \subseteq G$ and $Y \subseteq M \cup \overline{M}$,*

$$\begin{aligned} X^{\Uparrow} = &\{m \in M \mid \langle g, m\rangle \in I \text{ for all } g \in X\} \\ &\cup \{\overline{m} \in \overline{M} \mid \langle g, m\rangle \notin I \text{ for all } g \in X\} \end{aligned} \tag{4}$$

$$\begin{aligned} Y^{\Downarrow} = &\{g \in G \mid \langle g, m\rangle \in I \text{ for all } m \in Y\} \\ &\cap \{g \in G \mid \langle g, m\rangle \notin I \text{ for all } \overline{m} \in Y\} \end{aligned} \tag{5}$$

Definition 2. *Let $\mathbb{K} = \langle G, M, I\rangle$ be a formal context. A* mixed formal concept *in $\mathbb{K}$ is a pair of subsets $\langle X, Y\rangle$ with $X \subseteq G$ and $Y \subseteq M \cup \overline{M}$ such $X^{\Uparrow} = Y$ and $Y^{\Downarrow} = X$.*

Definition 3. *Let $\mathbb{K} = \langle G, M, I\rangle$ be a formal context and let $A, B \subseteq M \cup \overline{M}$, the context $\mathbb{K}$ satisfies a* mixed attribute implication *$A \to B$, denoted by $\mathbb{K} \models A \to B$, if $A^{\Downarrow} \subseteq B^{\Downarrow}$.*

For example, in Table 1, as we previously mentioned, two different situations were presented. Thus, in this new framework we have that $\mathbb{K} \not\models b \to d$ and $\mathbb{K} \models b \to \overline{d}$ whereas $\mathbb{K} \not\models b \to c$ either $\mathbb{K} \not\models b \to \overline{c}$.

Now, we are going to introduce the mining method for mixed attribute implications. The method is strongly based on the set of inference rules built by supplementing Armstrong's axioms with the following ones, introduced in [8]: let $a, b \in M \cup \overline{M}$ and $A \subseteq M \cup \overline{M}$,

`[Cont]` Contradiction: $\vdash a\overline{a} \to M\overline{M}$.
`[Rft]` Reflection: $Aa \to b \vdash A\overline{b} \to \overline{a}$.

The closure of an attribute set A wrt a set of mixed attribute implications Σ, denoted as $A^{+\!\!+}$, is defined as the biggest set such that $A \to A^{+\!\!+}$ can be inferred from Σ by using Armstrong's Axioms plus `[Cont]` and `[Rft]`. Therefore, a mixed implication $A \to B$ can be inferred from Σ if and only if B is a subset of the closure of A, i.e. $B \subseteq A^{+\!\!+}$.

The proposed mining method, depicted in Algorithm 1, uses the inference rules in such a way that it is not centered around the notion of key, but it extends, in a proper manner, the classical NextClosure algorithm [3].

Algorithm 1: Mixed Implications Mining

Data: $\mathbb{K} = \langle G, M, I \rangle$
Result: Σ set of implications

```
1  begin
2      Σ := ∅;
3      Y := ∅;
4      while Y < M do
5          foreach X ⊆ Y do
6              A := (Y ∖ X) ∪ X̄;
7              if Closed(A, Σ) then
8                  C := A^{⇓⇑};
9                  if A ≠ C then Σ := Σ ∪ {A → C ∖ A}
10         Y := Next(Y) // i.e. successor of Y in the lectic order
11     return Σ
12 end
```

The algorithm to calculate the mixed implicational system doesn't need to exhaustive traverse all the subsets of mixed attributes, but only those ones that are closed w.r.t. the set of implications previously computed. The **Closed** function is defined having linear cost and is used to discern when a set of attributes is not closed and thus, the context is not visited in this case.

Function `Closed(`A`,`Σ`)`: boolean

Data: $A \subseteq M \cup \overline{M}$ with $\mathrm{Pos}(A) \cap \mathrm{Neg}(A) = \varnothing$ and Σ being a set of mixed implications.
Result: '`true`' if A is closed wrt Σ or '`false`' otherwise.

```
1  begin
2      foreach B → C ∈ Σ do
3          if B ⊆ A and C ⊈ A then exit and return false if  B ∖ A = {a},
           A ∩ C̄ ≠ ∅, and ā ∉ A then exit and return false
4      return true
5  end
```

