



Appropriating Data from Structured Sources for Formal Concept Analysis

ORPHANIDES, Constantinos

Available from the Sheffield Hallam University Research Archive (SHURA) at:

<http://shura.shu.ac.uk/31513/>

A Sheffield Hallam University thesis

This thesis is protected by copyright which belongs to the author.

The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author.

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.

Please visit <http://shura.shu.ac.uk/31513/> and <http://shura.shu.ac.uk/information.html> for further details about copyright and re-use permissions.

Appropriating Data from Structured Sources for Formal Concept Analysis

Constantinos Orphanides

A thesis submitted in partial fulfilment of the requirements of
Sheffield Hallam University for the degree of
Doctor of Philosophy

September 2022

Abstract

Formal Concept Analysis (FCA) is a principled way of deriving a concept hierarchy from a collection of objects and their associated attributes, building on the mathematical theory of lattices and ordered sets.

To conduct FCA, some appropriation steps need to be taken on a set of data as a prerequisite. Firstly, the data need to be acquired from a data source and decisions need to be made about how the data will be analyzed, such as which objects will be included in the analysis, and how each attribute should be interpreted. They then need to be transformed into a formal context, which can then be visualized as a formal concept lattice.

Transforming a formal context into its constituent formal concepts is a process which is well defined and well understood in literature. The same holds true for converting formal concepts into a formal concept lattice.

On the other hand, the process of appropriating a dataset into a formal context tends to be an ad-hoc and bespoke one. ToscanaJ can produce formal contexts from a relational database, while ConExp can produce simple formal contexts in a manual fashion. In the CUBIST project, Dau developed a semi-automated, scalingless approach to generate formal contexts out of a triple store by concatenating the object-attribute pairs returned from the resulting table into their corresponding formal attributes, while Orphanides developed an approach that also provided scaling capabilities, albeit again relying on triple store data. Cubix, the final prototype of the CUBIST project, incorporated the approaches of Dau and Orphanides in an interactive web frontend.

FcaBedrock is an Open-source software (OSS) developed as part of this study, employing a series of steps to appropriate data for FCA in a semi-automated, user-driven environment. To underpin this work, we take a case study approach, using two case studies in particular: the UCI Machine Learning (ML) Repository—a dataset repository for the empirical analysis of machine learning algorithms—and the e-Mouse Atlas of Gene Expression (EMAGE), a database of anatomical terms for each Theiler Stage (TS) in mouse development.

We compare our approach with existing approaches, using datasets from the two case studies. The appropriation of the datasets become an integral part of our evaluation, providing the real-life context and use-cases of our proposed approach.

The UCI ML Repository and EMAGE case studies revealed how prior to this study, a multitude of existing data sources, types and formats, were either not accessible, or not easily accessible to FCA; the data appropriation processes were in most cases tedious and time-consuming, often requiring the manual creation of formal contexts out of datasets. In other cases, rigid, non-flexible approaches were developed, with hardcoded assumptions made about the underlying use-case they were developed for. This is unlike the software and techniques developed in this study, where the same semi-automated steps can consistently facilitate the appropriation of data for FCA, from the most common data sources and their underlying data types.

The aim of this study was to discover how effective each FCA approach is in appropriating data for FCA, answering the research question: *“How can data from structured sources, consisting of various data types, be acquired and appropriated for FCA in a semi-automated, user-driven environment?”*. FcaBedrock emerged as the best appropriation approach, by abstracting the issue of structured sources away using the ubiquitous CSV (Comma Separated Value) file format as an input source and by providing both automated and semi-automated means of creating meaningful formal contexts from a dataset. Dau’s CUBIST scalingless approach, while semi-automated, was restricted to RDF triple stores and did not provide any flexibility as to how each attribute of the dataset should be interpreted. Orphanides’s CUBIST scaleful approach, while providing more flexibility with its scaling capabilities, was again restricted to RDF triple stores. The CUBIST interactive approach improved upon those ideas, by allowing the user to drive the analysis via a user-friendly web frontend. ToscanaJ was restricted to only using SQL/NoSQL databases as an input source, while both ToscanaJ and ConExp provided no substantial appropriation techniques other than creating a formal context by hand.

Declaration

I hereby declare that:

1. I have not been enrolled for another award of the University, or other academic or professional organization, whilst undertaking my research degree.
2. None of the material contained in the thesis has been used in any other submission for an academic award.
3. I am aware of and understand the University's policy on plagiarism and certify that this thesis is my own work. The use of all published or other sources of material consulted have been properly and fully acknowledged.
4. The work undertaken towards the thesis has been conducted in accordance with the SHU Principles of Integrity in Research and the SHU Research Ethics Policy.

Name: Constantinos Orphanides

Date: Thursday 8th September, 2022

Award: Doctor of Philosophy (Ph.D.)

Institute: Industry and Innovation Research Institute (I²RI)

Director of Studies: Prof. Simon Andrews

Acknowledgements

I would like to thank my Director of Studies, Prof. Simon Andrews, for the invaluable help, guidance, and support he provided throughout this journey. I would also like to thank Dr. Simon Polovina, for his mentorship during my professional and academic tenure at Sheffield Hallam University.

Thank you to my parents Kyriacos and Katia, my siblings George and Ivy, and my close friends for their continuous encouragement, support and understanding.

Finally, I would like to thank my late, beloved grandmother, Evie Flevotoma, for instilling in me the belief that anything is possible, as long as you have the resilience to keep going, even when times get rough. You are dearly missed.

Cogito, ergo sum.

René Descartes, 1637

To my darling Chara, and our daughter Ioanna.

Table of contents

List of figures	xii
List of tables	xv
Glossary	xvi
1 Introduction	1
1.1 Research Rationale	1
1.2 Research Objectives	3
1.3 Thesis Overview	3
2 Research Methodology	5
2.1 Introduction	5
2.2 The Case Study Research Approach	6
2.2.1 Case Study Research Methods	8
2.3 Alternative Approaches	9
2.4 Challenges	9
2.5 Ethical Considerations	10
2.6 Concluding Summary	10
3 Structured Data and Their Data Types	11
3.1 Introduction	11
3.2 Structured Data	11
3.3 Data Types in Structured Data	13
3.3.1 Categorical	13

3.3.2	Boolean	13
3.3.3	Numerical	14
3.3.4	Dates	14
3.3.5	Unknown, missing, or empty values	14
3.3.6	Other	15
3.4	Concluding Summary	15
4	Formal Concept Analysis	16
4.1	Introduction	16
4.2	Formal Contexts	17
4.2.1	Formal Context File Formats	18
4.2.1.1	Burmeister (.cxt)	19
4.2.1.2	FIMI (.dat)	20
4.3	Formal Concepts	22
4.3.1	Computing Formal Concepts	23
4.4	Galois Connections	24
4.5	Concept Lattices	25
4.6	Data Booleanization	27
4.7	Data Discretization	27
4.8	Conceptual Scaling	29
4.8.1	Nominal and Dichotomic Scales	30
4.8.2	Ordinal Scales	30
4.8.3	Interordinal and Biordinal Scales	31
4.8.4	Contranominal Scales	31
4.8.5	An Example of Conceptual Scaling	32
4.9	Concluding Summary	34
5	Approaches for Appropriating Data for FCA	35
5.1	Introduction	35
5.2	The e-Mouse Atlas of Gene Expression	35
5.3	UCI Machine Learning Repository	38

5.3.1	Adult Dataset	39
5.3.2	Agaricus Lepiota Dataset	41
5.4	Existing Approaches	41
5.4.1	ToscanaJ	41
5.4.2	ConExp	45
5.4.3	CUBIST Scalingless Approach	47
5.4.4	CUBIST Scaleful Approach	49
5.4.5	CUBIST Interactive Approach	51
5.5	New Approaches	55
5.5.1	FcaBedrock	55
5.6	Concluding Summary	59
6	Experiments	61
6.1	CUBIST Scalingless Approach	61
6.1.1	Introduction	61
6.1.2	Utilized Applications	61
6.1.3	Experimental Results	62
6.1.4	Summary	65
6.2	CUBIST Scaleful Approach	66
6.2.1	Introduction	66
6.2.2	Utilized Applications	66
6.2.3	Experimental Results	66
6.2.4	Summary	69
6.3	CUBIST Interactive Approach	69
6.3.1	Introduction	69
6.3.2	Utilized Applications	70
6.3.3	Experimental Results	70
6.3.4	Summary	71
6.4	FcaBedrock	73
6.4.1	Introduction	73
6.4.2	Utilized Applications	73

6.4.3	Experimental Results	73
6.4.3.1	Experiment #1	73
6.4.3.2	Experiment #2	75
6.4.3.3	Experiment #3	80
6.4.4	Summary	81
7	Evaluation of the FCA Data Appropriation Approaches	82
7.1	Introduction	82
7.2	Essential Features for Enabling Data Appropriation for FCA . .	83
7.3	An Assessment of the FCA Data Appropriation Approaches . .	84
7.3.1	An Assessment of CUBIST's Scalingless Approach	84
7.3.2	An Assessment of CUBIST's Scaleful Approach	85
7.3.3	An Assessment of CUBIST's Interactive Approach	85
7.3.4	An Assessment of FcaBedrock's Guided Automation Ap- proach	86
7.3.5	Overall Ranking and Feature Overview	87
7.4	Concluding Summary	90
8	Conclusion and Future Work	92
8.1	A Review of the Chapters in this Thesis	92
8.2	Addressing of Research Objectives	93
8.3	Research Knowledge Contribution	95
8.4	Conclusion	96
8.5	Future Work	96
	References	98
	Appendix A Relevant Publications	106

List of figures

4.1	A mini-adult example, adapted from the UCI ML Repository's <i>Adult</i> dataset.	18
4.2	BNF of the Burmeister (.cxt) file format.	19
4.3	A mini-adult Burmeister (.cxt) context file.	20
4.4	BNF of the FIMI (.dat) file format.	21
4.5	A mini-adult FIMI (.dat) context file.	21
4.6	The InClose algorithm, adapted from [4].	24
4.7	A lattice corresponding to the Airlines context (table 4.1). . . .	25
4.8	Equal-width binning.	28
4.9	Equal-frequency binning.	28
4.10	A nominal scale of colours.	30
4.11	A dichotomic scale of numbers.	30
4.12	An ordinal scale of numbers.	30
4.13	An interordinal scale of numbers.	31
4.14	A biordinal scale of numbers.	31
4.15	A contranominal scale of numbers.	32
4.16	A concept lattice of the formal context in table 4.6.	34
5.1	Example image of an experimental result (A) and its associated spatial annotation (B). Adapted from the EMAGE database. . .	36
5.2	The EMAGE ontology, adapted from [28].	37
5.3	Metadata of the <i>Adult</i> dataset of the UCI ML Repository. . . .	38

5.4	The dataset browser of the UCI ML Repository (rotated 90° anticlockwise).	40
5.5	Workflow of a ToscanaJ Conceptual Information System.	43
5.6	Definition of realized scales in Elba.	44
5.7	A lattice of the triglyceride scale of figure 5.6, in Elba.	44
5.8	Using realized scales, defined in Elba, to analyze data in ToscanaJ.	45
5.9	Creating and/or modifying a formal context in ConExp.	46
5.10	Concept counting and visualization in ConExp.	46
5.11	An example of an RDF statement.	47
5.12	Transformation of a SPARQL query result (left) to an unscaled formal context (right) in SPARQL2Context. Adapted from [27].	48
5.13	Transformation of a SPARQL query result to an unscaled formal context in SPARQL2FCA.	50
5.14	Transformation of a SPARQL query result to a scaled formal context in SPARQL2FCA.	51
5.15	The CUBIST architecture.	52
5.16	Using NowaSearch to interactively navigate and filter the data of an ontology in a triple store, in CUBIST.	53
5.17	Conceptual scaling in CUBIST.	54
5.18	Visualizing a formal context as a concept lattice in Cubix.	55
5.19	High level overview of the FcaBedrock process.	56
5.20	Previewing a three column CSV file in FcaBedrock.	58
5.21	Automatically detecting attributes, and defining conceptual attribute scales, in FcaBedrock.	59
5.22	Outputting a formal context, in the Burmeister format, in FcaBedrock.	59
6.1	A SPARQL query of EMAGE data used in SPARQL2FCA. Adapted from [27].	62
6.2	The formal context and concept lattice of the data in table 6.1.	62

6.3	Object-unrestricted variant of the SPARQL query in figure 6.1. Adapted from [27].	64
6.4	The formal context and concept lattice of the object-unrestricted data in table 6.2.	64
6.5	SPARQL query of the number of vacancies posted by advertisers, per job board.	67
6.6	Transformation of a SPARQL query result to an unscaled formal context in SPARQL2FCA.	67
6.7	Object and attribute selection in SPARQL2FCA.	68
6.8	Conceptual scaling capabilities in SPARQL2FCA.	69
6.9	Filtering the EMAGE dataset for tissues where gene Bmp4 is detected, in Cubix.	71
6.10	Conceptual scaling in CUBIST.	72
6.11	Visualizing a formal context as a concept lattice in Cubix. . . .	72
6.12	Autodetecting the metadata of the Agaricus Lepiota dataset, in FcaBedrock.	74
6.13	Ignoring (excluding) the ‘class’ attribute in FcaBedrock.	75
6.14	A mushroom habitat/population lattice.	75
6.15	Restricting attribute values of ‘education’, in FcaBedrock. . . .	76
6.16	A lattice of pay per gender, per higher education, in the Adult dataset.	77
6.17	Automatic creation of equal frequency bins for ‘age’, in FcaBedrock.	78
6.18	Pay by age in the Adult dataset, with progressive scaling.	79
6.19	Pay by age in the Adult dataset, with discrete scaling.	79
6.20	Restricting ‘Gene’ to ‘Bmp4’, in FcaBedrock.	80
6.21	A concept lattice of Bmp4’s moderate or strong detection in tissues.	81
7.1	Converting the feature overview per FCA approach to a formal context, in FcaBedrock.	89

7.2	Converting the feature overview per FCA approach to a concept lattice, in ConExp.	90
-----	----------------------------------------------------------------------------------------------	----

List of tables

3.1	An example of structured data.	12
4.1	A formal context representing destinations of five airlines.	17
4.2	An example of data booleanization.	27
4.3	An example of data discretization.	29
4.4	Gender and age of 8 persons. Adapted from [91].	33
4.5	Gender scale and Age scale of the data in table 4.4.	33
4.6	A formal context of the data in table 4.4.	33
6.1	Genes detected in tissues of Theiler Stage 7. Adapted from [27].	63
6.2	Object-unrestricted variant of the results in table 6.1. Adapted from [27].	65
7.1	Essential features support of each FCA data appropriation ap- proach.	88

Glossary

API	Application Programming Interface
BI	Business Intelligence
BNF	Backus–Naur Form
CIS	Conceptual Information System
ConExp	Concept Explorer
CSV	Comma Separated Values
CTO	Chief Technology Officer
CUBIST	Combining and Uniting Business Intelligence with Semantic Technologies (EU FP7 project)
DSV	Delimiter Separated Values
EMAGE	e-Mouse Atlas of Gene Expression
EMAP	e-Mouse Atlas Project
EU	European Union
FCA	Formal Concept Analysis
FCI	Frequent Closed Itemsets
FIMI	Frequent Itemset Mining Implementation
FinTech	Financial Technology

FP7	Framework Programme 7
ISH	In Situ Hybridization
KISS	Keep It Simple and Straightforward
ML	Machine Learning
OSS	Open-source Software
PoC	Proof of Concept
RDF	Resource Description Framework
SPARQL	SPARQL Protocol and RDF Query Language
SQL	Structured Query Language
ST	Semantic Technologies
TCA	Triadic Concept Analysis
TSV	Tab Separated Values
TS	Theiler Stage
UCI	University of California, Irvine

Chapter 1

Introduction

This chapter introduces the rationale and objectives of this research. In section 1.1, we introduce the research rationale which demonstrates the need to explore novel ways of appropriating data from structured sources for Formal Concept Analysis. Section 1.2 states the research objectives and section 1.3 concludes with an overview of the thesis.

1.1 Research Rationale

It has been shown that Formal Concept Analysis (FCA) can be usefully applied to large sets of data [48, 78, 44] and that FCA has applications in knowledge discovery, data mining, data visualization, software development and the semantic web [73, 24, 10, 13, 14, 11, 12, 66]. Algorithms exist that are capable of processing large formal contexts in reasonable time [53, 4-6]. However, data is rarely in a form immediately suitable or appropriate for FCA tools and applications. Data often reside in their underlying data source, often requiring extraction in the form of flat-file tables of comma separated values (CSV). To carry out FCA, these data must be converted into formal contexts. The existing, typically many-valued, attributes must be converted into formal attributes. This can be an involved and time-consuming task, which usually requires a programming element. The task is, essentially, a process of discretizing and booleanizing data; taking each many-valued attribute and converting it into as

many boolean attributes as it has values. The incorporation of many-valued attributes into FCA is well understood [91, 37] and continuous data can be used for FCA by being conceptually scaled [69], or discretized as disjoint ranges of values [48]. However, there is less work describing how these techniques can be applied in an automated, reproducible way [10].

A process of data conversion and appropriation also leads to the possibility of diverse interpretation of the data. We define data appropriation for FCA as the steps applied on a dataset of a structured source (input) to produce a formal context (output); this can or cannot involve pre-processing steps between the input and the output phases. Different choices of which of the data to convert and how to convert them can, without careful documentation, lead to inconsistent and incomparable analyses [8] and lead to problems in measuring the performance of FCA tools and algorithms [56]. At CSTIW’09, a proposal was made that a tool be developed to automate the process of conversion and address these issues [3].

The aim of this work is to discover approaches by which data can be efficiently appropriated for FCA, using both semi-automated (the user driving the process) and automated approaches (appropriating a dataset with no input from the user). This work therefore uses two indicative case studies to answer the key research question:

“How can data from structured sources, consisting of various data types, be acquired and appropriated for FCA in a semi-automated, user-driven environment?”

The research uses the UCI Machine Learning (ML) Repository¹ [34] and the e-Mouse Atlas of Gene Expression (EMAGE)² database [72] as its case studies. It compares the appropriation approaches of the ToscanaJ³ [20] and ConExp⁴ [94] tools, Dau’s CUBIST scalingless approach [27], Orphanides’s

¹<https://archive.ics.uci.edu/ml/>

²<https://www.emouseatlas.org/emage/>

³<https://sourceforge.net/projects/toscanaj/>

⁴<https://sourceforge.net/projects/conexp/>

CUBIST scaleful approach [28], the CUBIST interactive approach [60] of the CUBIST⁵ EU FP7 (European Union’s Framework Programme 7) research project, and FcaBedrock⁶ [9]. Hence, this research explores existing and new FCA techniques and approaches, that can be used to address the issue of data appropriation for FCA.

1.2 Research Objectives

The objectives of this thesis are the following:

1. To understand FCA data appropriation issues and how they are dealt with.
2. To build on existing FCA data appropriation approaches and develop better, novel appropriation approaches.
3. To develop semi-automated, user-driven, FCA data appropriation techniques.
4. To apply existing and new FCA data appropriation approaches to the UCI ML Repository and EMAGE case studies.
5. To compare and evaluate the usefulness and effectiveness of the different FCA data appropriation approaches.

1.3 Thesis Overview

The thesis is organized into 8 chapters.

Chapter 1 presents the research rationale, the research objectives, and the key research question.

Chapter 2 presents the research methodology used to conduct the research, while also considering alternative approaches and ethical considerations. It

⁵<https://cordis.europa.eu/project/id/257403>

⁶<https://github.com/trashr0x/fcabedrock>

explains the reasoning behind choosing the multiple case study research method as the research methodology, and explains how the EMAGE database and the UCI Machine Learning Repository are used as the two critical case studies for this research.

Chapter 3 provides a brief overview of what structured data are, and the data types usually encountered in such data. The overview is given to familiarize the reader with these concepts, as appropriating data for FCA involves retrieving data from such data sources.

FCA is a crucial background component of the research. For FCA to be carried out, data need to be appropriated (or pre-processed) first, so that they can be converted to formal contexts, and subsequently visualized as concept lattices. Chapter 4 familiarizes the reader with FCA and its key concepts.

Chapter 5 presents the existing and new approaches for appropriating data for FCA. The approaches are demonstrated through examples extracted from the EMAGE database, as well as datasets taken from the UCI Machine Learning Repository. After discussing each approach, the unsuitable approaches are identified, and an explanation is provided on why they will not be further evaluated in this work.

Chapter 6 outlines the empirical experiments carried out for each approach identified as suitable in chapter 5, to assess their suitability for FCA data appropriation. Where possible, each experiment utilizes datasets from the two critical case studies. The experiments carried out are used to evaluate the approaches in chapter 7.

Chapter 7 evaluates the approaches, using DESMET as the research methodology. It identifies an essential feature list that an ideal FCA data appropriation approach should possess, and evaluates the investigated approaches against the essential feature list. The chapter concludes by ranking the proposed approaches against the essential feature list, thus identifying the most suitable FCA data appropriation approach.

Finally, the thesis concludes in chapter 8, which outlines the research knowledge contributions and discusses future work.

Chapter 2

Research Methodology

2.1 Introduction

FCA data appropriation issues were empirically studied using the UCI Machine Learning (ML) Repository and the e-Mouse Atlas of Gene Expression (EMAGE) database. These studies investigated the causes of data appropriation for FCA and how such issues can be dealt with using both existing and new FCA tools and data appropriation techniques. The studies were conducted as part of the research objectives outlined in section 1.2 of this thesis. The research objectives include understanding FCA data appropriation issues and proposing new, novel approaches for addressing data appropriation for FCA. In this chapter, the approaches adopted in these studies are explored and focus is given on how the research methods were applied.

The empirical studies have shown that the existing approaches do not provide an automated, reproducible way of appropriating data for FCA; data is rarely in a form immediately suitable or appropriate for FCA tools and applications. Data often reside in their underlying data source, often requiring extraction in the form of flat-file tables of comma separated values (CSV). For FCA to take place, these data must be converted into formal contexts as a prerequisite. The existing, typically many-valued, attributes must be converted into formal attributes. This can be an involved and time-consuming task which

usually requires a programming element. Even so, creating a formal attribute for each value in the attributes of a dataset will, more often than not, produce meaningless analyses when such formal contexts are analyzed or visualized as a concept lattice [3, 10, 9]. As an example, let us imagine a dataset containing ages, salaries, and professions for a large amount of people. Creating a formal attribute for each distinct age and salary figure in the dataset would yield no meaningful results, whereas having the ability to group ages into demographics of interest and classifying salaries into salary brackets would produce insights of higher value to the analyst. This is why the key research question of this thesis focuses on producing novel approaches for automated, user-driven, data appropriation for FCA.

As per section 1.1, our research question is:

“How can data from structured sources, consisting of various data types, be acquired and appropriated for FCA in a semi-automated, user-driven environment?”

Consequently, we investigate data appropriation techniques for FCA in structured sources. In particular, we focus on two structured data sources: the UCI ML Repository and the EMAGE database, acting as our two case studies in this work.

In the following sections, we explore how these research approaches were applied in this work. Section 2.2 explores the case study research approach and why the UCI ML Repository and the EMAGE database are deemed suitable as our two case studies. Sections 2.3, 2.4 and 2.5 present alternative research approaches that could have been used in this work, the challenges faced, and potential ethical considerations. The chapter concludes in section 2.6.

2.2 The Case Study Research Approach

Case study research focuses on empirically inquiring about contemporary phenomena, i.e. cases set within their real world context, particularly when

phenomena and context might not be clearly evident [95]. By allowing a study to investigate empirical occurrences while maintaining the overall qualities of real-life events, Schell in [79] considers case studies as one of the most flexible research designs.

Case study research, on the other hand, has received criticism for being less valuable, difficult to generalize from and biased by researchers. However, according to Yin [95, pp. 38–39, pp. 136–141] and others [49, 75, 96, 33], the use of appropriate research methodological practices allows for the validation and generalization of a case study’s results. Analytic generalization as an effective technique of generalizing findings from a case study research has been also identified by Yin [95]. The majority of misunderstandings relevant to the case study research approach have been argued as being untrue by Flyvbjerg [33]. The case study research approach is used in this body of work to ascertain how data from structured sources can be acquired and appropriated for FCA in a semi-automated, user-driven environment.

In this work, the UCI Machine Learning (ML) Repository [34] and the e-Mouse Atlas of Gene Expression (EMAGE) database [72] are used as case studies. These two case studies are of critical importance to this work, as they provide the needed data to effectively test the novel FCA approaches presented in this research. Critical cases are defined by Flyvbjerg [33] as being crucial to the general problem. In the process of identifying critical cases, Flyvbjerg [33] recommends that the researcher looks for the cases which “either clearly confirm or irrefutably contradict propositions and hypotheses”. A data or knowledge repository is an example of a critical case, as databases of real-life datasets from various disciplines are where FCA data appropriation issues are most likely to exist; the UCI ML Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms, with over 1000 citations, making it one of the top 100 most cited “papers” in all of computer science [34]. EMAGE is a database of *in situ* gene expression data in the mouse embryo and an accompanying suite of tools to search and analyze the data [72, 7]. EMAGE

played an important role as a critical case study in the European Union’s (EU) Framework Programme 7 (FP7) CUBIST (Combining and Uniting Business Intelligence with Semantic Technologies) project, where Business Intelligence (BI) was combined and united with Semantic Technologies (ST) to provide a web platform with intuitive, FCA-based, visual analytics [28].

In summary, the UCI ML Repository and the EMAGE database form good representatives of ideal and critical cases. They provide the real-life context for the investigation of data appropriation for FCA and the evaluation of existing and new data appropriation approaches for FCA. In this work, they are used as critical case studies to address the research question.

2.2.1 Case Study Research Methods

In this work, the UCI ML Repository and the EMAGE database are used as our structured sources, to answer the research question. This was accomplished by acquiring datasets from these structured sources and appropriating them for FCA using both automated and semi-automated means. In [95], Yin lists multiple data sources, experiments and observations as research methods and sources of evidence in a case study. Case studies can also contain elements stemming from other research methods, such as archival analyses, literature search, and observation [75]. The following research methods were used in our research:

- (i) **Multiple data sources:** A major strength of case studies is the opportunity to use many different sources of evidence, as it allows for investigators to address a broader range of historical, attitudinal, and behavioural issues. Thus, any findings or conclusion in a case study is likely to be much more convincing and accurate if it is based on multiple sources of information [95, pp. 97–99]. In this research, we use two structured data sources, comprising of numerous datasets, as our different sources of evidence.

- (ii) **Literature search:** Other FCA-related publications were consulted to understand how data are currently appropriated for FCA and how they can be appropriated for FCA in a user-driven, semi-automated setting.
- (iii) **Analysis of appropriation of data from structured sources:** Datasets were acquired from the UCI ML Repository and the EMAGE database, appropriated, and converted to formal contexts. It was then demonstrated how meaningful FCA analyses and visualizations can be facilitated when FCA data appropriation takes place in a user-driven, automated setting.

These research methods provided ample evidence about existing FCA data appropriation approaches and how they can be dealt with more effectively. The results were used to evaluate the data appropriation approaches in chapter 7.

2.3 Alternative Approaches

A quantitative or qualitative research design could have been used in this work. Quantitative research has been criticized as not being comprehensive enough, due to excluding the meaning which participants assign to events [18, 77]. Similarly, qualitative research has been criticized for its subjectivity and lack of generalizability [77]. However, the flexibility provided by the case study research approach used in this work, allows for the use of quantitative research to complement and reinforce the results of the qualitative research.

Furthermore, in terms of alternative approaches, artificial datasets could have been used instead of data originating from the two case studies. However, artificial datasets would most likely fall short in demonstrating the issues of FCA data appropriation, that this research aims to identify and improve upon.

2.4 Challenges

A challenge faced in this work was deciding whether each and every individual dataset from the UCI ML Repository and the EMAGE database should be

used in ascertaining and evaluating the usefulness of the FCA data appropriation approaches, developed as part of this thesis. For instance, the UCI ML Repository contains more than 600 datasets. From my experience as a software engineer and CTO (Chief Technology Officer) of a FinTech (Financial Technology) company, the conscious decision was made to use a subset of these datasets instead. The datasets used were chosen based on their appearance in existing FCA literature, as well as their appropriation complexity in terms of the number of objects, attributes, and attribute types they contain. Such datasets are most likely to guarantee that all FCA data appropriation issues are captured and identified, while adhering to the KISS principle (Keep It Simple and Straightforward), a principle commonly used in software and system design [54]. Analyzing all datasets of these structured data sources falls outside of the scope of this work.

2.5 Ethical Considerations

There are no ethical considerations to address in this thesis. The datasets acquired from the UCI ML Repository and the EMAGE database have citation policies which have been duly adhered to in this body of work.

2.6 Concluding Summary

This chapter explained the research methods applied in this thesis. The UCI ML Repository and the EMAGE database are the two case studies investigated for FCA data appropriation issues, through the use of the multiple case study methodology. Multiple data sources, literature search and data analysis were used as the case study research methods. The challenges encountered during this work and research ethics were also described in this chapter.

Chapter 3

Structured Data and Their Data Types

3.1 Introduction

In this chapter, we familiarize the reader with structured data and the data types typically encountered in a structured dataset. As demonstrated in chapter 4, data types form an important part of the FCA data appropriation process, as decisions have to be made about how each data type in a dataset should be appropriated, in order to produce meaningful data for FCA; different choices of which data to convert, and how each data type should be treated, can lead to inconsistent and incomparable analyses [10].

In section 3.2 we define structured data. In section 3.3 we define the data types which are typically encountered in a dataset. A concluding summary is provided in section 3.4.

3.2 Structured Data

Structured data are data that contain an identifiable structure and adhere to a predefined data model or schema. They conform to a tabular format, with relationships defined between the different rows and columns. They are

typically stored in traditional databases with identifiable rows and columns, such as relational databases [2]. They can be also stored in other sources such as spreadsheets and Delimiter Separated Value (DSV) files [70] such as CSV or TSV (comma separated and tab separated, respectively). CSV and TSV tabulate data using plain-text format, with a delimiting symbol—such as a comma, or tab—to distinguish between data entries. Unlike traditional databases, CSV and TSV are structured only on a syntactical level, and have no explicitly associated schema information.

An example of structured data is given in table 3.1, where each row represents a person, and each column represents an attribute of that person, such as their name and date of birth.

Id	First Name	Last Name	Date of Birth	Salary	Active
1	John	Doe	27/10/1986	42,000	Y
2	Jane	Doe	05/12/1994	27,500	Y
3	Homer	Simpson	12/05/1956	54,750	N
4	Richard	Roe	19/03/1958		Y
5	Margaret	Roe	20/07/1974	36,000	Y

Table 3.1: An example of structured data.

Querying and manipulating structured data, directly from their underlying structured data source, is possible using a programming element. In traditional databases, this typically involves using Structured Query Language (SQL), a programming language developed by IBM in the early 1970s for managing structured data. However, this requires an understanding of database programming languages and direct access to the database itself. When publicly available, structured data are typically programmatically accessible through an Application Programming Interface (API), or an online web portal or repository that usually provides an interactive query browser to query the underlying data source. Such repositories also provide exporting capabilities in user-readable formats such as CSV files and Excel sheets. For example, the EMAGE database is accessible via an online query browser with search capabilities, and allows the user to export the results as CSV files. On the other hand, the UCI Machine

Learning (ML) Repository does not allow querying of its data and only provides datasets as downloadable CSV files.

More important than where structured data reside and how they are presented to the user, are the actual data types that structured data consist of. For example, in table 3.1, the ‘Id’ column is a numerical type (integer in particular), while the ‘Date of Birth’ column is a date type. Distinguishing between data types form an important part of the FCA data appropriation process, as shall be evidenced later on in this thesis. We outline the data types most frequently encountered in structured sources in section 3.3.

3.3 Data Types in Structured Data

3.3.1 Categorical

Categorical data are *many-valued* attributes, referred to as *categorical* attributes in several datasets of the UCI ML Repository [34], and as *nominal* attributes in the field of Data Mining [82]. Simon [80] defines categorical data as “data that consist of a small number of values, each corresponding to a specific category value”, a definition more related to the field of Statistics; Machine Learning practitioners refer to this process as *one-hot encoding* [41]. An example would be the marital status of a person; one can be single, married, divorced, widowed, or of unknown marital status. If ‘marital status’ is considered as an attribute, then it would be classified as categorical, as it has five possible categories (values).

3.3.2 Boolean

Boolean data are based on a symbolic logic theory developed and conceived by mathematician George Boole in 1854 [23]. Boolean data, logic and operators have become an essential part of modern computers, given the fact that computers are based on the binary system where 1 means *ON* and 0 means *OFF* [59]. Boolean attributes can only have two values representing true or false,

such as the ‘Active’ column in table 3.1. They are also considered a special case of categorical attributes [82], as they are essentially a categorical attribute where only two possible values are allowed.

3.3.3 Numerical

Numerical, also known as *continuous* data, are data that have real numbers (i.e. the union of rational and irrational numbers) as values, such as the ‘Salary’ column in table 3.1. Tan et al [82] explain that “practically, real values can only be measured and represented using a finite number of digits”, and that such types are “typically represented as floating-point variables”.

3.3.4 Dates

Dates are data types which store an instant in time, such as the ‘Date of Birth’ column in table 3.1, expressed as a calendar date and optionally a time of day. The combination of both is usually referred to as ‘datetime’. A date instance is, therefore, a subset of datetime, in that it simply does not define the time portion; the hours, minutes, seconds, and milliseconds are effectively set to zero. While technically distinct, we consider dates and datetimes to be the same in this thesis.

3.3.5 Unknown, missing, or empty values

Datasets often contain missing values. Encountering a missing value in a dataset is usually interpreted as false, although missing might also mean it is unknown whether an object features that particular value [91]. An example of a missing value can be seen in the fourth row of table 3.1, where the salary of Richard Roe is either undisclosed, not known, or missing.

3.3.6 Other

While more data types certainly exist, they are usually sub-types of the types listed above. For example, integers and decimals are both numerical data types.

3.4 Concluding Summary

This chapter explained what structured data are, how they are typically accessed and how they are queried or extracted from their data source. It outlined the various data types usually encountered in a dataset and noted how they form an important part of the FCA data appropriation process. By not distinguishing between data types and guidelines on how to convert them, the appropriation process can lead to diverse interpretations of data, thus facilitating inconsistent and incomparable analyses taking place.

Chapter 4

Formal Concept Analysis

4.1 Introduction

In this chapter we discuss Formal Concept Analysis (FCA) and its constituent components. We define formal contexts in section 4.2 and explain how formal concepts are computed from formal contexts in section 4.3. Sections 4.4 and 4.5 explain how the Galois connections between the formal concepts of a formal context are used to produce concept lattices. Sections 4.6, 4.7 and 4.8 explain how data booleanization, data discretization and conceptual scaling are used in FCA to portray the conceptual structure and hierarchy that exist in data. The chapter concludes in section 4.9.

Formal Concept Analysis (FCA) was introduced in the 1990s by Rudolf Wille and Bernhard Ganter [37], building on applied lattice and order theory developed by Birkhoff and others in the 1930s [22]. It was initially developed as a subsection of Applied Mathematics based on the mathematization of concepts and concepts hierarchy, where a concept is constituted by its *extension*, comprising of all objects which belong to the concept, and its *intension*, comprising of all attributes (properties, meanings) which apply to all objects of the extension [90]. The set of objects and attributes, together with their relation to each other, form a *formal context*, which can be represented by a cross table such as the one in table 4.1.

Table 4.1: A formal context representing destinations of five airlines.

Airlines	Latin America	Europe	Canada	Asia Pacific	Middle East	Africa	Mexico	Caribbean	USA
Air Canada	×	×	×	×	×		×	×	×
Air New Zealand		×		×					×
Nippon Airways		×		×					×
Ansett Australia				×					
Austrian Airlines		×	×	×	×	×			×

4.2 Formal Contexts

The cross table in table 4.1 shows a formal context representing destinations for five airlines. The elements on the left side are formal objects; the elements at the top are formal attributes. If an object has a specific property (formal attribute), it is indicated by placing a cross in the corresponding cell of the table. An empty cell indicates that the corresponding object does not have the corresponding attribute. In the Airlines context above, Air Canada flies to Latin America (since the corresponding cell contains a cross), but does not fly to Africa (since the corresponding cell is empty). However, an empty cell might also mean that it is unknown whether the corresponding object has the corresponding attribute [91].

In mathematical terms, a formal context is defined as a triple $\mathbb{K} := (G, M, I)$, with G being a set of objects, M a set of attributes and I a relation defined between G and M . The relation I is understood to be a subset of the cross product between the sets it relates, so $I \subseteq G \times M$. If an object g has an attribute m , then $g \in G$ relates to $m \in M$ by I , so we write $(g, m) \in I$, or gIm , which is read as: the object g has attribute m . The elements of G and M are called *formal objects* and *formal attributes*, respectively. For an arbitrary

subset of objects $A \subseteq G$, a derivation operator is defined to obtain the set of attributes, common to the objects in A , as follows:

$$A \mapsto A^I = \{m \in M \mid \forall g \in A : gIm\}$$

Similarly, for an arbitrary subset of attributes $B \subseteq M$, a derivation operator is defined to obtain the set of objects, common to the attributes in B , as follows:

$$B \mapsto B^I = \{g \in G \mid \forall m \in B : gIm\}$$

4.2.1 Formal Context File Formats

The most popular formal context file formats for FCA are the Burmeister (.cxt file extension) and FIMI (.dat file extension). The two file formats are presented below, along with their Backus–Naur Form (BNF) specification, which is a metasyntax notation for context-free grammars, often used to describe the syntax of languages such as programming languages or file formats. For more information on BNF we refer the reader to Backus [16] and Knuth [52].

To demonstrate the formal context file formats, we use an example adapted from the UCI ML Repository’s *Adult*¹ data set (figure 4.1). It contains just eight instances and five attributes: *age*, *education*, *employment*, *sex* and *US-citizen*. There are two classes indicating a salary above or below \$50k.

```

39, Bachelors, Clerical, Male, Yes,<=50K
50, Bachelors, Managerial, Female, Yes,<=50K
38, HS-grad, Unskilled, Male, Yes,<=50K
53, 11th, Unskilled, Male, Yes, <=50K
28, Bachelors, Professional, Female, Yes,>50K
37, Masters, Managerial, Female, No,<=50K
49, ?, Clerical, Female, No,<=50K
52, HS-grad, Managerial, Male, Yes,>50K

```

Figure 4.1: A mini-adult example, adapted from the UCI ML Repository’s *Adult* dataset.

¹<https://archive.ics.uci.edu/ml/datasets/adult>

4.2.1.1 Burmeister (.cxt)

The Burmeister (.cxt) file format is a popular format for FCA tools. It originally appeared as a supported file format in Burmeister's ConImp software [25]. It is structured in the following format:

1. A line containing the character "B".
2. An empty line.
3. A line with the number of objects.
4. A line with the number of attributes.
5. An empty line.
6. A list of all formal objects, each on a separate line.
7. A list of all formal attributes, each on a separate line.
8. The body of the formal context as a grid, where crosses are used to indicate true values and dots are used to indicate false values.

A BNF metasyntax notation of the Burmeister format is shown in figure 4.2.

```

1  <Burmeister> ::= "B" <NewLine>
2                      <NewLine>
3                      <NumberOfFormalObjects> <NewLine>
4                      <NumberOfFormalAttributes> <NewLine>
5                      <NewLine>
6                      <FormalObjects> <NewLine>
7                      <FormalAttributes> <NewLine>
8                      <FormalContext>
9
10 <NumberOfFormalObjects> ::= [0-9]+
11 <NumberOfFormalAttributes> ::= [0-9]+
12 <FormalObjects> ::= (<Value> <NewLine>)* <Value>
13 <FormalAttributes> ::= (<Value> <NewLine>)* <Value>
14 <FormalContext> ::= (("X" | ".")+ <NewLine>)*
15                  ("X" | ".")
16 <Value> ::= [a-z]+
17 <NewLine> ::= "\r\n"
```

Figure 4.2: BNF of the Burmeister (.cxt) file format.

The Burmeister formal context of the mini-adult example, in figure 4.1, can be seen in figure 4.3.

```

B
8
15

0
1
2
3
4
5
6
7
age-<30
age-30to<40
age-40to<50
age->=50
education-Bachelors
education-Masters
education-11th
education-HS-grad
employment-Clerical
employment-Managerial
employment-Professional
employment-Unskilled
sex-Male
sex-Female
US-citizen
.X..X...X...X.X
...XX....X...XX
.X.....X...XX.X
...X..X....XX.X
X...X.....X..XX
.X...X...X...X.
..X.....X....X.
...X...X.X..X.X

```

Figure 4.3: A mini-adult Burmeister (.cxt) context file.

4.2.1.2 FIMI (.dat)

FIMI (.dat) is a less popular FCA file format, originating from the Frequent Itemset Mining Implementation (FIMI) data mining community [38]. The

FIMI community has developed multiple algorithms for generating the so-called *Frequent Closed Itemsets* (FCI) [40], which are precisely concept intents corresponding to large concept extents. As opposed to Burmeister, a FIMI file only consists of rows of numbers; each row represents a formal object and each number represents a formal attribute index. The ordering of the attributes is as one would expect from the formal context, taking the first column of the context to be attribute one, and so on.