3 Mixed concept lattices

As we have mentioned, the goal of this paper is to develop a deep study of the generalized algebraic framework. In this section we are going to introduce the main results of this paper providing the properties of the generalized concept lattice. The main pillar of our new framework are the two derivation operators introduced in Equations 4 and 5. The following theorem ensures that the pair of these operators is a Galois connection:

Theorem 1. *Let $\mathbb{K} = \langle G, M, I\rangle$ be a formal context. The pair of derivation operators $(\Uparrow, \Downarrow)$ introduced in Definition 1 is a Galois Connection.*

Proof. We need to prove that, for all subsets $X \subseteq G$ and $Y \subseteq M \cup \overline{M}$,

$$X \subseteq Y^{\Downarrow} \text{ if and only if } Y \subseteq X^{\Uparrow}$$

First, assume $X \subseteq Y^{\Downarrow}$. For all $a \in Y$, we distinguish two cases:

1. If $a \in \mathrm{Pos}(Y)$, exists $m \in M$ with $a = m$ and, for all $g \in X$, since $X \subseteq Y^{\Downarrow}$, $\langle g, m\rangle \in I$ and therefore $a = m \in X^{\Uparrow}$.
2. If $a \in \mathrm{Neg}(Y)$, exits $m \in M$ with $a = \overline{m}$ and, for all $g \in X$, since $X \subseteq Y^{\Downarrow}$, $\langle g, m\rangle \notin I$ and therefore $a = \overline{m} \in X^{\Uparrow}$.

Conversely, assume $Y \subseteq X^{\Uparrow}$ and $g \in X$. To ensure that $g \in Y^{\Downarrow}$, we need to prove that $\langle g, a\rangle \in I$ for all $a \in \mathrm{Pos}(Y)$ and $\langle g, \overline{a}\rangle \notin I$ for all $a \in \mathrm{Neg}(Y)$, which is straightforward from $Y \subseteq X^{\Uparrow}$. □

Therefore, above theorem ensures that ${}^{\Uparrow\circ\Downarrow}$ and ${}^{\Downarrow\circ\Uparrow}$ are closure operators. Furthermore, as in the classical case, both closure operators provide two dually isomorphic lattices. We denote by $\mathfrak{B}^{\sharp}(G, M, I)$ to the lattice of mixed concepts with the relation

$$\langle X_1, Y_1\rangle \leq \langle X_2, Y_2\rangle \text{ iff } X_1 \subseteq X_2 \text{ (or equivalently, iff } Y_1 \supseteq Y_2)$$

Moreover, as in the classical FCA, mixed implications and mixed concept lattice make up the two sides of the same coin, i.e. the information mined from the mixed formal context may be dually represented by means of a set of mixed attribute implications or a mixed concept lattice.

As we shall see later in this section, unlike the classical FCA, mixed concept lattices are restricted to an specific lattice subclass. There exist specific properties that lattices may observe to be considered a valid lattice structure which corresponds to a mixed formal context. In fact, this is one of the main goal of this paper, the characterization of the lattices in the mixed formal concept analysis.

In Table 3 six different lattices are depicted. In the classical framework, all of them may be associated with formal contexts, i.e. in the classical framework any lattice corresponds with a collection of formal context. Nevertheless, in the mixed attribute framework this property does not hold anymore. Thus, in Table 3, as we shall prove later in this paper, lattices 3 and 5 cannot be associated with a mixed formal context.

The following two definitions characterizes two kind of significant sets of attributes that will be used later:

Definition 4. *Let $\mathbb{K} = \langle G, M, I\rangle$ be a formal context. A set $A \subseteq M \cup \overline{M}$ is named consistent set if* $\mathrm{Pos}(A) \cap \mathrm{Neg}(A) = \varnothing$.

The set of consistent sets are going to be denoted by $\mathbb{C}$tts, i.e.

$$\mathbb{C}\mathrm{tts} = \{A \subseteq M \cup \overline{M} \mid \mathrm{Pos}(A) \cap \mathrm{Neg}(A) = \varnothing\}$$

If $A \in \mathbb{C}$tts then $|A| \leq |M|$ and, in the particular case where $|A| = |M|$, we have $\mathrm{Tot}(A) = M$. This situation induces the notion of *full* set:

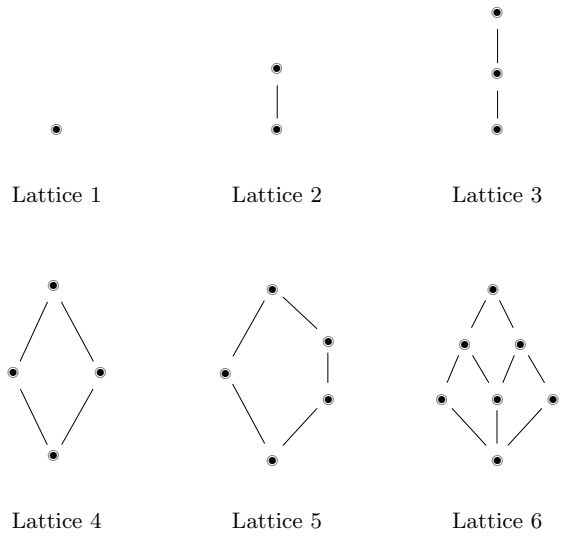


Table 3. Scheletons of some lattices

Definition 5. *Let $\mathbb{K} = \langle G, M, I\rangle$ be a formal context. A set $A \subseteq M \cup \overline{M}$ is said to be full consistent set if $A \in \mathbb{C}\text{tts}$ and $\text{Tot}(A) = M$.*

The following lemma, which characterize the boundary cases, is straightforward from Definition 1.

Lemma 1. *Let $\mathbb{K} = \langle G, M, I\rangle$ be a formal context. Then $\varnothing^{\Uparrow} = M \cup \overline{M}$, $\varnothing^{\Downarrow} = G$ and $(M \cup \overline{M})^{\Downarrow} = \varnothing$.*

In the classical framework, the concept lattice $\mathcal{B}(G, M, I)$ is bounded by $\langle M^{\downarrow}, M\rangle$ and $\langle G, G^{\uparrow}\rangle$. However, in this generalized framework, as a direct consequence from above lemma, the lower and upper bounds of $\mathcal{B}^{\sharp}(G, M, I)$ are $\langle \varnothing, M \cup \overline{M}\rangle$ and $\langle G, G^{\Uparrow}\rangle$ respectively.

Lemma 2. *Let $\mathbb{K} = \langle G, M, I\rangle$ be a formal context. The following properties hold:*

1. *For all $g \in G$, $\{g\}^{\Uparrow}$ is a full consistent set.*
2. *For all $g_1, g_2 \in G$, if $g_1 \in \{g_2\}^{\Uparrow\Downarrow}$ then $\{g_1\}^{\Uparrow} = \{g_2\}^{\Uparrow}$.* [1]
3. *For all $X \subseteq G$, $X^{\Uparrow} = \bigcap_{g\in X}\{g\}^{\Uparrow}$.*

Proof. 1. It is obvious because, for all $m \in M$, $\langle g, m\rangle \in I$ or $\langle g, m\rangle \notin I$ and $\{g\}^{\Uparrow} = \{m \in M \mid \langle g, m\rangle \in I\} \cup \{\overline{m} \in \overline{M} \mid \langle g, m\rangle \notin I\}$ being a disjoint union. Thus, $\text{Tot}(\{g\}^{\Uparrow}) = M$ and $\text{Pos}(\{g\}^{\Uparrow}) \cap \text{Neg}(\{g\}^{\Uparrow}) = \varnothing$.

[1] That is, g_1 and g_2 have exactly the same attributes.

2. Since $(\Uparrow, \Downarrow)$ is a Galois connection, $g_1 \in \{g_2\}^{\Uparrow\Downarrow}$ (i.e. $\{g_1\} \subseteq \{g_2\}^{\Uparrow\Downarrow}$) implies $\{g_2\}^{\Uparrow} \subseteq \{g_1\}^{\Uparrow}$. Moreover, by item 1, both $\{g_1\}^{\Uparrow}$ and $\{g_2\}^{\Uparrow}$ are full consistent and, therefore, $\{g_1\}^{\Uparrow} = \{g_2\}^{\Uparrow}$.
3. In the same way that occurs in the classical framework, since $(\Uparrow, \Downarrow)$ is a Galois connection between $(2^G, \subseteq)$ and $(2^{M \cup \overline{M}}, \subseteq)$, for any $X \subseteq G$, we have that $X^{\Uparrow} = \left(\bigcup_{g \in X} \{g\}\right)^{\Uparrow} = \bigcap_{g \in X} \{g\}^{\Uparrow}$. □

The above elementary lemmas lead to the following theorem emphasizing a significant difference with respect to the classical construction and it focuses on how the inclusion of new objects influences the structure of mixed concept lattice.