It is structured in the following format:

1. A list of all formal objects, each on a separate line. Each line contains the indexes of the formal attributes featured by the formal object.
2. In each line, formal attribute indexes are separated by a white space.

A BNF metasyntax notation of the FIMI format is shown in figure 4.4.

```

1 <FIMI> ::= <RowOfFormalAttributes>*
2
3 <RowOfFormalAttributes> ::= (<FormalAttributeIndex> <WhiteSpace>)*
4                               <FormalAttributeIndex> <NewLine>
5
6 <FormalAttributeIndex> ::= [0-9]+
7 <WhiteSpace>           ::= " "
8 <NewLine>              ::= "\r\n"
```

Figure 4.4: BNF of the FIMI (.dat) file format.

The FIMI formal context of the mini-adult example, in figure 4.1, can be seen in figure 4.5.

```

2 5 9 13 15
4 5 10 14 15
2 8 12 13 15
4 7 12 13 15
1 5 11 14 15
2 6 10 14
3 9 14
4 8 10 13 15
```

Figure 4.5: A mini-adult FIMI (.dat) context file.

4.3 Formal Concepts

In a given formal context $\mathbb{K} := (G, M, I)$, a pair (A, B) is a *formal concept* if and only if $A \subseteq G$, $B \subseteq M$, $A^I = B$ and $B^I = A$. The set A is called the *extension* of the formal concept and the set B is called the *intension* of the formal concept. A formal concept is, therefore, a closed set of object-attribute relations, in that its extension contains all objects that have the attributes in its intension, and the intension contains all attributes shared by the objects in its extension [89]. For example, in the Airlines context (table 4.1), it can be seen from the cross-table that Air Canada and Austrian Airlines fly to both the USA and Europe. However, this does not constitute a formal concept because both airlines also fly to Asia Pacific, Canada and the Middle East. Adding these destinations completes (closes) the formal concept, because one can neither enlarge the attribute set, nor the object set:

$$(\{Air\ Canada, Austrian\ Airlines\}, \{Europe, Canada, Asia\ Pacific, Middle\ East, USA\}).$$

Concepts can only live in relationships with many other concepts where the *subconcept-superconcept-relation* plays a prominent role. Being a subconcept of a superconcept means that the extension of the subconcept is contained in the extension of the superconcept, which is equivalent to the relationship that the intension of the subconcept contains the intension of the superconcept [86, p. 201]. The subconcept-superconcept-relation is mathematized by:

$$(A_1, B_1) \leq (A_2, B_2) : \Longleftrightarrow A_1 \subseteq A_2 \ (\Longleftrightarrow B_1 \supseteq B_2)$$

Thus, this partial order of formal concepts models their natural hierarchy, where more general concepts have more inclusive extents and less inclusive intents,

and more specific concepts have less inclusive extents and more inclusive intents [21].

The set of all formal concepts of a formal context $\mathbb{K} := (G, M, I)$ is denoted by $\mathfrak{B}(G, M, I)$, or $\mathfrak{B}(\mathbb{K})$.

4.3.1 Computing Formal Concepts

Formal contexts of even modest size can generate an exponentially large number of formal concepts. In a formal context $\mathbb{K} := (G, M, I)$, there can be a maximum of $2^{\min(|G|, |M|)}$ (also denoted as $2^{|G| \wedge |M|}$) formal concepts.

When computing formal concepts programmatically, boolean matrices are usually used to represent the formal context [4, 5]. Similarly to formal contexts, the rows and columns of the boolean matrices are computationally interchangeable; the same will apply if ‘A’, ‘extent’, ‘object’ and ‘row’ are substituted for ‘B’, ‘intent’, ‘attribute’ and ‘column’ (and vice-versa).

There are several algorithms for computing the formal concepts in a formal context – reviews of such algorithms, that take a variety of approaches, can be found in [6, 15, 97, 56].

An example of such an algorithm is given in figure 4.6, implemented in the formal concept miner In-Close². It is conceptually based on Kuznetsov’s *Close-By-One* algorithm [55]. In Close-By-One, I is a boolean matrix representing a formal context with m rows (representing a set of objects) and n columns (representing a set of attributes). A formal concept is represented by an extent $A[r]$ (an ordered list of objects) and an intent $B[r]$ (an ordered list of attributes), where r is the index of the concept. j is the current attribute, r is the index of the concept being currently closed, r_{new} is a global index of the candidate new concept and y is a starting column. InClose(r, y) means ‘incrementally close concept r , beginning at attribute y ’. IsCanonical utilizes canonicity (lexicographical order) to examine if a concept is new, thus circumventing the

²<https://sourceforge.net/projects/inclose/>

need to explicitly search for repeated results [36, 4]. For a more thorough and technical explanation of how the algorithm works, refer to [4, 5].

```

1: function INCLOSE( $r, y$ )
2:    $r_{new} \leftarrow r_{new} + 1$ ;
3:   for  $j \leftarrow y$  upto  $n - 1$  do
4:      $A[r_{new}] \leftarrow 0$ ;
5:     foreach  $i \in A[r]$  do
6:       if  $I[i][j]$  then
7:          $A[r_{new}] \leftarrow A[r_{new}] \cup \{i\}$ ;
8:       end if
9:     end for
10:    if  $|A[r_{new}]| > 0$  then
11:      if  $|A[r_{new}]| = |A[r]|$  then
12:         $B[r] \leftarrow B[r] \cup \{j\}$ ;
13:      else
14:        if  $IsCanonical(r, j - 1)$  then
15:           $B[r_{new}] \leftarrow B[r] \cup \{j\}$ ;
16:           $InClose(r_{new}, j + 1)$ ;
17:        end if
18:      end if
19:    end if
20:  end for
21: end function

```

Figure 4.6: The InClose algorithm, adapted from [4].

4.4 Galois Connections

Another central notion of FCA is a duality called a ‘Galois connection’, which is often observed between items that relate to each other in a given domain, such as objects and attributes. A Galois connection implies that “if one makes the sets of one type larger, they correspond to smaller sets of the other type, and vice versa” [69]. For example, taking the concept that we formed previously by inspecting the cross-table in table 4.1:

$$(\{Air\ Canada, Austrian\ Airlines\}, \{Europe, Canada, Asia\ Pacific, Middle\ East, USA\})$$

If Africa is added to the list of destinations, the set of airlines reduces to $\{\textit{Austrian Airlines}\}$. Similarly, if Nippon Airways is added to the list of airlines, the list of destinations reduces to $\{\textit{Europe}, \textit{Asia Pacific}, \textit{USA}\}$.

4.5 Concept Lattices

The Galois connections between the formal concepts of a formal context can be visualized in a *concept lattice* (figure 4.7), which is an intuitive way of discovering hitherto undiscovered information in data and portraying the natural hierarchy of concepts that exist in a formal context.

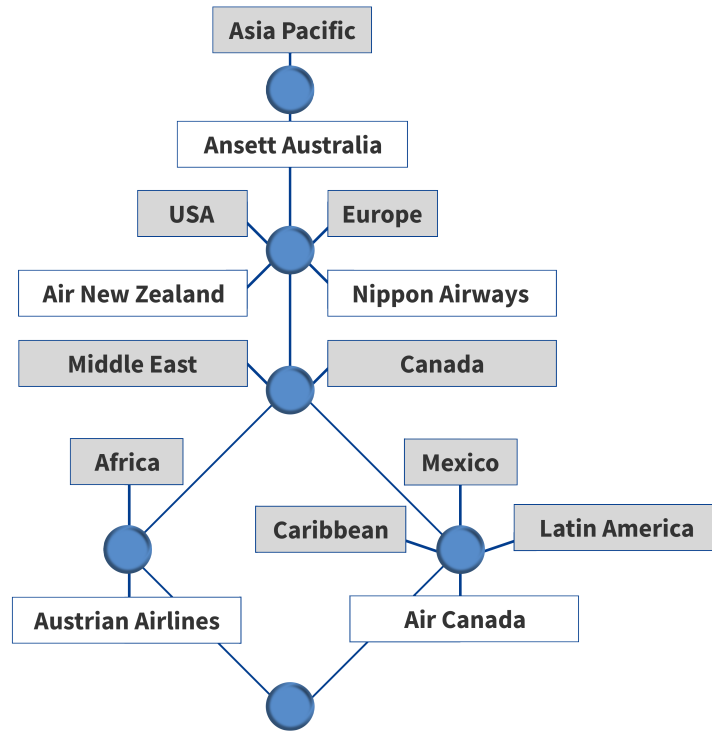


Figure 4.7: A lattice corresponding to the Airlines context (table 4.1).

A concept lattice consists of the set of formal concepts of a formal context $\mathbb{K} := (G, M, I)$ and the subconcept-superconcept-relation between the concepts. It is denoted by $\mathfrak{B}(G, M, I)$, or $\mathfrak{B}(\mathbb{K})$, and is called the *concept lattice of \mathbb{K}* . The nodes in figure 4.7 represent formal concepts. It is conventional that formal

objects are noted slightly below and formal attributes slightly above the nodes, which they label.

A concept lattice can provide valuable information when one knows how to read it. As an example, the node which is labelled with the formal attribute ‘Asia Pacific’ shall be referred to as *Concept A*. To retrieve the extension of Concept A (the objects which feature the attribute ‘Asia Pacific’), one begins at the node where the attribute is labelled and traces all paths which lead down from the node. Any objects one meets along the way are the objects which have that particular attribute. Looking at the lattice in figure 4.7, if one takes the attribute ‘Asia Pacific’ and traces all paths which lead down from the node, one will collect all the objects. Thus Concept A can be interpreted as ‘All airlines fly to Asia Pacific’.

Similarly, the node which is labelled with the formal object ‘Air New Zealand’ shall be referred to as *Concept B*. To retrieve the intension of Concept B (the attributes of ‘Air New Zealand’), one begins at the node where the object is labelled and traces all paths which lead up from the node. Any attributes one meets along the way, are the attributes of that particular object. Looking at the lattice once again, if one takes the object ‘Air New Zealand’ and traces all paths which lead up from the node, one will collect the attributes ‘USA’, ‘Europe’, and ‘Asia Pacific’. This can be interpreted as ‘The Air New Zealand airline flies to the USA, Europe and Asia Pacific’. The concept that we formed previously by inspecting the cross-table in 4.1,

$$(\{Air\ Canada, Austrian\ Airlines\}, \{Europe, Canada, Asia\ Pacific, Middle\ East, USA\})$$

is the node in the centre of the lattice; the one labelled with ‘Middle East’ and ‘Canada’. It becomes quite clear, for example, that although Air New Zealand and Nippon Airways also fly to Europe, the USA and Asia Pacific, only Air Canada and Austrian Airlines fly to Canada and the Middle East as well.

Although the airline context is a small example of FCA, visualizing the formal context clearly shows that concept lattices provide richer information

than from looking at the cross-table alone. This type of hierarchical intelligence that is gleaned from FCA is not so readily available from other forms of data analysis.

4.6 Data Booleanization

Data booleanization is the process of taking each many-valued attribute in a dataset and converting it into as many boolean attributes as it has values [45]. It is the approach used to convert categorical attributes in FCA, where a formal context attribute is created for each of the values of the attribute [3].

An example of data booleanization is shown in table 4.2, where the categorical *Colour* attribute has been booleanized to as many attributes as it has values: *Red* and *Blue*. As roses are specified as red in the dataset on the left, roses will have their red attribute set to true and their blue attribute set to false in the booleanized dataset on the right.

<i>Initial</i>		<i>Booleanized</i>		
Flower	Colour	Flower	Red	Blue
Roses	Red	Roses	T	F
Violets	Blue	Violets	F	T

Table 4.2: An example of data booleanization.

4.7 Data Discretization

Data discretization is defined as “a process of converting continuous data attribute values into a finite set of intervals with minimal loss of information” [47]. In simpler terms, ranges are used to scale continuous values [9]. In [47], Jin et al explain how a plethora of data mining tasks usually involve dealing with continuous attributes, and how continuous attributes decrease its performance. This is resolved by producing discretized values of a continuous attribute and replacing it with the new values. The ranges created when discretizing a

continuous attribute are also called *bins*. In the data binning process, the original data values which fall in a given small interval (the so-called bin) are replaced by a value representative of that interval [62].

Several discretization/binning methods exist [31, 30, 92]. Some well-known binning methods are *equal-width binning* and *equal-frequency binning* [26]. Consider a numerical attribute X_i , sorted in ascending order (smaller first), with v_{min} and v_{max} being the minimum and maximum values respectively. In equal-width binning (figure 4.8), data are divided into k intervals with the intervals having width $w = (v_{max} - v_{min})/k$. In equal-frequency binning (figure 4.9), the values are divided into k intervals so that each interval contains approximately the same number of values. Thus, each interval contains v_{count}/k adjacent values.

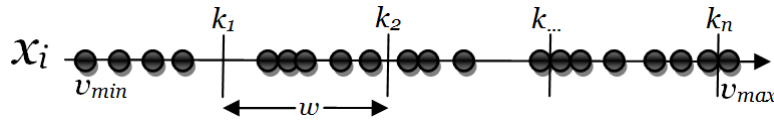


Figure 4.8: Equal-width binning.

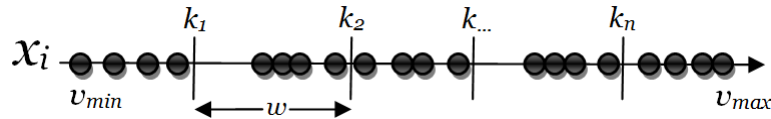


Figure 4.9: Equal-frequency binning.

An example of data discretization is shown in table 4.3, where final year project grades were classified under ranges rather than individually. The ranges were created manually, representing a grading structure akin to the British undergraduate degree classification system [84]. In a larger dataset, this would reduce the number of attributes significantly, and would make questions such as “*how many people achieved first class honours on their final year project?*” easier to answer.

<i>Initial</i>		<i>Discretized</i>				
Name	Grade	Name	>70%	60–69%	50–59%	<50%
John D.	65%	John D.	F	T	F	F
Jane D.	79%	Jane D.	T	F	F	F
Simon D.	53%	Simon D.	F	F	T	F
Mark R.	61%	Mark R.	F	T	F	F
Jill R.	71%	Jill R.	T	F	F	F

Table 4.3: An example of data discretization.

4.8 Conceptual Scaling

More often than not, datasets contain attributes which allow a range of values, rather than a single value. This can be modeled by a *many-valued context* [81].

In mathematical terms, a many-valued context is a quadruple (G, M, W, I) where I is a ternary relation between those three sets, i.e. $I \subseteq G \times M \times W$ such that $(g, m, w_1), (g, m, w_2) \in I$ always implies $w_1 = w_2$ [90, p. 12]. The elements of G , M , and W are called *objects*, *attributes* and *attribute values*, respectively. $(g, m, w) \in I$ is read as: the object g has the attribute value w for the attribute m , and can be simply written as $m(g) = w$ [37, 43].

Many-valued contexts can be transformed into formal contexts using *conceptual scaling*, where every many-valued attribute is transformed into a *scale*, each scale being a formal (single-valued) context itself. To achieve this, we assign to each many-valued attribute $m \in M$ a conceptual scale \mathbb{S}_m , which is a formal context $\mathbb{S}_m := (G_m, M_m, I_m)$ with $m(G) = \{m(g) \mid \forall g \in G\} \subseteq G_m$. The choice of these scales is a matter of interpretation; \mathbb{S}_m should ideally be selected in such a way so that it reflects the implicitly given structure of the attribute values, as well as the issues pertaining to the analysis of the dataset [68].

Some examples of conceptual scales are briefly presented in sections 4.8.1, 4.8.2, 4.8.3, and 4.8.4. For further definitions and examples of conceptual scales, refer to Ganter and Wille’s book “Formal Concept Analysis: Mathematical Foundations” [37].

4.8.1 Nominal and Dichotomic Scales

A nominal scale is defined by the context $\mathbb{N}_n := (\mathbf{n}, \mathbf{n}, =)$. It can be used to scale attributes, the values of which mutually exclude each other, e.g. a categorical attribute of colours (figure 4.10).

=	Red	Green	Blue
Red	×		
Green		×	
Blue			×

Figure 4.10: A nominal scale of colours.

A special case of nominal scaling is the dichotomic scale, defined by the context $\mathbb{D} := (\{\mathbf{0}, \mathbf{1}\}, \{\mathbf{0}, \mathbf{1}\}, =)$. It is isomorphic to the scales \mathbb{N}_2 and $\mathbb{M}_{1,1}$ (see section 4.8.3). It can be used to scale boolean attributes with values of the kind $\{yes, no\}$, $\{true, false\}$, and so on (figure 4.11).

	0	1
0	×	
1		×

Figure 4.11: A dichotomic scale of numbers.

4.8.2 Ordinal Scales

An ordinal scale is defined by the context $\mathbb{O}_n := (\mathbf{n}, \mathbf{n}, \leq)$. It can be used to scale attributes, the values of which are ordered, and where each value implies a weaker one. For example, $1 \leq 3$ implies that $1 \leq 2$, and so on (figure 4.12).

	≤ 1	≤ 2	≤ 3	≤ 4
1	×	×	×	×
2		×	×	×
3			×	×

Figure 4.12: An ordinal scale of numbers.

4.8.3 Interordinal and Biordinal Scales

An interordinal scale is defined by the context $\mathbb{I}_n := (\mathbf{n}, \mathbf{n}, \leq) \mid (\mathbf{n}, \mathbf{n}, \geq)$. It is an alternative to ordinal scaling, and can be used, for example, for questionnaire-like type of attributes comprising of so-called *polar* values, like “rather agree” and “rather disagree” (figure 4.13).

	≤ 1	≤ 2	≤ 3	≤ 4	≥ 1	≥ 2	≥ 3	≥ 4
1	×	×	×	×	×			
2		×	×	×	×	×		
3			×	×	×	×	×	
4				×	×	×	×	×

Figure 4.13: An interordinal scale of numbers.

Attributes with polar values often lend themselves to biordinal scaling [37]. A biordinal scale is defined by the context $\mathbb{M}_{n,m} := (\mathbf{n}, \mathbf{n}, \leq) \dot{\cup} (\mathbf{m}, \mathbf{m}, \geq)$. Each object is assigned one of the two poles, allowing graduations (figure 4.14). For example, the values $\{\textit{strongly disagree}, \textit{disagree}, \textit{agree}, \textit{strongly agree}\}$ suggest the following scaling: *agree* and *disagree* mutually exclude each other, *strongly agree* implies *agree*, and *strongly disagree* implies *disagree*.

	≤ 1	≤ 2	≤ 3	≤ 4	≥ 5	≥ 6
1	×	×	×	×		
2		×	×	×		
3			×	×		
4				×		
5					×	
6					×	×

Figure 4.14: A biordinal scale of numbers.

4.8.4 Contranominal Scales

A contranominal scale is defined by the context $\mathbb{C}_n := (\mathbf{n}, \mathbf{n}, \neq)$ (figure 4.15). It is a scale not typically encountered in real datasets. It has, however, important theoretical meaning, due to the fact that it gives rise to 2^n formal concepts, which is the maximum number of formal concepts a formal context can generate

(refer to section 4.3.1 for a reminder). This makes it an ideal scale for testing the efficiency of formal concept mining algorithms and concept lattice visualization algorithms [43].

\neq	1	2	3	4
1		×	×	×
2	×		×	×
3	×	×		×
4	×	×	×	

Figure 4.15: A contranominal scale of numbers.

4.8.5 An Example of Conceptual Scaling

In [91], Wolff gives an example of a table containing the gender and age of eight persons (table 4.4). In order to convert this many-valued context into a formal context, two scales (contexts) were created; the first scale represented their gender, containing ‘m’ for male and ‘f’ for female as possible values (the equivalent of a nominal or dichotomic scale). The second scale represented their age, but instead of having the actual ages for each person, five formal attributes (the so called *scale attributes*) were produced; ‘<18’, ‘<40’, ‘<=65’, ‘>65’ and ‘>=80’ (the equivalent of a biordinal scale). Both scales can be seen at table 4.5. Adam is 21 years old, so the object has the ‘<40’ and ‘<=65’ formal attributes, as the object is both younger than 40 and younger than 65 (but not younger than 18, nor older than 65 or 80). George is 90 years old, so the object has the ‘>65’ and ‘>=80’ formal attributes, as the object is both older than 65 and 80 (but not younger than 18, 40 or 65). The two scales were then merged to form the complete formal context representing ‘the age and gender of eight persons’ (table 4.6). Its corresponding concept lattice can be seen in figure 4.16, where it can be easily seen how Eva, being the youngest of them all, is younger than 18, and hence also younger than 40 and 65.

	Gender	Age
Adam	M	21
Betty	F	50
Chris	?	66
Dora	F	88
Eva	F	17
Fred	M	?
George	M	90
Harry	M	50

Table 4.4: Gender and age of 8 persons. Adapted from [91].

Gender scale

	m	f
m	×	
f		×
?		

Age scale

	<18	<40	≤65	>65	≥80
17	×	×	×		
21		×	×		
50			×		
66				×	
88				×	×
90				×	×
?					

Table 4.5: Gender scale and Age scale of the data in table 4.4.

Persons	male	female	age <18	age <40	age ≤65	age >65	age ≥80
Adam	×			×	×		
Betty		×			×		
Chris						×	
Dora		×				×	×
Eva		×	×	×	×		
Fred	×						
George	×					×	×
Harry	×				×		

Table 4.6: A formal context of the data in table 4.4.

Therefore, conceptual scaling not only facilitates the conversion of many-valued contexts into formal contexts, but also provides structured conceptual

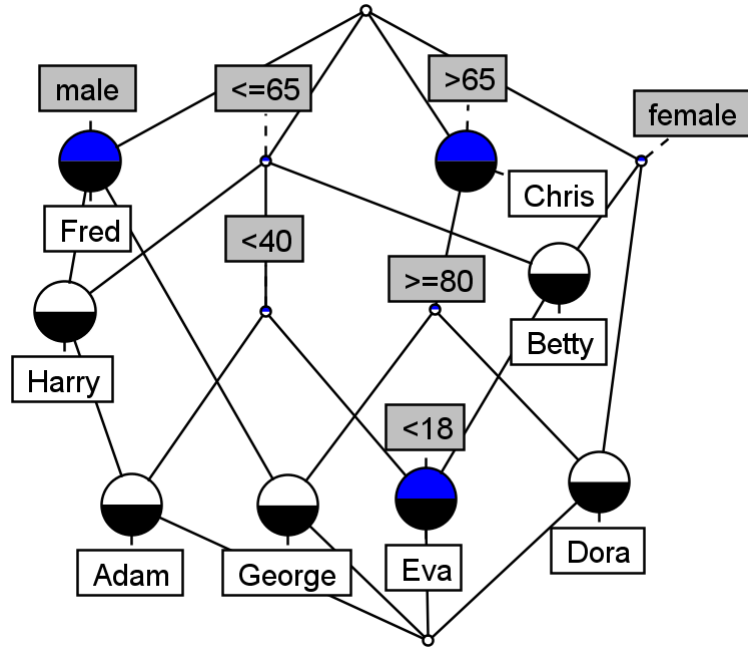


Figure 4.16: A concept lattice of the formal context in table 4.6.

meta-information about the data and allows for the visualization of the resulting conceptual hierarchy [81].

4.9 Concluding Summary

In this chapter we explained what Formal Concept Analysis is and demonstrated the process of taking a set of data, converting it into a formal context, and visualizing it as a concept lattice. Through examples, we demonstrated how data booleanization, data discretization and conceptual scaling allow for the generation of formal contexts to conduct meaningful FCA analyses and intuitive visualizations, by portraying the conceptual structure and hierarchies that exist in data. These notions are essential to the FCA data appropriation process, as shall be discussed in chapter 5.

Chapter 5

Approaches for Appropriating Data for FCA

5.1 Introduction

This chapter discusses the existing and new approaches developed for appropriating data for Formal Concept Analysis (FCA). Prior to this discussion, we outline the structured data sources which are used as our two critical case studies, and examples of the underlying data they contain: the e-Mouse Atlas of Gene Expression (EMAGE) database and the UCI Machine Learning (ML) Repository. This is to familiarize the reader with the types of datasets that will be used in this chapter, and subsequent ones.

Sections 5.2 and 5.3 discuss EMAGE and the UCI ML Repository, respectively. Sections 5.4 and 5.5 present the existing and new FCA data appropriation approaches. Finally, the chapter concludes in section 5.6.

5.2 The e-Mouse Atlas of Gene Expression

The e-Mouse Atlas of Gene Expression (EMAGE) is an online, biological resource which contains in situ gene expression information at all states of mouse development [85, 71, 72].

5.2 The e-Mouse Atlas of Gene Expression

A gene is a unit of instructions that provides directions to how a protein should be created. Gene expression information describes whether or not a gene is expressed (active) in a location. Such information allows biologists to discover relationships between genes, in particular when and where they are co-expressed.

The gene expression information is acquired by experimentation on a mouse embryo, where each embryo corresponds to a particular point in time of the *developmental mouse*, i.e. the mouse from conception all the way up until birth. There are a total of 26 distinct time windows called Theiler Stages (TS) [83], each having its own anatomy, and corresponding anatomy ontology, called EMAP (an acronym for e-Mouse Atlas Project) [17]. There is also a number of 3D models which represent different stages of mouse development.

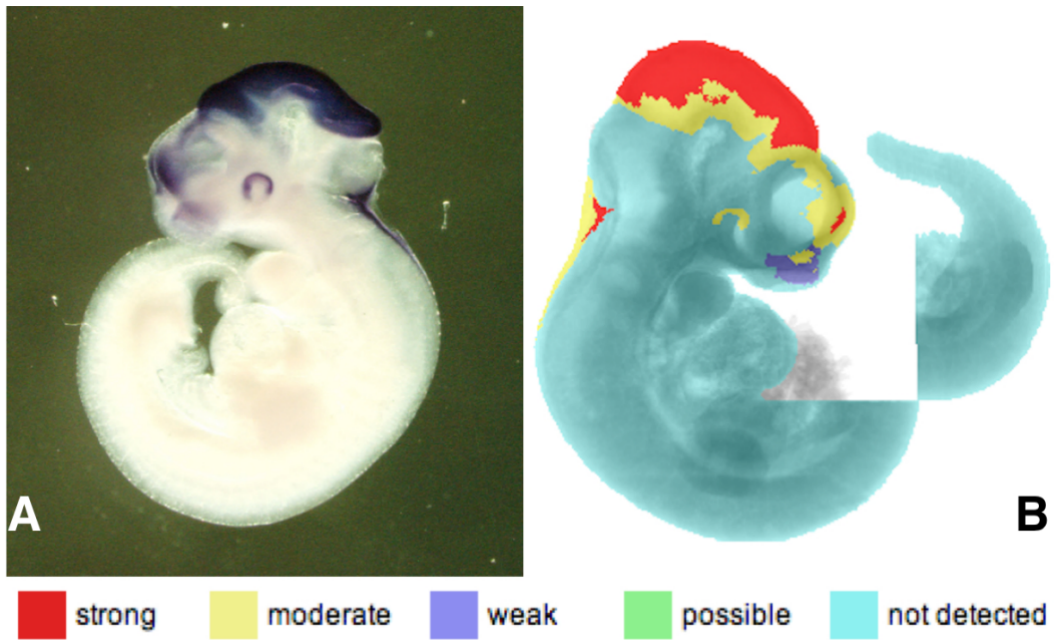


Figure 5.1: Example image of an experimental result (A) and its associated spatial annotation (B). Adapted from the EMAGE database.

Results derived from in situ hybridization (ISH) experiments are documented as images displaying an area of a mouse, during a particular Theiler stage, where certain subsections of the mouse are coloured (figure 5.1). Coloured areas indicate that the gene has been detected (or is expressed) in that location, while

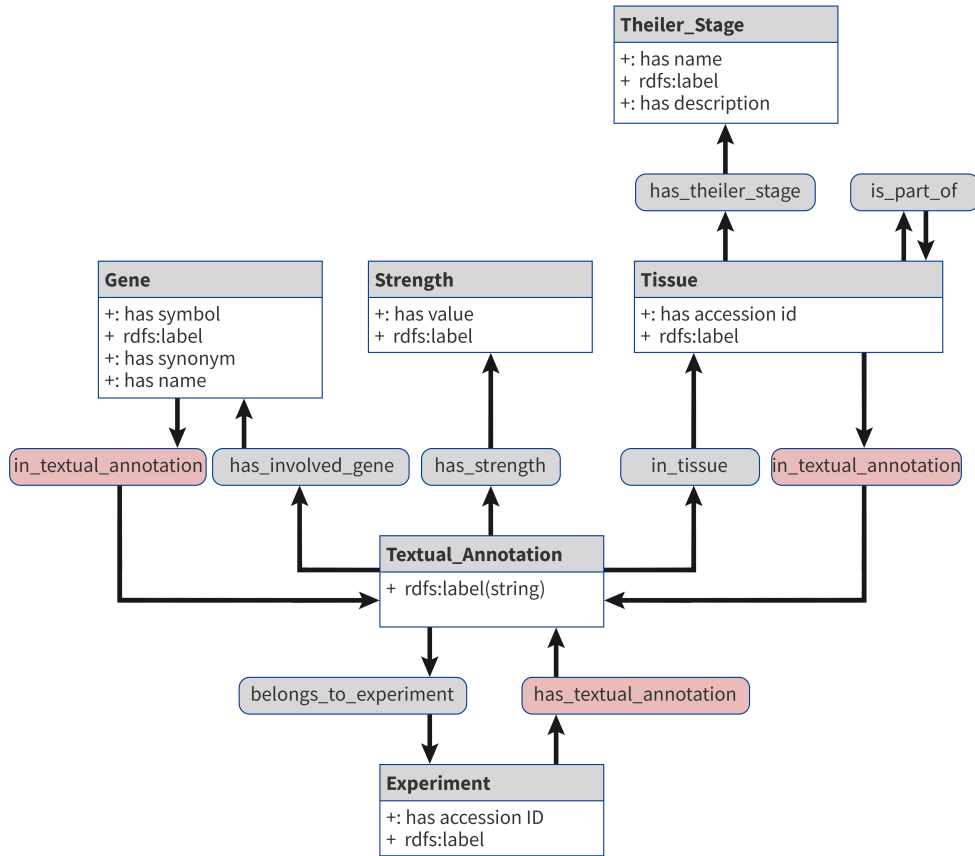


Figure 5.2: The EMAGE ontology, adapted from [28].

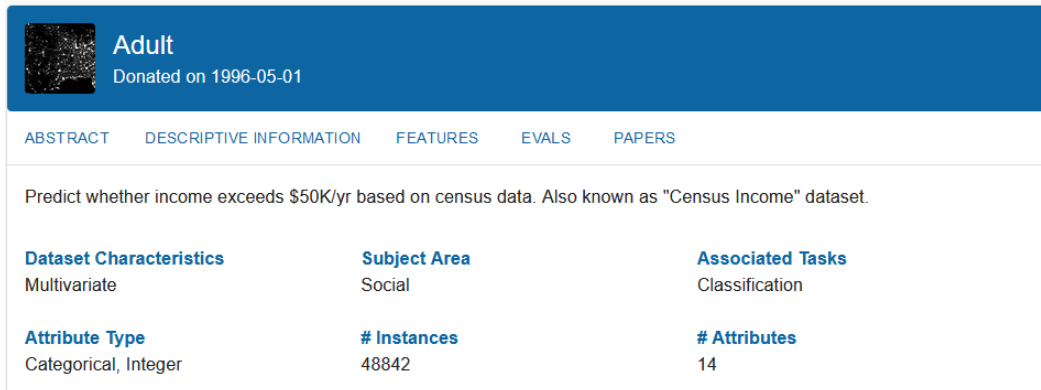
the colour intensity of each area indicates the strength (or level) of expression: the stronger the expression of the gene, the higher the intensity of the colour. Such results are analyzed by a human expert, manually, under a microscope, where the structures which the gene is expressed and the levels of expression are determined. Levels of expression are described using natural language terms such as strong, moderate, weak or present. For example, the gene *Bmp4* is strongly expressed in the future brain from TS15. Such statements are referred to as *textual annotations*, which represent the structured version of a subset of the original unstructured data (e.g. (A) in figure 5.1). Figure 5.2 shows the ontology of the EMAGE dataset. It demonstrates the relationship between genes and their strength of detection in a tissue during a particular Theiler stage, remarked by a textual annotation during an experiment.

Furthermore, experimental images (e.g. (A) in figure 5.1) can be mapped onto 3D models of the mouse, creating the so-called *spatial annotations* (e.g. (B) in figure 5.1). Spatial annotations are normally generated by EMAGE, while textual annotations are generated by the human experts performing the experiment. In this work, we will be only utilizing textual annotations.

5.3 UCI Machine Learning Repository

The UCI Machine Learning (ML) Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms, with over 1000 citations, making it one of the top 100 most cited “papers” in all of computer science [34].

Datasets in the UCI ML Repository are provided as standalone, downloadable CSV files. They are usually accompanied by a description explaining where the dataset was sourced from, the task the dataset was initially used for, and an explanation of the data it contains. It also provides metadata about the dataset itself, such as number of instances (objects), number of attributes, types of attributes encountered in the dataset, and whether the dataset contains missing values. An example of such metadata can be seen in figure 5.3.



Adult		
Donated on 1996-05-01		
ABSTRACT	DESCRIPTIVE INFORMATION	FEATURES
EVALS	PAPERS	
Predict whether income exceeds \$50K/yr based on census data. Also known as "Census Income" dataset.		
Dataset Characteristics	Subject Area	Associated Tasks
Multivariate	Social	Classification
Attribute Type	# Instances	# Attributes
Categorical, Integer	48842	14

Figure 5.3: Metadata of the *Adult* dataset of the UCI ML Repository.

Browsing through the UCI ML Repository data is possible through the dataset browser (figure 5.4). It provides the ability to filter datasets by parameters like characteristics, subject areas, and attribute types. Sorting capabilities are also provided, allowing the user to quickly identify datasets having the most objects or attributes, for example.

The UCI ML Repository contains hundreds of datasets (622 at the time of writing). Discussing each and every dataset is outside of the scope of this thesis. Instead, we discuss two of its most popular datasets:

- the Adult¹ dataset (1994 US census on income)
- the Agaricus Lepiota² dataset (properties of edible and poisonous mushrooms of the Agarica Lepiota family)

The datasets were chosen for being frequently used in FCA publications and for being some of the most popular datasets of the UCI ML Repository. Furthermore, they feature a variety of attribute types, allowing us to explore the data booleanization, data discretization, and conceptual scaling capabilities of the current and new FCA appropriation approaches.

5.3.1 Adult Dataset

The Adult (also known as “Census Income”) dataset predicts whether income exceeds \$50000/year based on US census income data in 1994. It features 48842 instances and 14 attributes, with their corresponding attribute types in parentheses: *age* (continuous), *workclass* (categorical), *fnlwgt* (continuous), *education* (categorical), *education-num* (continuous), *marital-status* (categorical), *occupation* (categorical), *relationship* (categorical), *race* (categorical), *sex* (categorical), *capital-gain* (continuous), *capital-loss* (continuous), *hours-per-week* (continuous), and *native-country* (categorical).

¹<https://archive.ics.uci.edu/ml/datasets/adult>

²<https://archive.ics.uci.edu/ml/datasets/mushroom>

Options

Characteristics

Subject Area

Associated Tasks

Attributes

Less than 10

10 to 100

More than 100

Instances

Less than 100

100 to 1000

More than 1000

Attribute Types

Datasets

Sort byEXPAND ALL

Internet Advertisements

ISOLET

Multiple Features

Human Activity Recognition Using Smartphones

FMA: A Dataset For Music Analysis

Classification

Classification

Classification

Classification, Clustering

Classification, Clustering

Multivariate

Multivariate

Multivariate

Multivariate, Time-Series

Multivariate, Time-Series

3.28K instances

7.8K instances

2K instances

10.3K instances

106.57K instances

1559 attributes

618 attributes

649 attributes

561 attributes

518 attributes

Figure 5.4: The dataset browser of the UCI ML Repository (rotated 90° anticlockwise).

5.3.2 Agaricus Lepiota Dataset

The Agaricus Lepiota (also known as “Mushrooms”) dataset describes the physical characteristics of 4208 edible and 3916 poisonous mushrooms belonging to the Agaricus Lepiota family of mushrooms. It features 8124 instances and 22 attributes, with their corresponding attribute types in parentheses: *cap-shape* (categorical), *cap-surface* (categorical), *cap-color* (categorical), *bruises?* (boolean), *odor* (categorical), *gill-attachment* (categorical), *gill-spacing* (categorical), *gill-color* (categorical), *gill-size* (categorical), *stalk-shape* (categorical), *stalk-root* (categorical with missing values), *stalk-surface-above-ring* (categorical), *stalk-surface-below-ring* (categorical), *stalk-color-above-ring* (categorical), *stalk-color-below-ring* (categorical), *veil-type* (categorical), *veil-color* (categorical), *ring-number* (categorical), *ring-type* (categorical), *spore-print-color* (categorical), *population* (categorical), and *habitat* (categorical).

5.4 Existing Approaches

5.4.1 ToscanaJ

ToscanaJ is a Java-based implementation of the Toscana software, originally written in C++. It is a complete suite of tools for creating and using Conceptual Information Systems (CIS). CIS are defined as “systems that store, process, and present information using concept-oriented representations supporting tasks like data analysis, information retrieval, or theory building in a human-centered way” [20]. Hence, analyzing data originating from a structured data source, using FCA as a data analysis technique, is a CIS.

ToscanaJ can create *realized scales*, when provided with a many-valued context and a conceptual scale. Realized scales are defined as “formal contexts which have the objects of the many-valued context as objects and the attributes of the scale as attributes” [20]. It then visualizes the realized scales as concept lattices. ToscanaJ is defined in its manual as a browsing frontend [19], meaning that it only acts as a read-only view to explore the conceptual information.

A component of ToscanaJ called Elba allows connecting to relational databases and creating conceptual systems out of the relational data. An extension has also been developed that allows connecting to RDF triple stores [29]. Editing of diagrams and contexts is possible, but only for data produced out of relational database tables or RDF triples.

Siena, another component of ToscanaJ, performs similar operations to Elba but without the need for a database. It is most suited for novice FCA users who want to setup a small ToscanaJ system. Similar to ToscanaJ, it does not allow direct editing of data; it is stated that this is planned as a future update. Siena does, however, accept formal contexts in the Burmeister (.cxt) format and conceptual schema files from ToscanaJ as input.

Developing a CIS with the ToscanaJ toolkit involves a workflow typical to the development of other information systems, where the development of the system is broken down into phases such as design, implementation and testing. During the design and implementation phase, a CIS engineer (an expert user of the CIS) will develop a conceptual schema in collaboration with the domain expert (a non-expert user of the CIS). The schema defines the structures which will be used to access the system and typically involves the CIS engineer having to define conceptual scales manually in Elba. The testing phase is usually a highly iterative and time-consuming process, involving several back and forths between the CIS engineer and the domain expert. Finally, once the system is considered complete, the CIS end-user uses ToscanaJ (the browsing frontend of the CIS) to explore, or analyze the data of, the underlying database or triple store, using the conceptual schema developed in Elba during the design and implementation phase. The workflow is depicted in figure 5.5.

We now present an example of the ToscanaJ workflow, using the example ‘Diabetes’ database provided in ToscanaJ. It contains various pathological characteristics of a sample population having diabetes, such as their sex, family history of diabetes, cholesterol and triglyceride levels, and so on. First, the realized scales are created manually in Elba (figure 5.6). Each scale is defined as a formal context, where the attributes of the context are the characteristics

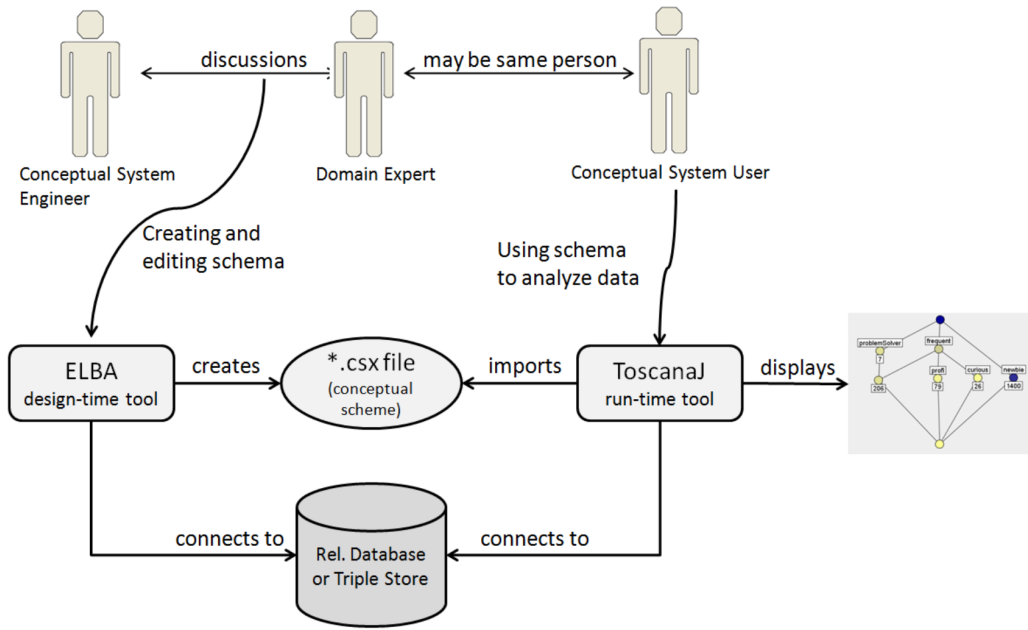


Figure 5.5: Workflow of a ToscanaJ Conceptual Information System.

themselves, and the objects are the filters which will match against the objects in the database (the equivalent of the `WHERE` clause in SQL and SPARQL). For example, a person is classified as having highly pathological triglyceride levels when `TRIG_0 > 200`, while normal triglyceride levels are satisfied with the condition `TRIG_0 > 0 AND TRIG_0 ≤ 150`. The scale can be modified by manually changing the crosses in the formal context, or extended by manually adding or removing attributes (characteristics) and objects (filters) to it. The concept lattice of the realized scale is depicted in figure 5.7. Finally, the conceptual schema can be imported in ToscanaJ, where the realized scales can be used to query the actual data of the underlying database or triple store (figure 5.8).