Theorem 2. *Let $\mathbb{K} = \langle G, M, I \rangle$ be a formal context, g_0 be a new object, i.e. $g_0 \notin G$, and $Y \subseteq M$ be the set of attributes that g_0 satisfies. Then, there exists $g \in G$ such that $\{g\}^{\Uparrow} = \{g_0\}^{\Uparrow}$ if and only if there exists an isomorphism between $\mathcal{B}^{\sharp}(G, M, I)$ and $\mathcal{B}^{\sharp}(G \cup \{g_0\}, M, I \cup \{\langle g_0, m \rangle \mid m \in Y\})$.*

That is, if a new different object (an object that differs at least in one attribute from each object in the context) is added to the formal context then the mixed concept lattice changes.

Proof. Obviously, if there exists $g \in G$ such that $\{g\}^{\Uparrow} = \{g_0\}^{\Uparrow}$, from Lemma 2 g and g_0 have exactly the same attributes, and moreover the lattices $\mathcal{B}^{\sharp}(G, M, I)$ and $\mathcal{B}^{\sharp}(G \cup \{g_0\}, M, I \cup \{\langle g_0, m \rangle \mid m \in Y\})$ are isomorphic.

Conversely, if the mixed concept lattices are isomorphic, there exists $X \subseteq G$ such that the closed set $X^{\Uparrow}$ in $\mathcal{B}^{\sharp}(G, M, I)$ coincides with $\{g_0\}^{\Uparrow}$. Thus, in the mixed concept lattice $\mathcal{B}^{\sharp}(G \cup \{g_0\}, M, I \cup \{\langle g_0, m \rangle \mid m \in X\})$, by Lemma 2, we have that $\{g_0\}^{\Uparrow} = X^{\Uparrow} = \cap_{g \in X} \{g\}^{\Uparrow}$. Moreover, since $\{g_0\}^{\Uparrow}$ is a full consistent set, $X \neq \varnothing$ because of, by Lemma 1, $\varnothing^{\Uparrow} = M \cup \overline{M}$. Therefore, for all $g \in X$ (there exists at least one $g \in X$), $g_0 \in \{g\}^{\Uparrow}$ and, by Lemma 2, $\{g\}^{\Uparrow} = \{g_0\}^{\Uparrow}$. □

Example 1. Let $\mathbb{K}_1 = (\{g1, g2\}, \{a, b, c\}, I_1)$ and $\mathbb{K}_2 = (\{g1, g2, g3\}, \{a, b, c\}, I_2)$ be formal contexts where I_1 and I_2 are the binary relations depicted in Table 4. Note that $\mathbb{K}_2$ is built from $\mathbb{K}_1$ by adding the new object $g3$. In the classical framework, the concept lattices $\mathcal{B}(\{g1, g2\}, \{a, b, c\}, I_1)$ and $\mathcal{B}(\{g1, g2, g3\}, \{a, b, c\}, I_2)$ are isomorphic. See Figure 1.

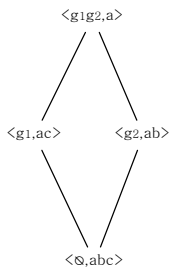
I_1	a	b	c
$g1$	×		×
$g2$	×	×	

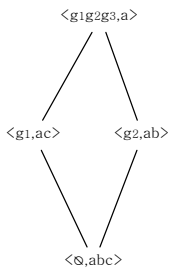
I_2	a	b	c
$g1$	×		×
$g2$	×	×	
$g3$	×		

Table 4. The formal contexts $\mathbb{K}_1$ and $\mathbb{K}_2$

However, the lattices of mixed concepts cannot be isomorphic because the new object $g3$ is not a repetition of one existing object. See Figure 2.

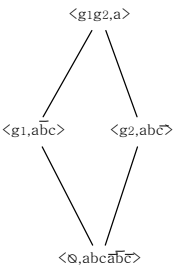
The following theorem characterizes the atoms of the new concept lattice $\mathcal{B}^{\sharp}$.