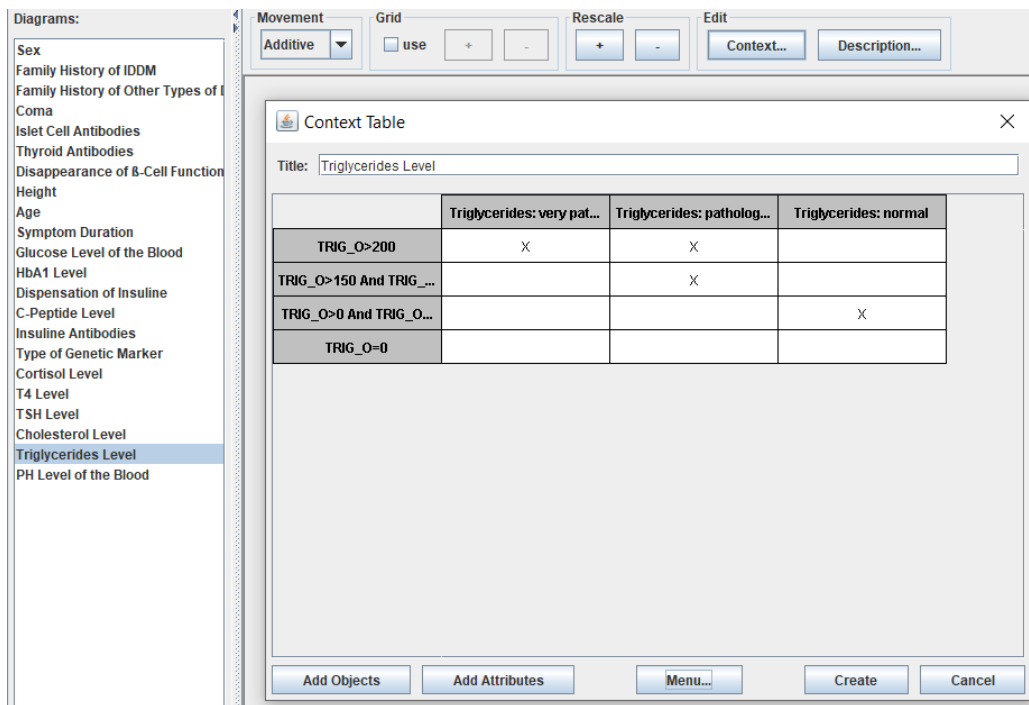


Figure 5.6: Definition of realized scales in Elba.

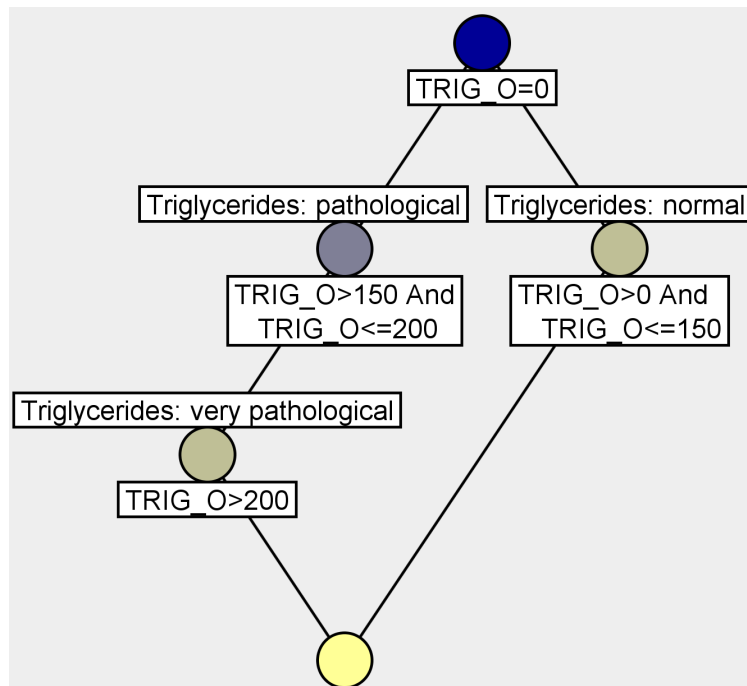


Figure 5.7: A lattice of the triglyceride scale of figure 5.6, in Elba.

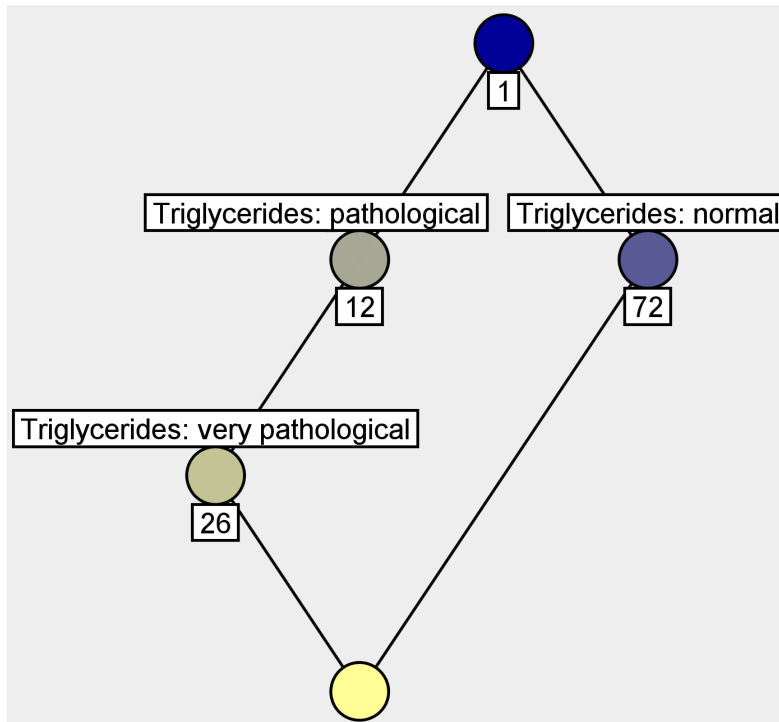


Figure 5.8: Using realized scales, defined in Elba, to analyze data in ToscanaJ.

5.4.2 ConExp

ConExp (Concept Explorer) is, primarily, an FCA visualization tool, available as Open-source Software (OSS) at Sourceforge [93]. It “implements basic functionality needed for study and research of Formal Concept Analysis” [94]. It can be used for creating formal contexts by hand, computing formal concepts in a formal context, and visualizing a formal context as a concept lattice. It also supports examining implications between attributes valid in a context and calculating association rules. A modern re-implementation called ConExp-NG³ (Concept Explorer Next Generation), by different authors, exists as OSS on GitHub, with the re-implementation focusing on usability, maintainability and extensibility.

ConExp does not support creating a formal context from datasets or structured data sources. As depicted in figure 5.9, a simple formal context has been created from a small subset of the Agaricus Lepiota dataset of the UCI ML

³<https://github.com/fcatools/conexp-ng>

	A	B	C	D	E	F	G	H	I
		bruises?	gill-size-broad	gill-size-narrow	veil-type-partial	veil-type-universal	ring-number-none	ring-number-one	ring-number-two
0		X	X		X		X		
1		X		X	X				X
2				X	X		X		
3		X	X		X			X	
4				X	X		X		

Figure 5.9: Creating and/or modifying a formal context in ConExp.

Repository. ConExp only allows importing formal contexts in the Burmeister format, or creating formal contexts by hand, in an Excel-like fashion similar to ToscanaJ. The formal context can then be visualized as a concept lattice and display the number of concepts, as shown in figure 5.10.

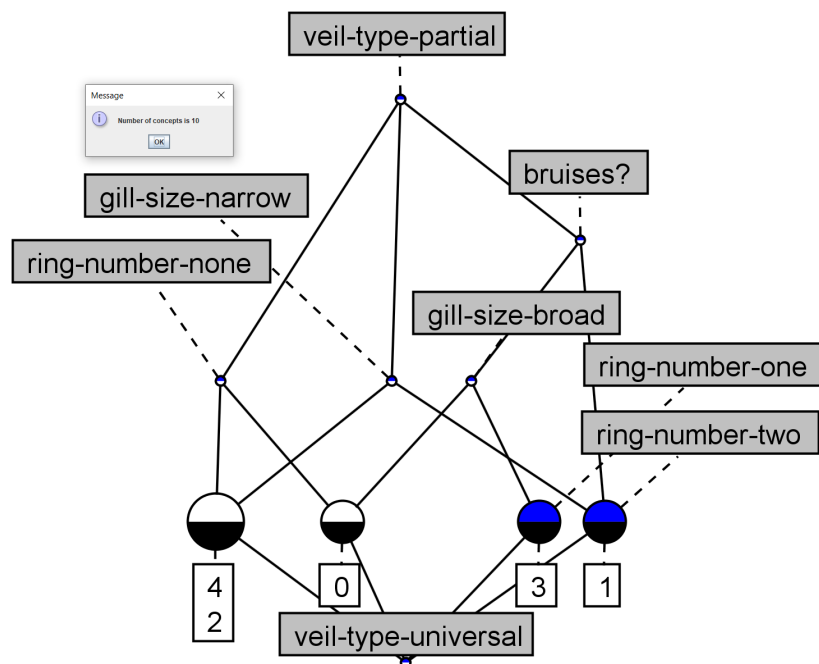


Figure 5.10: Concept counting and visualization in ConExp.

5.4.3 CUBIST Scalingless Approach

The CUBIST scalingless approach was developed by Dau [27] for the CUBIST EU FP7 project, as a Proof of Concept (PoC) for creating formal contexts out of RDF triple stores.

Triple stores are a type of graph database that store semantic data, using the Resource Description Framework (RDF) for describing the data. The data in an RDF triple store are stored in triples, in the RDF *subject-object-predicate* format, and queried using SPARQL (a recursive acronym for SPARQL Protocol and RDF Query Language), an RDF query language. The RDF subject-object-predicate format provides the ability to take any subject and connect it to any other object, by using the predicate (verb) to demonstrate the type of relationship between them [67, 14]. For example, the sentence “Ioanna was born in 2021” can be stored as an RDF statement in a triple store, as it describes the relationship between the subject ‘Ioanna’ and the object ‘2021’ of the sentence, by the predicate ‘born in’ (figure 5.11).

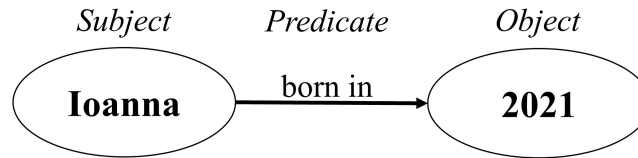


Figure 5.11: An example of an RDF statement.

A prototype tool, called SPARQL2Context, was created to demonstrate the approach. SPARQL2Context is not publicly available, therefore this section explains the approach without providing any screenshots of the tool.

The approach starts by converting the results of a SPARQL query to a formal context, without having the possibility to further manipulate the context. Essentially, the SPARQL query is the driving force behind the creation of the formal context. The SPARQL queries are written by hand, and knowledge of SPARQL and RDF triple stores is required. Similar to ToscanaJ, a SPARQL expert and domain expert are required, although one person could have both roles. SPARQL queries return results as tables, where the columns are the

query variables and the rows are the objects satisfying the conditions of the SPARQL query. Empty cells (missing, unknown, or empty values) are possible if the `OPTIONAL` SPARQL clause is applied. To resolve ambiguity, the variable names of the SPARQL query determine whether the variable is to be used as an object, if the name starts with ‘o’, or an attribute, if it starts with ‘a’. Variables not following those naming conventions are not used in the generated context. Where more than one variable starts with ‘o’ or ‘a’, the results of these variables are concatenated (using dash as a divider) to produce object names or attribute names, respectively. When a row in the result set produces both an object and an attribute, the incidence relation is set by placing a row in the corresponding cell of the formal context [27].

An artificial example of the approach is depicted in figure 5.12. The first 3 rows of the formal context have all their attribute values set to false, as no incidence relation exists between the corresponding object-attribute combinations. O4 only features A3, and O5–O6 only feature A4. Objects O5–O7 all feature attributes A4 and A5–A6, and as such have an incidence relation set at the A4 and A5–A6 formal attributes.

Obj1	Obj2	Att1	Att2
O1			
O1	O2		
	O3		
		A1	
		A1	A2
			A2
O4		A3	
O5	O6	A4	
O5	O7	A5	A6
O5	O7	A4	

Results	A1	A1-A2	A2	A3	A4	A5-A6
O1						
O1-O2						
O3						
O4				×		
O5-O6					×	
O5-O7					×	×

Figure 5.12: Transformation of a SPARQL query result (left) to an unscaled formal context (right) in SPARQL2Context. Adapted from [27].

5.4.4 CUBIST Scaleful Approach

The CUBIST scaleful approach was developed by Orphanides for the CUBIST EU FP7 project, again as a Proof of Concept (PoC) for creating formal contexts out of RDF triple stores. It is a precursor of the formal context creation capabilities employed in the interactive approach of CUBIST’s final prototype (see section 5.4.5). It extends the ideas of the CUBIST scalingless approach by allowing objects and attributes returned from a SPARQL query to be interchangeable, effectively allowing the transposition of a formal context and not requiring object variable names to start with an ‘o’ or attribute variable names to start with an ‘a’. It also provides limited scaling capabilities, such as making decisions on how to scale categorical, boolean and continuous attributes. A prototype tool, called SPARQL2FCA, was similarly created to demonstrate the approach. SPARQL2FCA is not publicly available and has no associated publication; the reader is instead referred to Dau and Andrews [28] and Melo et al [60], where parts of the formal context creation capabilities of this approach have been incorporated.

The approach starts by converting the results of a SPARQL query to a formal context, allowing for both unscaled and scaled formal contexts to be created. Similar to the CUBIST scalingless approach in subsection 5.4.3, the creation of the formal context is primarily driven by the SPARQL query itself. The SPARQL queries are written by hand, and SPARQL expertise is required to produce the queries. SPARQL query results are returned in tables, where the columns are the query variables, and the rows are the objects satisfying the conditions of the query. Missing, unknown, or empty values are allowed. There is no need to denote which variable is an object or an attribute in the SPARQL query itself; instead, a choice must be made about which variables are to be treated as objects or attributes after the result set is returned. When more than one variable are to be treated as attributes, no concatenation takes place; each attribute is converted individually. Where more than one variable are to be treated as objects, the results are concatenated (using dash as a divider)

to produce object names. Lastly, the possibility is given to create an unscaled formal context, using the same ideas of the CUBIST scalingless approach, or a formal context with conceptual scaling applied.

The screenshot shows the SPARQL2FCA Context Scaler application. It has three tabs: 'Build It', 'Scale It', and 'Save It'. The 'Build It' tab is active.

Your Query

```

SELECT ?jb ?disc ?adv ?numvac
WHERE {
  ?jb joci:name ?jbname .
  ?disc joci:name ?discname .
  ?adv joci:adv ?advname .
  ?adv joci:jb ?jb .
  ?adv joci:disc ?disc .
  ?adv joci:numvac ?numvac .
}
GROUP BY ?discname ?jbname
ORDER BY DESC(?NumOfVacancies)

```

Repository Connection:
Repository ID:
Execute

SPARQL Query Results

Jobboard	Advertiser	NumOfVacancies
JB_163	ADV_0	107247
JB_83	ADV_133121	23595
JB_36	ADV_0	21335
JB_83	ADV_312636	18537
JB_4	ADV_487187	12220

Unscaled Context

Column to treat as objects: Column to treat as attributes: Create formal context

Advertiser	JB_163	JB_83	JB_36	JB_4	JB_11	JB_97	JB_100	JB_1	JB_86
ADV_0	X	.	X	.	.	.	X	.	.
ADV_133121	.	X
ADV_312636	.	X
ADV_487187	.	.	.	X
ADV_712144	.	.	.	X
ADV_37746	X

Export formal context Visualise in CUBIX

Figure 5.13: Transformation of a SPARQL query result to an unscaled formal context in SPARQL2FCA.

An example of the approach is depicted in figure 5.13. The dataset in this example originates from a CUBIST use-case partner, consisting of job vacancies advertised in the United Kingdom’s leading job boards, as well as employers’ own websites [27, 65]. The SPARQL query returns the number of vacancies each advertiser has placed on a job board (the query is limited to 15 results for presentation purposes). In this example, advertisers are used as objects and job boards are used as attributes, although transposition is possible by having the flexibility to choose whether a variable is an object or an attribute. Finally, an unscaled formal context is created, using job boards as objects and advertisers as attributes. Where a job board features a particular advertiser, a corresponding cross is placed in the formal context.

In figure 5.14, the same dataset is used to produce a formal context with conceptual scaling applied. Here, job boards and advertisers are used as objects, while number of vacancies are used as an attribute and declared as continuous (numerical), where equal width binning is used to create four disjoint ranges of values (refer to section 4.7 for a reminder on equal width binning). It is not possible to define the number of bins; all binning techniques in this approach are hardcoded to always produce four bins. Finally, a formal context is created using Jobboard–Advertiser pairs as objects, and four ranges of number of vacancies as formal attributes.

The screenshot shows the 'SPARQL2FCA Context Scaler' application window. It has three tabs: 'Build It', 'Scale It' (selected), and 'Save It'. The 'Scale Your Query' section contains a table with columns: Name, Type, Is Attribute, and Convert. Below this table are dropdowns for 'Scaling: Discrete' and 'Binning: EqualWidth', along with a 'Create formal context' button. The 'Scaled Context' section displays a table with columns: Jobboard--Advertiser, NumOfVacancies-1300to<26486, NumOfVacancies-26486to<52972, NumOfVacancies-52972to<105944, and NumOfVacancies-105944to<=107247. At the bottom right are buttons for 'Export formal context' and 'Visualise in CUBIX'.

Name	Type	Is Attribute	Convert
Jobboard	Categorical	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Advertiser	Categorical	<input type="checkbox"/>	<input checked="" type="checkbox"/>
NumOfVacancies	Continuous	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Scaling: Discrete Binning: EqualWidth Create formal context

Jobboard--Advertiser	NumOfVacancies-1300to<26486	NumOfVacancies-26486to<52972	NumOfVacancies-52972to<105944	NumOfVacancies-105944to<=107247
JB_163--ADV_0	-	-	-	X
JB_83--ADV_133121	X	-	-	-
JB_36--ADV_0	X	-	-	-
JB_83--ADV_312636	X	-	-	-
JB_4--ADV_487187	X	-	-	-
JB_4--ADV_712144	X	-	-	-
JB_11--ADV_37746	X	-	-	-
JB_97--ADV_668209	X	-	-	-
JB_100--ADV_0	X	-	-	-
JB_1--ADV_28	X	-	-	-

Export formal context Visualise in CUBIX

Figure 5.14: Transformation of a SPARQL query result to a scaled formal context in SPARQL2FCA.

5.4.5 CUBIST Interactive Approach

The CUBIST interactive approach [27] incorporated the scalingless and scaleful CUBIST approaches in a web application called Cubix [60], CUBIST’s state-of-the-art visualization tool. While some of the CUBIST software components have been released to the public as Open-source Software (OSS), other components of the CUBIST architecture are commercial, proprietary software. Therefore,

it is no longer possible to access or replicate the CUBIST system without commercial access to some of its components. A high-level overview of the CUBIST workflow is depicted in figure 5.15.

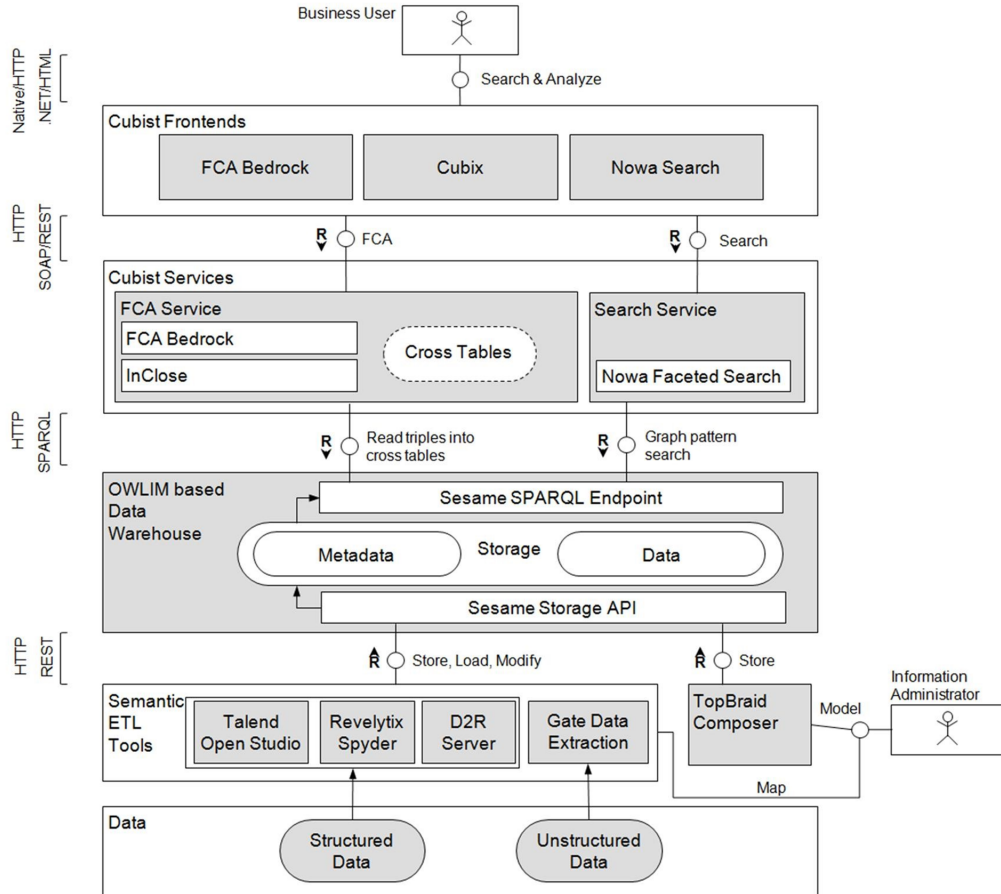


Figure 5.15: The CUBIST architecture.

Contrasting to the previous two CUBIST approaches, knowledge of SPARQL or RDF triple stores is not necessary. Rather, the approach is an interactive one: an initial SPARQL query is executed in the backend, retrieving the key entities of the ontology being investigated and presenting them to the end-user. The end-user is then able to browse through all the possible values of each entity, filtering the results where deemed necessary, in a web component called NowaSearch (figure 5.16). Decisions can be made about whether an entity should be treated as an object or an attribute. As the approach is an interactive one, there is no need to denote which variable is an object or an

attribute in the SPARQL query itself; instead, a choice must be made about which variables are to be treated as objects or attributes, after the result set is returned. When more than one variable are to be treated as attributes, no concatenation takes place; each attribute is converted individually. Where more than one variable are to be treated as objects, the results are concatenated (using dash as a divider) to produce object names. Limited conceptual scaling capabilities are possible, with support for categorical, boolean or continuous scales, as depicted in figure 5.17. Once the end-user is satisfied with how the data should be treated, a new SPARQL query is produced, incorporating the filters the end-user has selected for the analysis (the equivalent of the **WHERE** clause in SQL and SPARQL). This produces a SPARQL result set, which is then converted into a scaled formal context based on the selections of the end-user. Finally, the user is able to visualize the produced formal context as a lattice, or other intuitive visualizations such as sunburst diagrams, matrices and trees [60], in Cubix.

The screenshot shows the NowaSearch interface. On the left is a sidebar with filters for 'Experiment', 'Gene', 'has symbol', 'Strength', 'has value', 'Textual Annotation', 'Theiler Stage', and 'Tissue'. The 'has symbol' filter is set to 'Bmp4'. The 'Strength' filter is set to 'detected', 'strong', and 'moderate'. The 'has value' filter is set to 'detected'. The 'Textual Annotation' filter is set to 'eye lid', 'anterior', 'external', 'olfactory', 'aorta', 'pulmonary artery', 'stomach', 'rectum', 'midgut', 'left lung', 'right lung', 'mesoderm', 'allantois', 'chorion', 'diencephalon', 'mesenchyme', and 'extraembryonic ectoderm'. The 'Theiler Stage' filter is set to 'TS09', 'TS15', 'TS20', 'TS23', 'TS11', and 'TS17'. The 'Tissue' filter is set to 'extraembryonic component', 'infundibular recess of 3rd ventricle', 'latero-nasal process', 'medial-nasal process', 'vibrissa', 'pineal primordium', 'cochlea', 'utricle', 'eyelid', 'anterior', 'external', 'olfactory', 'aorta', 'pulmonary artery', 'stomach', 'rectum', 'midgut', 'left lung', 'right lung', 'mesoderm', 'allantois', 'chorion', 'diencephalon', 'mesenchyme', and 'extraembryonic ectoderm'. A 'Clear' button is at the bottom of the sidebar.

On the right is a data table with columns 'Object: Tissue', 'has symbol', and 'has value'. The table contains 30 rows of data. The 'has symbol' column is 'Bmp4' for all rows. The 'has value' column is 'strong' for the first 24 rows and 'detected' for the last 6 rows.

Object: Tissue	has symbol	has value
extraembryonic component TS09	Bmp4	strong
infundibular recess of 3rd ventricle TS15	Bmp4	strong
latero-nasal process TS20	Bmp4	strong
medial-nasal process TS20	Bmp4	strong
vibrissa TS23	Bmp4	strong
pineal primordium TS23	Bmp4	strong
cochlea TS23	Bmp4	strong
utricle TS23	Bmp4	strong
eyelid TS23	Bmp4	strong
anterior TS23	Bmp4	strong
external TS23	Bmp4	strong
olfactory TS23	Bmp4	strong
aorta TS23	Bmp4	strong
pulmonary artery TS23	Bmp4	strong
stomach TS23	Bmp4	strong
rectum TS23	Bmp4	strong
midgut TS23	Bmp4	strong
left lung TS23	Bmp4	strong
right lung TS23	Bmp4	strong
mesoderm TS11	Bmp4	detected
allantois TS11	Bmp4	detected
chorion TS11	Bmp4	detected
diencephalon TS15	Bmp4	detected
mesenchyme TS17	Bmp4	detected
extraembryonic ectoderm TS09	Bmp4	detected

Figure 5.16: Using NowaSearch to interactively navigate and filter the data of an ontology in a triple store, in CUBIST.

Scaling parameters for each attribute

has symbol (Gene)
 add property name: ☒ Yes
 attribute type:

has value (Strength)
 add property name: ☒ Yes
 attribute type:

General scaling parameters

minSupport Object: minSupport Attribute: faultTolerance:

Data Table

Object: Tissue	has symbol	has value
extraembryonic component TS09	Bmp4	strong
infundibular recess of 3rd ventricle TS15	Bmp4	strong
latero-nasal process TS20	Bmp4	strong
medial-nasal process TS20	Bmp4	strong
vibrissa TS23	Bmp4	strong
pineal primordium TS23	Bmp4	strong
cochlea TS23	Bmp4	strong
utricle TS23	Bmp4	strong
eyelid TS23	Bmp4	strong

Figure 5.17: Conceptual scaling in CUBIST.

An example of the approach starts with figure 5.16, where the CUBIST end-user is exploring the ontology of the EMAGE dataset. The initial SPARQL query has returned the key entities of the dataset, such as the genes, tissues and Theiler stages of mouse development. The user is interested in examining in which tissues the gene Bmp4 is strongly or moderately expressed, as evidenced by the filters the user has added at the left of the image. On the right, the user is presented with mice tissues that satisfy the conditions of the filters. The user is then prompted to choose how each attribute in the query should be treated or scaled, as demonstrated in figure 5.17. The user is now able to create a formal context having tissues as objects, the Bmp4 gene as the only attribute, and strong or moderate levels of detection as possible attribute values. Finally, the formal context is visualized as a concept lattice in Cubix (figure 5.18).

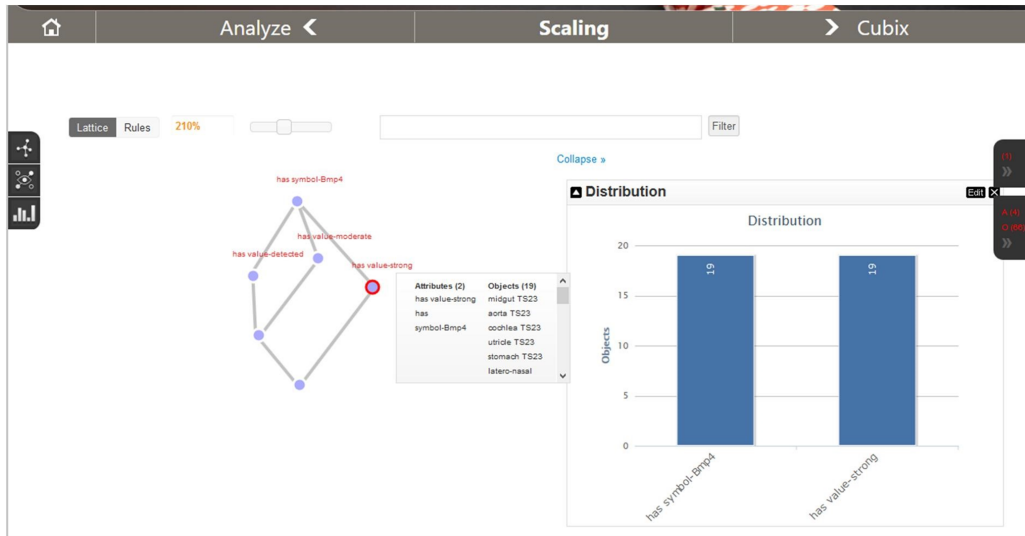


Figure 5.18: Visualizing a formal context as a concept lattice in Cubix.

5.5 New Approaches

5.5.1 FcaBedrock

FcaBedrock⁴ is an OSS tool for creating formal contexts for Formal Concept Analysis (FCA), placing sole emphasis on the FCA data appropriation process [9, 10]. In other words, it is a data preparation, or pre-analysis tool, for cleaning or transforming data for FCA. It was initially developed after a proposal at CSTIW'09 was made that a tool be developed, to automate the process of FCA data appropriation and address issues such as deciding which data to convert, and how to convert them [3]. Initial versions of the tool were created as part of a BSc Computing undergraduate final year project [64], while subsequent versions have been developed as part of this thesis.

FcaBedrock implements a novel approach, using a process of guided automation. The guided automation is achieved by the user supplying the tool with metadata for appropriation, such as which attributes are to be included, how each attribute should be treated, and how they should be named. Such metadata can be also automatically detected by FcaBedrock, after scanning a dataset provided in any Delimiter Separated Value (DSV) format such as CSV

⁴<https://github.com/trashr0x/fcabledrock>

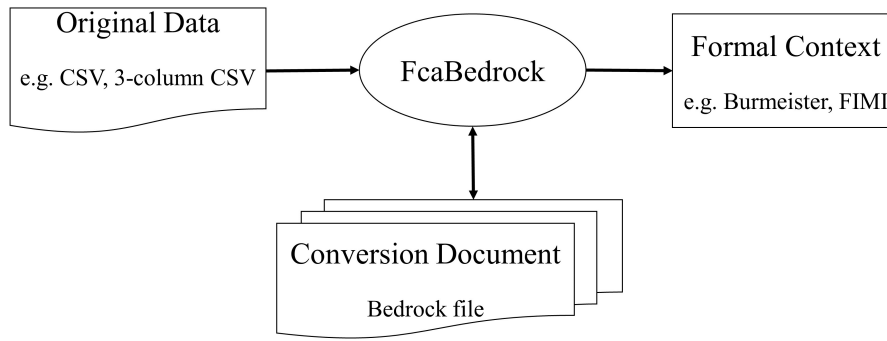


Figure 5.19: High level overview of the FcaBedrock process.

or TSV (comma-separated and tab-separated, respectively). The metadata can then be used to convert the data into formal contexts, in the Burmeister (.cxt) and FIMI (.dat) standard FCA formats. The metadata are reusable and exportable as text documents called Bedrock files. Bedrock files act as a reference of how a dataset was appropriated, allowing for analyses to be repeated, different interpretations to be made in subsequent analyses, and for appropriation metadata to be shared with other users of the tool. A high level overview of the FcaBedrock process is depicted in figure 5.19.

FcaBedrock supports two data input formats: many column DSV and three column DSV. Any delimiter can be used to distinguish between columns in the dataset, though comma is the most traditional option. In many column CSV files, each row represents an object and each column represents an attribute. Such files are typical when exporting datasets from a relational database, for example. The three column CSV format has gained popularity as a standardized format and utilizes the same approach as the Resource Description Format (RDF) [1]. This enables importing datasets exported from semantic databases, such as RDF triple stores. When metadata autodetection is used during the reading of a three column CSV file, FcaBedrock will treat the first value as a formal object name, the second value as an attribute name, and the third value as an attribute value.

FcaBedrock supports categorical, boolean, continuous and date attribute as possible attribute types. Categorical are many-valued attributes, producing one

formal attribute for every value of the attribute in the dataset (the equivalent of a nominal scale). Boolean attributes are treated as a subset of categorical values, where only two possible values are allowed (the equivalent of a dichotomic scale). Continuous attributes can be treated as categorical attributes, where each numerical value of the attribute produces a formal attribute in the formal context (this would make sense when investigating individual Theiler stages in EMAGE, for example). They can be also discretized, using user-defined disjoint ranges (e.g. *0-10*, *10-20...*), hierarchical scaling (the equivalent of ordinal scaling, e.g. *>10*, *>20...*), or automatically calculated equal width and equal frequency bins with a user-defined number of bins. Date attributes are treated in the same way as continuous attributes.

Through the guided automation process, FcaBedrock provides the ability to exclude attributes (attribute exclusion), or restrict an attribute to specific attribute values (object exclusion). For example, an attribute could contain free-text, making it unsuitable for conversion, or may not be of importance to the current analysis. Furthermore, by restricting an attribute to specific attribute values, it is possible to create sub-contexts that focus on particular business questions. Using the Agaricus Lepiota dataset as an example, it is possible to restrict the analysis to only contain poisonous mushrooms. This will result in a formal context containing only objects satisfying these restrictions. An example of the FcaBedrock workflow is presented below.

In this example, the EMAGE dataset is exported from a triple store as a three column CSV file. The file contains 732681 triples of mouse development data. Let us assume that an analyst wants to investigate whether gene Bmp5 is strongly expressed in the endoderm and mesoderm tissues, across a range of Theiler stages. The workflow starts by the user loading the EMAGE dataset in FcaBedrock. FcaBedrock will then preview some lines from the file, allowing the user to quickly verify that the correct dataset is loaded, and make decisions about whether the dataset contains relational data or triples, whether the first line should be used to produce attribute names for the columns, and what delimiter to use (figure 5.20).

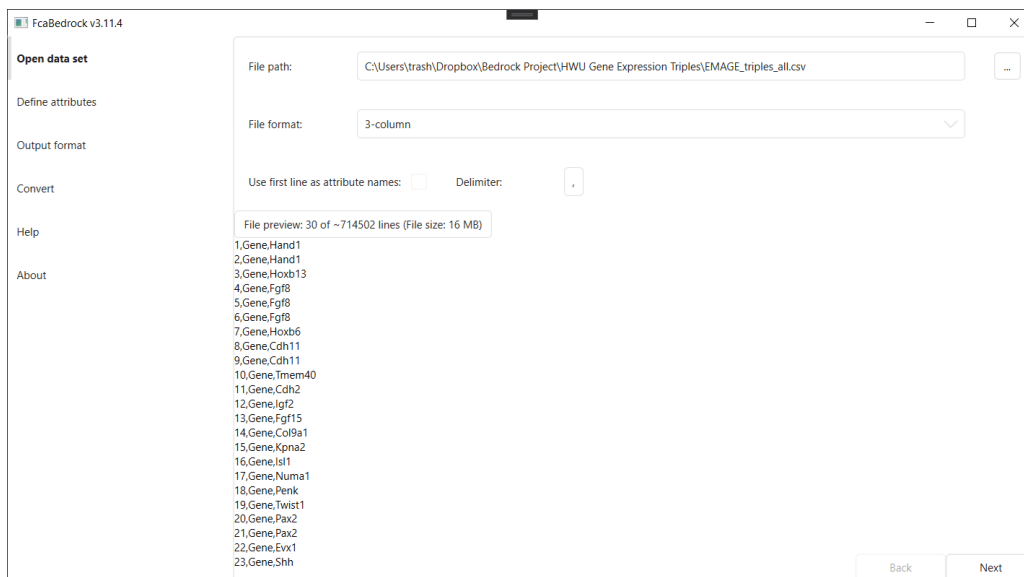


Figure 5.20: Previewing a three column CSV file in FcaBedrock.

Next, the user chooses to autodetect the EMAGE metadata on-the-fly. FcaBedrock will scan the file, and present the user with the list of attributes present in the dataset, along with their respective attribute values. For each attribute, the user is able to choose whether it should be converted or not, whether the analysis should be restricted to specific attribute values, and how each attribute should be scaled. As per the inquiry, the user restricts the Gene attribute to ‘Bmp5’, the Tissue attribute to ‘endoderm’ and ‘mesoderm’, the Strength attribute to ‘strongly detected’, and conceptually scales Theiler stages using four bins of equal frequency (figure 5.21). Once FcaBedrock automatically calculates the equal frequency bin ranges and presents them to the analyst, a further decision is made to only investigate Theiler Stages 3–8. Finally, the user chooses to output the formal context in the Burmeister format (figure 5.22). FcaBedrock creates the formal context, which can now be further analyzed for association rules, implication rules, or visualized as a concept lattice in tools such as ConExp.

5.6 Concluding Summary

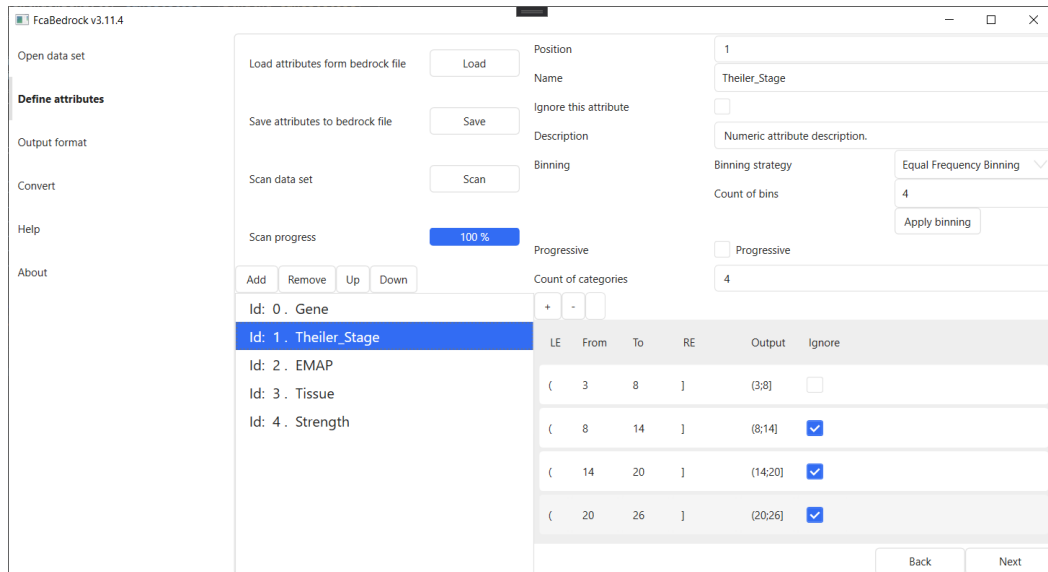


Figure 5.21: Automatically detecting attributes, and defining conceptual attribute scales, in FcaBedrock.