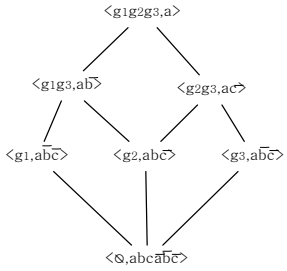




$\mathcal{B}(\{g1, g2\}, \{a, b, c\}, I_1)$ $\mathcal{B}(\{g1, g2, g3\}, \{a, b, c\}, I_2)$

Fig. 1. Lattices obtained in the classical framework





$\mathcal{B}^\sharp(\{g1, g2\}, \{a, b, c\}, I_1)$ $\mathcal{B}^\sharp(\{g1, g2, g3\}, \{a, b, c\}, I_2)$

Fig. 2. Lattices obtained in the extended framework

Theorem 3. *Let $\mathbb{K} = \langle G, M, I\rangle$ be a formal context. The set of atoms in the lattice $\mathcal{B}^\sharp(G, M, I)$ is $\{\langle\{g\}^{\Uparrow\Downarrow}, \{g\}^{\Uparrow}\rangle \mid g \in G\}$.*

Proof. First, fixed $g_0 \in G$, we are going to prove that the mixed concept $\langle\{g_0\}^{\Uparrow\Downarrow}, \{g_0\}^{\Uparrow}\rangle$ is an atom in $\mathcal{B}^\sharp(G, M, I)$. If $\langle X, Y\rangle$ is a mixed concept such that $\langle\varnothing, M \cup \overline{M}\rangle < \langle X, Y\rangle \leq \langle\{g_0\}^{\Uparrow\Downarrow}, \{g_0\}^{\Uparrow}\rangle$, then $\{g_0\}^{\Uparrow} \subseteq Y = X^{\Uparrow} \subsetneq M \cup \overline{M}$. By Lemma 2, $\{g_0\}^{\Uparrow} \subseteq X^{\Uparrow} = \bigcap_{g\in X}\{g\}^{\Uparrow}$. Moreover, for all $g \in X \neq \varnothing$, by Lemma 2, both $\{g_0\}^{\Uparrow}$ and $\{g\}^{\Uparrow}$ are full consistent sets and, since $\{g_0\}^{\Uparrow} \subseteq \{g\}^{\Uparrow}$, we have $\{g_0\}^{\Uparrow} = \{g\}^{\Uparrow}$. Therefore, $\{g_0\}^{\Uparrow} = X^{\Uparrow} = Y$ and $\langle X, Y\rangle = \langle\{g_0\}^{\Uparrow\Downarrow}, \{g_0\}^{\Uparrow}\rangle$.

Conversely, if $\langle X, Y\rangle$ is an atom in $\mathcal{B}^\sharp(G, M, I)$, then $X \neq \varnothing$ and there exists $g_0 \in X$. Since $(\Uparrow, \Downarrow)$ is a Galois connection, $\{g_0\}^{\Uparrow} \supseteq X^{\Uparrow} = Y$ and, therefore, $\langle\{g_0\}^{\Uparrow\Downarrow}, \{g_0\}^{\Uparrow}\rangle \leq \langle X, Y\rangle$. Finally, since $\langle X, Y\rangle$ is an atom, we have that $\langle X, Y\rangle = \langle\{g_0\}^{\Uparrow\Downarrow}, \{g_0\}^{\Uparrow}\rangle$. □

The following theorem establishes the characterization of the mixed concept lattice, proving that atoms and join irreducible elements are the same notions.

Theorem 4. *Let $\mathbb{K} = \langle G, M, I\rangle$ be a formal context. Any element in $\mathcal{B}^\sharp(G, M, I)$ is $\vee$-irreducible if and only if it is an atom.*

Proof. Obviously, any atom is $\vee$-irreducible. We are going to prove that any $\vee$-irreducible element belongs to $\{\langle\{g\}^{\Uparrow\Downarrow}, \{g\}^{\Uparrow}\rangle \mid g \in G\}$. Let $\langle X, Y\rangle$ be a $\vee$-irreducible element. Then, by Lemma 2, $Y = X^{\Uparrow} = \bigcap_{g\in X}\{g\}^{\Uparrow}$. Let X' be the smaller set such that $X' \subseteq X$ and $Y = \bigcap_{g\in X'}\{g\}^{\Uparrow}$. If X' is a singleton, then $\langle X, Y\rangle \in \{\langle\{g\}^{\Uparrow\Downarrow}, \{g\}^{\Uparrow}\rangle \mid g \in G\}$.