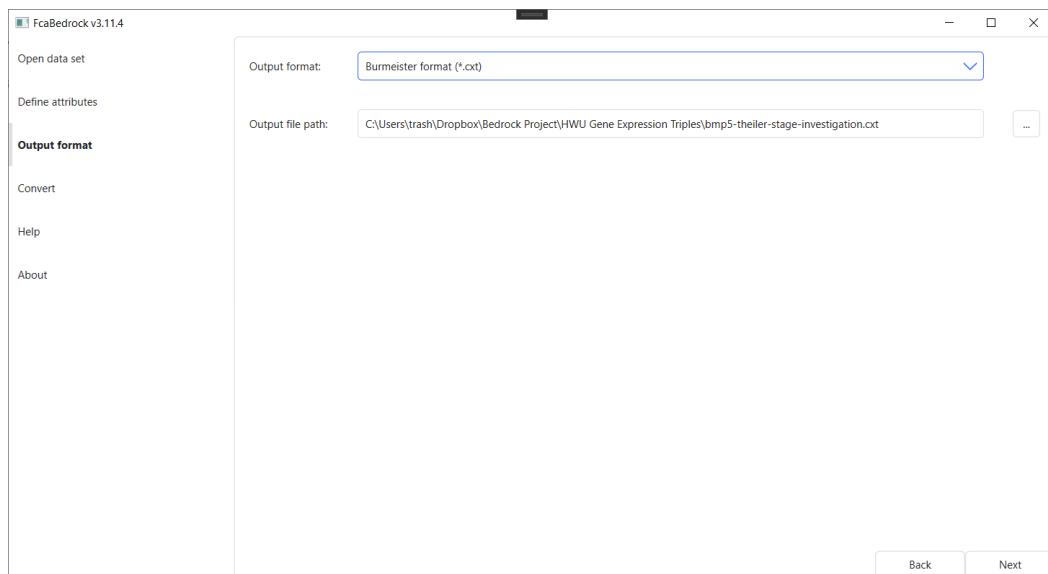


Figure 5.22: Outputting a formal context, in the Burmeister format, in FcaBedrock.

5.6 Concluding Summary

This chapter explained the existing and new approaches for appropriating data for FCA. It identified how ToscanaJ, the CUBIST scalingless approach and the CUBIST scaleful approach are heavily dependent on an underlying structured data source, and how expertise in database query languages, such as SQL

and SPARQL, are required to appropriate the data for FCA. The approach of ConExp involves either importing a formal context created outside of ConExp, or creating a formal context by hand. The CUBIST interactive approach improves on the CUBIST scalingless and scaleful approaches by employing an interactive approach, where the end-user is driving the appropriation, rather than hardcoded SPARQL queries. Finally, the FcaBedrock approach is explained, which uses a novel approach of guided automation and allows for the appropriation of data from any structured source, provided that the data have been first exported from a structured data source as CSV files.

ToscanaJ, while powerful with its data exploration and conceptual scaling capabilities as a CIS, is not a good candidate for data appropriation. In fact, the process to appropriate datasets from the EMAGE and UCI ML Repository case studies, or any other structured source, is a counter-intuitive one: datasets have to be exported from the underlying data source, and then re-imported in a relational database, or a triple store, using the ToscanaJ triple store extension developed in [29]. Furthermore, conceptual scales have to be manually defined through a highly iterative process, usually involving several back and forths between the database expert and the domain expert. The drawbacks are, however, reasonable and expected; ToscanaJ is a browsing frontend, meant to act as a read-only view to explore conceptual information, and was not built for appropriating data for FCA. Consequently, ToscanaJ will not be further evaluated in this work. Similarly, ConExp will not be further evaluated in this work due to providing no data appropriation capabilities other than creating formal contexts in an Excel-like, manual fashion.

The three approaches from CUBIST and the FcaBedrock approach are further experimented with and evaluated in chapters 6 and 7.

Chapter 6

Experiments

6.1 CUBIST Scalingless Approach

6.1.1 Introduction

Dau’s CUBIST scalingless approach is described in detail in [27]. It involves the creation of scalingless formal contexts out of triple stores, through the use of SPARQL queries as the driving force behind the analysis (refer to section 5.4.3 for an explanation of the approach). In the following subsection, it is demonstrated how object restricted, attribute restricted, object unrestricted and attribute unrestricted analysis can take place using SPARQL and its `OPTIONAL` clause or `UNION` operator. The experiment uses the EMAGE case study as its data source.

6.1.2 Utilized Applications

The software applications used in this approach are:

- SPARQL2Context, which creates scalingless formal contexts out of SPARQL queries.
- OWLIM, an RDF triple store.
- ConExp, a concept lattice visualizer.

6.1.3 Experimental Results

Figure 6.1 shows a SPARQL query executed against the EMAGE dataset. The query aims at retrieving the tissues of Theiler Stage 7, along with the genes that are detected in those tissues.

```

1  SELECT DISTINCT ?obj ?att WHERE {
2  {
3      ?x1 rdf:type :Tissue ; rdfs:label ?obj .
4      ?x1 :has_theiler_stage :Theiler_stage_TS07 .
5      ?x3 rdf:type :Gene ; rdfs:label ?att .
6      ?x2 rdf:type :Textual_Annotation .
7      ?x2 :in_tissue ?x1 .
8      ?x2 :has_involved_gene ?x3 .
9      ?x2 :has_strength :level_detected_derived .
10 }
11 ORDER BY ?obj ?att

```

Figure 6.1: A SPARQL query of EMAGE data used in SPARQL2FCA. Adapted from [27].

The results of the query are shown in table 6.1. There is a total of 5 tissues and 12 genes expressed in those tissues, during Theiler Stage 7. The corresponding formal context and concept lattice are depicted in figure 6.2.

	Otx2	Fgf8	Fgf4	Zic3	Zic2	Fgf5	Smad3	Bmp4	Sox17	Cited2	Otx1	T
EMAP:62	X	X	X	X	X	X						
EMAP:67					X		X	X				
EMAP:72									X			
EMAP:61	X	X								X		
EMAP:57					X						X	X

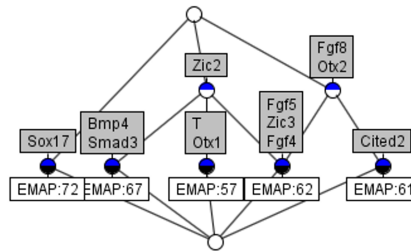


Figure 6.2: The formal context and concept lattice of the data in table 6.1.

An interesting observation is that the SPARQL query does not retrieve all tissues of Theiler Stage 7, but rather only the Theiler Stage 7 tissues where a gene is detected. Thus, the objects (tissues) are restricted, in that

?obj	?att
EMAP:62	Otx2
EMAP:62	Fgf8
EMAP:62	Fgf4
EMAP:62	Zic3
EMAP:62	Zic2
EMAP:62	Fgf5
EMAP:67	Zic2
EMAP:67	Smad3
EMAP:67	Bmp4
EMAP:72	Sox17
EMAP:61	Otx2
EMAP:61	Fgf8
EMAP:61	Cited2
EMAP:57	Otx1
EMAP:57	Zic2
EMAP:57	T

Table 6.1: Genes detected in tissues of Theiler Stage 7. Adapted from [27].

we cannot ascertain the tissues where genes are not detected during Theiler Stage 7. Similarly, not all genes of the tissues associated with Theiler Stage 7 are returned. Thus, the attributes (genes) are restricted, in that we cannot ascertain the genes that do not belong to tissues detected in Theiler Stage 7. As both objects and attributes are restricted, this results in the formal context not having any empty rows or columns.

Removing the restriction on attributes is possible, but would result in a hard to read, or visualize, formal context: the dataset contains more than 7000 genes, which would produce a formal attribute for each of them. However, removing the restriction on objects is reasonable, as there are only 16 tissues associated with Theiler Stage 7. This is possible by utilizing either the `OPTIONAL` clause or the `UNION` operator of SPARQL. The object-unrestricted variant of the query is shown in figure 6.3 and its results are shown in table 6.2. The corresponding formal context and concept lattice are depicted in figure 6.4. Notice how, since the query was converted to be object-unrestricted, the number of objects (tissues) has increased, but the number of attributes (genes) has not. It is

now straightforward to see which tissues exist in Theiler Stage 7, regardless of whether they have a gene detected in this Theiler stage.

```

1  SELECT DISTINCT ?obj ?att WHERE
2  {
3    ?x1 rdf:type :Tissue ; rdfs:label ?o1 .
4    ?x1 :has_theiler_stage :theiler_stage_TS07 .
5    OPTIONAL
6    {
7      ?x2 rdf:type :Gene ; rdfs:label ?a1 .
8      ?ta :in_tissue ?x1 ;
9          :has_involved_gene ?x2 ;
10         :has_strength :level_detected_derived .
11    }
12  }
13  ORDER BY ?obj ?att

```

Figure 6.3: Object-unrestricted variant of the SPARQL query in figure 6.1. Adapted from [27].

	Smad3	Fgf4	Gja1	Zic3	Foxa2	Eomes	Smad2	Bmp4	Etv5
EMAP:53	X								
EMAP:43		X	X	X					
EMAP:50									
EMAP:52			X		X				
EMAP:45				X		X	X	X	
EMAP:42							X		X
EMAP:25772									
EMAP:44									
EMAP:46									
EMAP:47									
EMAP:48									
EMAP:49									
EMAP:51									
EMAP:54									
EMAP:55									
EMAP:56									

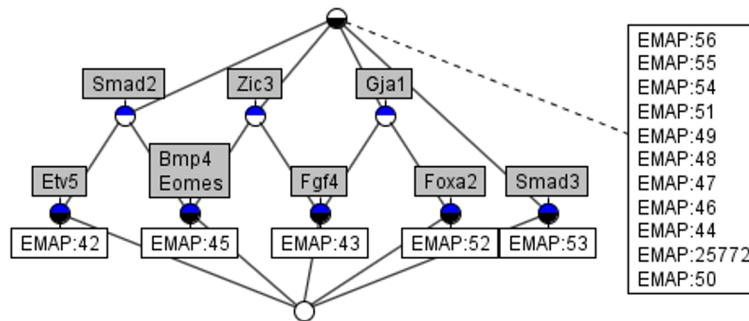


Figure 6.4: The formal context and concept lattice of the object-unrestricted data in table 6.2.

?obj	?att
EMAP:53	Smad3
EMAP:43	Fgf4
EMAP:43	Gja1
EMAP:43	Zic3
EMAP:50	
EMAP:52	Foxa2
EMAP:52	Gja1
EMAP:45	Eomes
EMAP:45	Zic3
EMAP:45	Smad2
EMAP:45	Bmp4
EMAP:42	Etv5
EMAP:42	Smad2
EMAP:25772	
EMAP:44	
EMAP:46	
EMAP:47	
EMAP:48	
EMAP:49	
EMAP:51	
EMAP:54	
EMAP:55	
EMAP:56	

Table 6.2: Object-unrestricted variant of the results in table 6.1. Adapted from [27].

6.1.4 Summary

Dau’s CUBIST scalingless approach allows for flexible SPARQL queries to be written, when querying the underlying RDF triple store, and generating scalingless formal contexts out of the returned data. It provides the ability to restrict the analysis to objects or attributes of interest. However, a good level of understanding of SPARQL and RDF triples is required to utilize this approach.

6.2 CUBIST Scaleful Approach

6.2.1 Introduction

Orphanides’s CUBIST Scaleful Approach [28] extends Dau’s CUBIST scalingless approach of creating formal contexts out of RDF triple stores using SPARQL, by allowing objects and attributes returned from a SPARQL query to be interchangeable, and by providing conceptual scaling capabilities (refer to section 5.4.4 for an explanation of the approach). As data of the EMAGE case study is categorical in nature, this experiment uses the job vacancies RDF dataset described in section 5.4.4 instead, as it also contains continuous attributes which allow for the demonstration of its conceptual scaling capabilities. However, for all intents and purposes, the experiment of subsection 6.1.3 will yield identical results, when performed using the current approach.

6.2.2 Utilized Applications

The software applications used in this approach are:

- SPARQL2FCA, which creates scaleful formal contexts out of SPARQL queries.
- OWLIM, an RDF triple store.
- ConExp, a concept lattice visualizer.

6.2.3 Experimental Results

Let us assume we want to investigate which advertisers post the most job vacancies across various job boards. To do so, we need to gather all advertisers, and count the vacancies they have posted against each job board. The SPARQL query needed to answer this question is shown in figure 6.5.

The results of the query and its corresponding unscaled formal context are shown in figure 6.6. In the unscaled formal context, advertisers are used as objects, and job boards are used as attributes. Advertiser 0 (‘ADV_O’) seems

6.2 CUBIST Scaleful Approach

to be the most popular vacancy poster, with 107247 vacancies posted on job board 163 ('JB_163') alone.

```

1  SELECT (?discname AS ?Advertiser)
2         (?jbname AS ?Jobboard)
3         (COUNT(?vac) AS ?NumOfVacancies)
4  WHERE
5  {
6    ?vac joci:hasDiscipline ?disc .
7    ?vac joci:advertisedOn ?jb .
8    ?jb joci:name ?jbname .
9    ?disc joci:name ?discname .
10 }
11 GROUP BY ?discname ?jbname
12 ORDER BY DESC(?NumOfVacancies)

```

Figure 6.5: SPARQL query of the number of vacancies posted by advertisers, per job board.

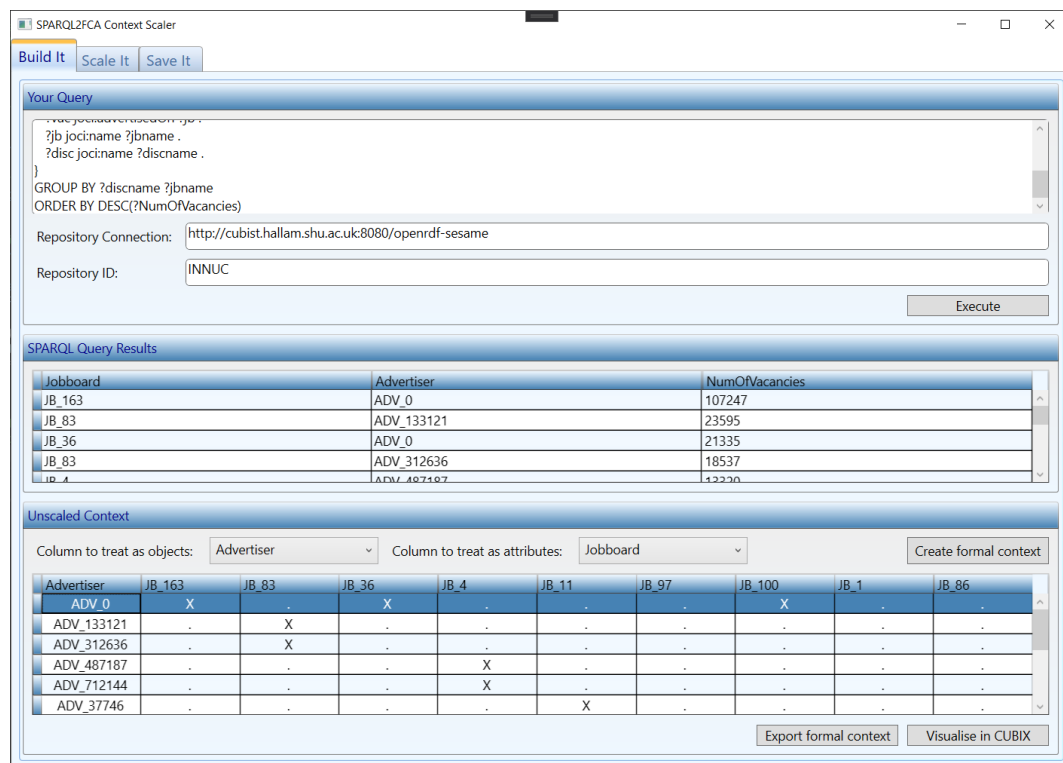


Figure 6.6: Transformation of a SPARQL query result to an unscaled formal context in SPARQL2FCA.

6.2 CUBIST Scaleful Approach

Figure 6.7 shows how SPARQL2FCA’s criteria of which columns to treat as objects and which columns to treat as attributes is not driven by the SPARQL query, but by the end-user. Here, an unscaled formal context is created where the number of vacancies are the objects, and the job boards are the attributes. However, the lack of conceptual scaling is evident, as if vacancy counts were treated as attributes rather than objects, then one formal attribute would be created for each vacancy count value.

Unscaled Context

Column to treat as objects: NumOfVacancies Column to treat as attributes: Jobboard Create formal context

NumOfVacancies	JB_163	JB_83	JB_36	JB_4	JB_11	JB_97	JB_100	JB_1	JB_86
107247	X	-	-	-	-	-	-	-	-
23595	-	X	-	-	-	-	-	-	-
21335	-	-	X	-	-	-	-	-	-
18537	-	X	-	-	-	-	-	-	-
13320	-	-	-	X	-	-	-	-	-
10221	-	-	-	X	-	-	-	-	-
7470	-	-	-	-	X	-	-	-	-

Export formal context Visualise in CUBIX

Figure 6.7: Object and attribute selection in SPARQL2FCA.

In figure 6.8, the conceptual scaling capabilities of SPARQL2FCA are demonstrated. By selecting the number of vacancies column to be the only attribute in the formal context, the job boards and advertisers are treated as objects and are concatenated to *Jobboard–Advertiser* pairs. Furthermore, the number of vacancies attribute has been declared as continuous, and equal width has been chosen as the preferred binning technique; therefore, a total of 4 disjoint ranges of equal width have been automatically created by the tool for the number of vacancies column. Notice how the scaled results are essentially the same as their unscaled counterpart, but there are only 4 formal attributes now, as opposed to 32 previously (i.e. one formal attribute for each distinct vacancy count value in our dataset). This results in less-cluttered, manageable formal contexts, which are easier to visualize.

6.3 CUBIST Interactive Approach

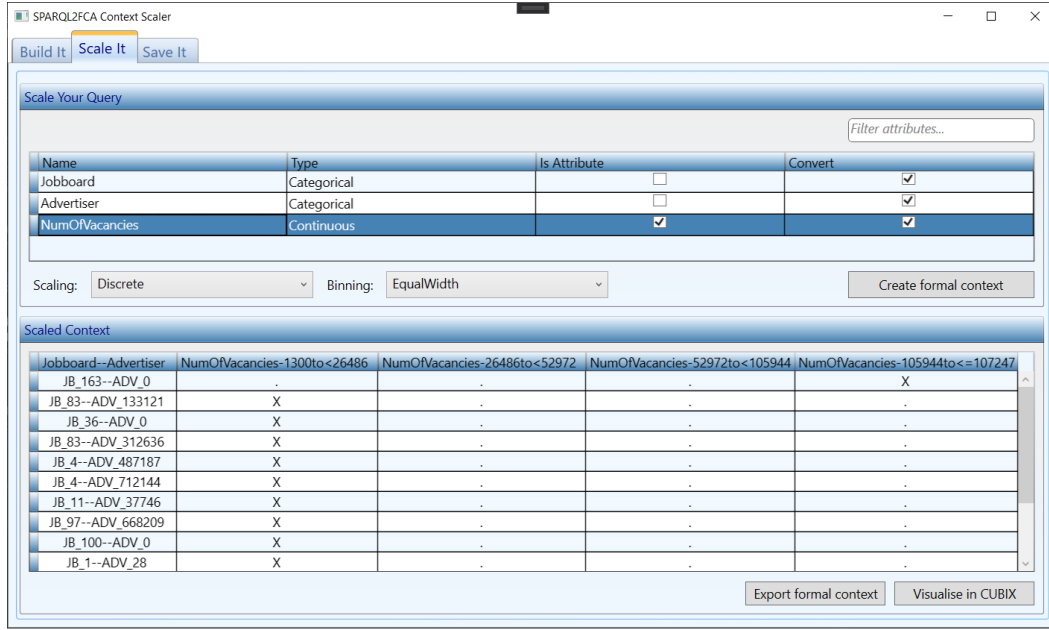


Figure 6.8: Conceptual scaling capabilities in SPARQL2FCA.

6.2.4 Summary

Orphanides's CUBIST scaleful approach complements and extends Dau's CUBIST scalingless approach of creating formal contexts out of SPARQL queries in RDF triple stores. It allows for objects and attributes in a formal context to be interchangeable, and features partial capabilities of conceptual scaling, particularly for continuous attributes. However, a good understanding of SPARQL and RDF triples is also required to utilize this approach.

6.3 CUBIST Interactive Approach

6.3.1 Introduction

The CUBIST interactive approach incorporates the scalingless and scaleful CUBIST approaches in a web application called Cubix. It involves the creation of scalingless and scaleful formal contexts out of a triple store. The end-user is driving the exploration and analysis through direct interaction with the

Cubix frontend (refer to section 5.4.5 for an explanation of the approach). The experiment uses the EMAGE case study as its data source.

6.3.2 Utilized Applications

The software applications used in this approach are:

- Cubix, the web application frontend of the CUBIST project.
- OWLIM, an RDF triple store.