Finally, we prove that X' is necessarily a singleton. In other case, a bipartition of X' in two disjoint sets Z_1 and Z_2 can be made satisfying $Z_1 \cup Z_2 = X'$, $Z_1 \neq \varnothing$, $Z_2 \neq \varnothing$ and $Z_1 \cap Z_2 \neq \varnothing$. Then, $Y = \bigcap_{g\in Z_1}\{g\}^{\Uparrow} \cap \bigcap_{g\in Z_2}\{g\}^{\Uparrow} = Z_1^{\Uparrow} \cap Z_2^{\Uparrow}$ and so $\langle X, Y\rangle = \langle Z_1^{\Uparrow\Downarrow}, Z_1^{\Uparrow}\rangle \vee \langle Z_2^{\Uparrow\Downarrow}, Z_2^{\Uparrow}\rangle$ and $Z_1^{\Uparrow} \neq Y \neq Z_2^{\Uparrow}$. However, it is not posible because $\langle X, Y\rangle$ is $\vee$-irreducible. □

As a final end point of this study, we may conclude that unlike in the classical framework, not every concept lattice may be linked with a formal context. Thus, lattices number 3 and 5 from Table 3 cannot be associated with a mixed formal context. Both of them have one element which is not an atom but, at the same time, it is a join irreducible element in the lattice. More specifically, there does not exists a mixed concept lattice with three elements.

4 Conclusions

In this work we have presented an algebraic study of a general framework to deal with negative and positive information. After considering new derivation operators we prove that they constitutes a Galois connection. The main results of the work are devoted to establish the new relation among mixed concept lattices and mixed formal concepts. Thus, the most outstanding conclusions are that:

- the inclusion of a new (and different) object in a formal concept has a direct effect in the structure of the lattice, producing a different lattice.
- no any kind of lattice may be associated with a mixed formal context, which induces a restriction in the structure that mixed concept lattice may have.

Acknowledgements

Supported by grant TIN2011-28084 of the Science and Innovation Ministry of Spain, co-funded by the European Regional Development Fund (ERDF).

References

1. R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules in Large Databases. In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, pages 487–499, Santiago de Chile, Chile, 1994. Morgan Kaufmann Publishers Inc.
2. J.F. Boulicaut, A. Bykowski, and B. Jeudy. Towards the tractable discovery of association rules with negations. In *FQAS*, pages 425–434, 2000.
3. B. Ganter. Two basic algorithms in concept analysis. *Technische Hochschule, Darmstadt*, 1984.
4. G. Gasmi, S. Ben Yahia, E. Mephu Nguifo, and S. Bouker. Extraction of association rules based on literalsets. In *DaWaK*, pages 293–302, 2007.
5. J.L. Guigues and V. Duquenne. Familles minimales d implications informatives resultant d un tableau de donnees binaires. *Mathematiques et Sciences Sociales*, 95:5–18, 1986.
6. H. Mannila, H. Toivonen, and A. Inkeri Verkamo. Efficient algorithms for discovering association rules. In *KDD Workshop*, pages 181–192, 1994.
7. R. Missaoui, L. Nourine, and Y. Renaud. Generating positive and negative exact rules using formal concept analysis: Problems and solutions. In *ICFCA*, pages 169–181, 2008.
8. R. Missaoui, L. Nourine, and Y. Renaud. An inference system for exhaustive generation of mixed and purely negative implications from purely positive ones. In *CLA*, pages 271–282, 2010.
9. R. Missaoui, L. Nourine, and Y. Renaud. Computing implications with negation from a formal context. *Fundam. Inform.*, 115(4):357–375, 2012.
10. A. Revenko and S. Kuznetzov. Finding errors in new object intents. In *CLA*, pages 151–162, 2012.
11. J.M. Rodriguez-Jimenez, P. Cordero, M. Enciso, and A. Mora. Negative attributes and implications in formal concept analysis. *Procedia Computer Science*, 31(0):758 – 765, 2014. 2nd International Conference on Information Technology and Quantitative Management, ITQM 2014.
12. R. Wille. Restructuring lattice theory: an approach based on hierarchies of concepts. In *Rival, I. (ed.): Ordered Sets*, pages 445–470. Boston, 1982.

Author Index

Title:	CLA 2014, Proceedings of the Eleventh International Conference on Concept Lattices and Their Applications
Publisher:	Pavol Jozef Šafárik University in Košice
Expert advice:	Library of Pavol Jozef Šafárik University in Košice (http://www.upjs.sk/pracoviska/univerzitna-kniznica)
Year of publication:	2014
Number of copies:	70
Page count:	XII + 280
Authors sheets count:	15
Publication:	First edition
Print:	Equilibria, s.r.o.

ISBN 978–80–8152–159–1