6.3.3 Experimental Results

In this experiment, let us assume we are exploring the EMAGE dataset, and are interested in seeing the tissues in which gene Bmp4 is strongly or moderately expressed. To do so, filters need to be applied on the EMAGE entities. Consequently, the Gene entity is filtered to only include ‘Bmp4’ and the Strength entity is filtered to only include values ‘detected’, ‘strong’ and ‘moderate’. Finally, the entire Tissue entity is selected. In doing so, the underlying RDF triple store is queried to only return data satisfying these conditions (figure 6.9). Similar to other approaches, this is a form of object restriction: the EMAGE dataset is filtered to only include tissues (the objects) where gene Bmp4 (the attribute), is strongly or moderately detected (attribute values).

We can now proceed to the next screen, where we can configure our scaling parameters, to create a scaleful formal context. The available scaling options are categorical (many-valued), boolean (binary), and continuous (numerical). As ‘Gene’ and ‘Strength’ are many-valued attributes, we configure them as ‘Categorical’ attributes (figure 6.10). This will result in a scaleful formal context being created, where one formal attribute will be created for each possible attribute value of ‘Gene’ and ‘Strength’.

The formal context is then passed to the visualization screen and visualized as a concept lattice (figure 6.11). There are a total of 25 tissues in which gene

The screenshot displays the CUBIST Interactive Approach web interface. At the top, there are three tabs: 'Analyze', 'Scaling', and 'Graph Vis'. The 'Analyze' tab is active. On the left, there is a sidebar with various filters. The 'Gene' filter is set to 'Bmp4'. The 'Strength' filter is set to 'detected', 'strong', and 'moderate'. The 'Tissue' filter is set to 'Tissue'. A 'Clear' button is located below the filters. The main area shows a table with columns 'Object: Tissue', 'has symbol', and 'has value'. The table lists 25 tissues, all with 'Bmp4' as the symbol and 'strong' or 'detected' as the value. A 'Refresh' button is located in the top right corner of the table.

Object: Tissue	has symbol	has value
extraembryonic component TS09	Bmp4	strong
infundibular recess of 3rd ventricle TS15	Bmp4	strong
latero-nasal process TS20	Bmp4	strong
medial-nasal process TS20	Bmp4	strong
vibrissa TS23	Bmp4	strong
pineal primordium TS23	Bmp4	strong
cochlea TS23	Bmp4	strong
utricle TS23	Bmp4	strong
eyelid TS23	Bmp4	strong
anterior TS23	Bmp4	strong
external TS23	Bmp4	strong
olfactory TS23	Bmp4	strong
aorta TS23	Bmp4	strong
pulmonary artery TS23	Bmp4	strong
stomach TS23	Bmp4	strong
rectum TS23	Bmp4	strong
midgut TS23	Bmp4	strong
left lung TS23	Bmp4	strong
right lung TS23	Bmp4	strong
mesoderm TS11	Bmp4	detected
allantois TS11	Bmp4	detected
chorion TS11	Bmp4	detected
diencephalon TS15	Bmp4	detected
mesenchyme TS17	Bmp4	detected
extraembryonic ectoderm TS09	Bmp4	detected

Figure 6.9: Filtering the EMAGE dataset for tissues where gene Bmp4 is detected, in Cubix.

Bmp4 is detected. As evidenced by the highlighted concept (node), out of the 25 tissues it is detected, it is strongly detected in 19 of them, with most of them occurring during Theiler Stage 23.

6.3.4 Summary

The CUBIST interactive approach features identical functionality to the CUBIST scaleful approach. However, there is a fundamental difference: knowledge of SPARQL or RDF triple stores is no longer a requirement. Instead, data entities of interest along with filtering criteria are visually selected by the user, which are then automatically converted to SPARQL queries in the backend. However, while the SPARQL queries and the triple store have been hidden away and replaced by an interactive web interface, this approach is still heavily reliant on them.

6.4 FcaBedrock

6.4.1 Introduction

FcaBedrock is a desktop tool which appropriates data for Formal Concept Analysis, developed as part of this thesis. Refer to section 5.5.1 for an explanation of the approach.

This experiment uses multiple datasets to demonstrate the approach. The three CUBIST approaches are heavily reliant on SPARQL and triple stores, and as such can only be used against the EMAGE case study. In the following experiment, various datasets of the UCI ML Repository case study are also examined.

6.4.2 Utilized Applications

The software applications used in this approach are:

- FcaBedrock, a desktop tool which appropriates data for FCA, developed as part of this thesis.
- ConExp, a concept lattice visualizer.

6.4.3 Experimental Results

6.4.3.1 Experiment #1

In our first experiment, we utilize the Agaricus Lepiota dataset of the UCI ML Repository case study (refer to section 5.3.2 for a description of the dataset). Using the metadata autodetection functionality, the metadata of the dataset were automatically created (figure 6.12). The dataset was converted in FcaBedrock in its entirety, i.e. all of the objects, attributes, and attribute values were included in the analysis. This resulted in a formal context having 124 attributes and 220000 concepts; far too many to visualize.

However, let us assume we are interested to explore the relationship between mushroom habitats and population types. We can thus exclude all other

FcaBedrock v3.11.4

Open data set

Define attributes

Output format

Convert

Help

About

Load attributes from bedrock file

Save attributes to bedrock file

Autodetect metadata

Scan progress: 100 %

Position: 21

Name: population

Ignore this attribute: ☐

Description: Default attribute description.

Count of categories: 6

+ -

Input	Output	Filtered
scattered	s	<input type="checkbox"/>
numerous	n	<input type="checkbox"/>
abundant	a	<input type="checkbox"/>
several	v	<input type="checkbox"/>
solitary	y	<input type="checkbox"/>
clustered	c	<input type="checkbox"/>

Id: 13 . stalk-surface-below-ring

Id: 14 . stalk-color-above-ring

Id: 15 . stalk-color-below-ring

Id: 16 . veil-type

Id: 17 . veil-color

Id: 18 . ring-number

Id: 19 . ring-type

Id: 20 . spore-print-color

Id: 21 . population

Id: 22 . habitat

Figure 6.12: Autodetecting the metadata of the Agaricus Lepiota dataset, in FcaBedrock.

attributes from the analysis, and only focus on the two attributes in question. This is easily doable by choosing the ‘Ignore this attribute’ option for every attribute we wish to exclude from the analysis (figure 6.13). This attribute restriction capability reduced the formal attributes to a total of 13: 7 for the categories of ‘habitat’, and 6 for the categories of ‘population’. The resulting concept lattice can be seen in figure 6.14. It can be seen that most clustered mushrooms are found, in similar numbers, in habitats such as woods, leaves and waste grounds, but not in other habitats. Solitary mushrooms are mostly found in woods, but can be occasionally found elsewhere.

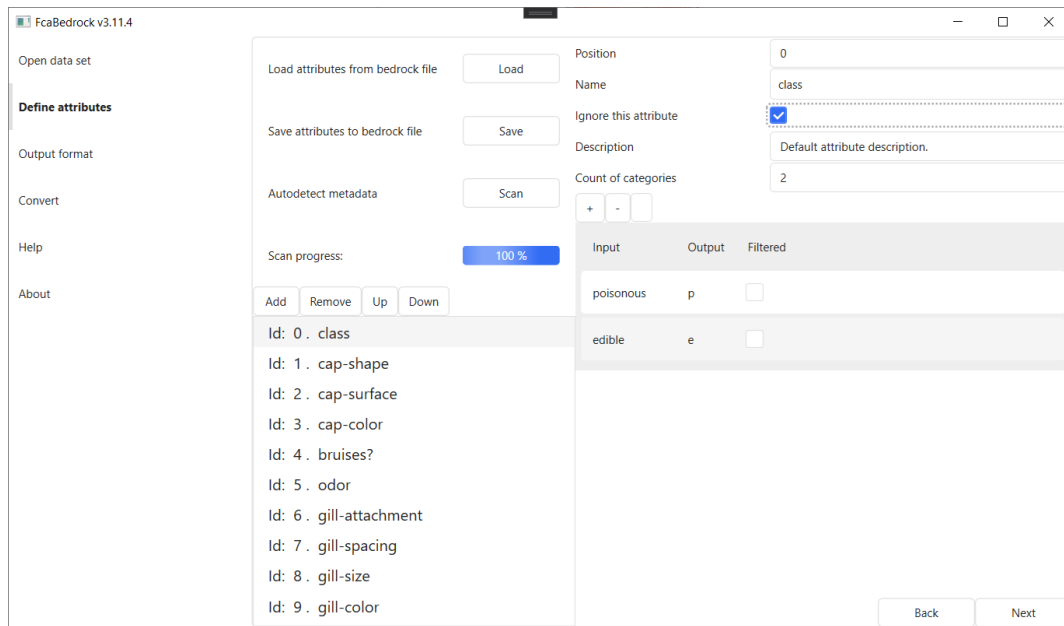


Figure 6.13: Ignoring (excluding) the ‘class’ attribute in FcaBedrock.

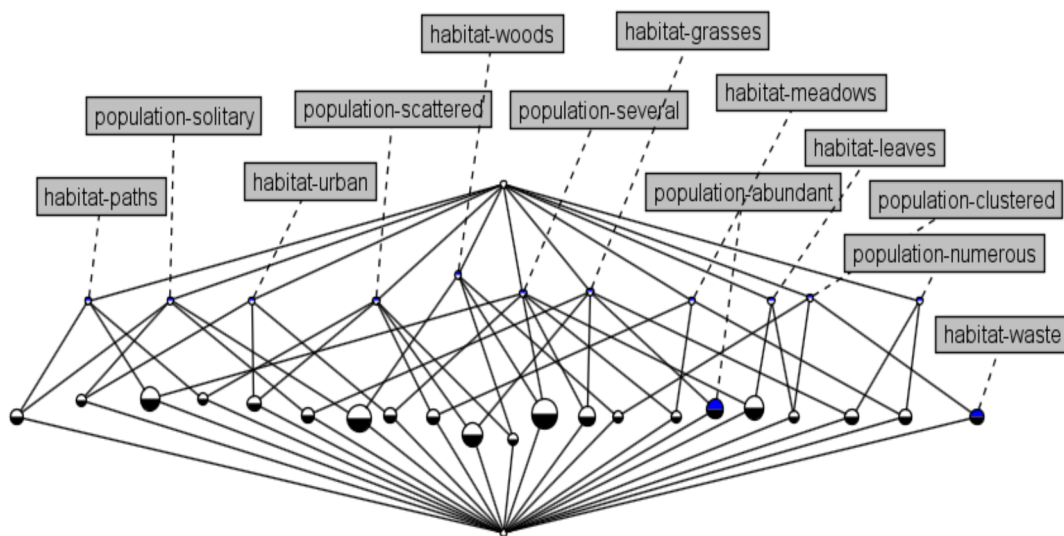


Figure 6.14: A mushroom habitat/population lattice.

6.4.3.2 Experiment #2

In this experiment, we carry out an analysis using the Adult dataset of the UCI ML Repository case study (refer to section 5.3.1 for a description of the dataset). The dataset contains US income census data of 32561 adults with attributes such as age, gender and education. The input file is in three column

CSV format, containing 488415 triples. The formal context produced, when all of the suitable attributes are converted, contains over 100000 concepts.

However, let us say we are interested in investigating the relationship between pay and gender, in adults who have had a higher education. To carry out this analysis, we employ both the attribute and object restriction capabilities of FcaBedrock, by only converting the ‘sex’, ‘class’ and ‘education’ attributes, and by restricting the ‘education’ attribute to ‘Bachelors’, ‘Masters’ and ‘Doctorate’. This is achieved by checking the ‘Filtered’ checkbox for every attribute value we want to include in the analysis (figure 6.15). This results in a formal context with 7491 objects (out of the initial 32561), 7 formal attributes (out of the initial 22161), and 37 concepts (out of the initial 100000+). The resulting concept lattice can be seen in figure 6.16.

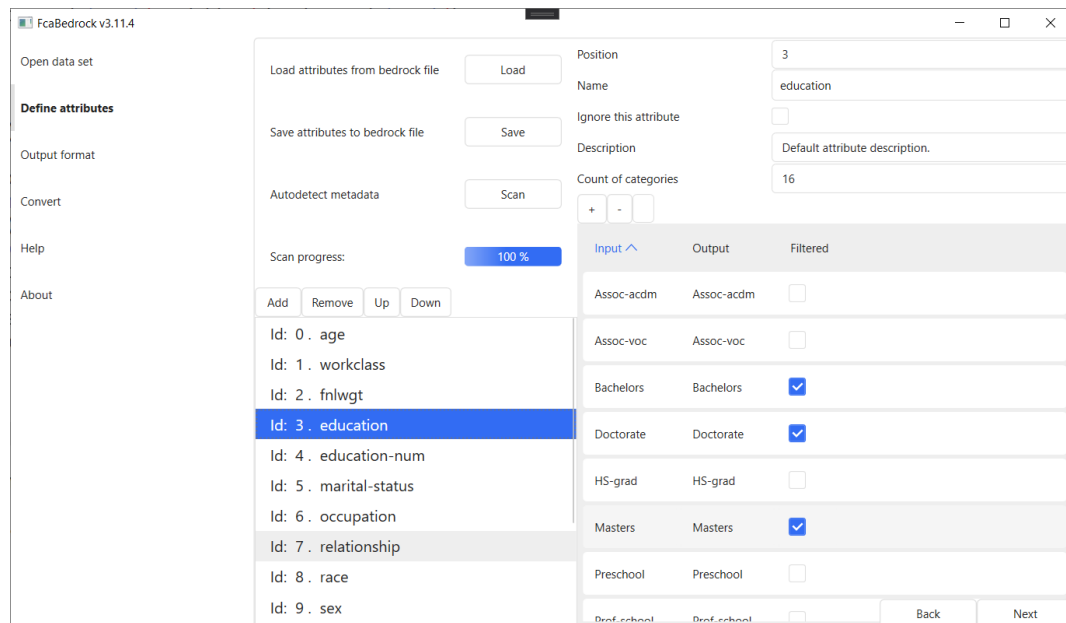


Figure 6.15: Restricting attribute values of ‘education’, in FcaBedrock.

While slightly cluttered, the concept lattice is still readable. The lattice seems to confirm the presence of a ‘gender gap’. For all degrees, males seem to earn more than \$50k in a higher proportion than females. This is particularly evident at Master’s level, where 65% of males and only 33% of females earn more than \$50k.

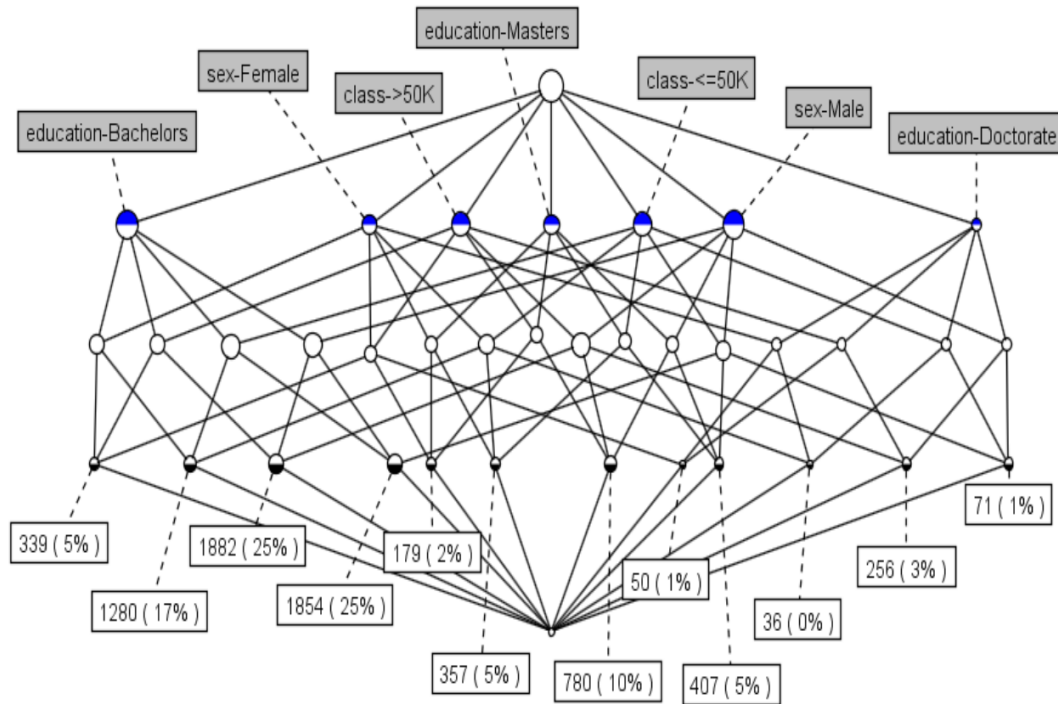


Figure 6.16: A lattice of pay per gender, per higher education, in the Adult dataset.

Finally, let us say we are interested to see how age correlates with pay. The age attribute contains 76 possible values; it does not seem reasonable to investigate each age value individually. Instead, we let FcaBedrock automatically create ranges for us. By selecting equal frequency as a binning strategy and inputting 4 as the desired number of bins, FcaBedrock suggests the following ranges, in the form of *half-open intervals* [74, p. 31]: $(16, 34]$ (i.e. $17 < 35$), $(34, 52]$ (i.e. $35 < 53$), $(52, 70]$ (i.e. $53 < 71$) and $(70, 87]$ (i.e. $71 < 88$) (figure 6.17). Finally, we choose to only include the ‘age’ and ‘class’ attributes in our analysis.

Figure 6.18 shows the resulting concept lattice. The age attribute has been progressively (i.e. ordinal) scaled so that the natural age hierarchy can be portrayed in the lattice (refer to section 4.8 for a reminder). We can see that the further we travel down the lattice, the more the population decreases. This is expected, as the age groups become narrower on the way down, and wider on the way up. Figure 6.19 displays the same data, but this time the age attribute

The screenshot shows the FcaBedrock v3.11.4 application window. The sidebar on the left contains links for 'Open data set', 'Define attributes', 'Output format', 'Convert', 'Help', and 'About'. The main panel is divided into several sections:

- Load attributes from bedrock file:** Includes a 'Load' button.
- Save attributes to bedrock file:** Includes a 'Save' button.
- Scan data set:** Includes a 'Scan' button.
- Scan progress:** A progress bar showing 100% completion.
- Position:** A dropdown menu set to '0'.
- Name:** A text field containing 'age'.
- Ignore this attribute:** An unchecked checkbox.
- Description:** A text field containing 'Numeric attribute description.'
- Binning strategy:** A dropdown menu set to 'Equal Frequency'.
- Count of bins:** A text field set to '4'.
- Apply binning:** A button.
- Progressive:** A checked checkbox.
- Count of categories:** A text field set to '4'.
- Buttons:** 'Add', 'Remove', 'Up', and 'Down' buttons are located above a list of attributes.

The list of attributes includes:

- Id: 0 . age (highlighted in blue)
- Id: 1 . workclass
- Id: 2 . fnlwgt
- Id: 3 . education
- Id: 4 . education-num
- Id: 5 . marital-status
- Id: 6 . occupation
- Id: 7 . relationship
- Id: 8 . race
- Id: 9 . sex

Below the attribute list, a table displays the binning results for the 'age' attribute:

LE	From	To	RE	Output	Ignore
(16	34]	(16;34]	<input type="checkbox"/>
(34	52]	(34;52]	<input type="checkbox"/>
(52	70]	(52;70]	<input type="checkbox"/>
(70	87]	(70;87]	<input type="checkbox"/>

At the bottom right of the main panel, there are 'Back' and 'Next' buttons.

Figure 6.17: Automatic creation of equal frequency bins for ‘age’, in FcaBedrock.

has been scaled discretely; that is, the age ranges are disjoint. An inspection of the lattice reveals that 76% of adults make less than \$50k, with 39% of them belonging to the youngest age group. Only 24% of adults make more than \$50k, with the most belonging to the 35to<53 age group. Only 2% of the population is over 70, and only 86 adults of that age group make more than \$50k. This is interesting, but at the same time not surprising, as most adults in that demographic are usually retired by that age.

6.4.3.3 Experiment #3

In the third and final experiment, the EMAGE case study is used as a data source. FcaBedrock cannot read data directly from a triple store, so the EMAGE data have been exported from the triple store as a three column CSV file. The exported dataset contains 732681 triples.

Similarly to previous experiments, let us say we are interested in seeing which tissues the gene Bmp4 is strongly detected in, during Theiler Stages 18, 19 and 20. After autodetecting the metadata of the dataset in FcaBedrock, we restrict the ‘Gene’ attribute to the ‘Bmp4’ gene, the ‘Strength’ attribute to ‘strong’ and ‘detected’, and the ‘Theiler_Stage’ attribute to ‘strong’ and ‘detected’ (figure 6.20). We exclude the ‘EMAP’ attribute from the analysis, to help narrow the results down. Once again, the object restriction and attribute restriction functionalities are employed to answer our query. Finally, the formal context is created and visualized as a concept lattice in figure 6.21. The highlighted nodes in the lattice show that gene Bmp4 is only strongly detected in two tissues, and when it is strongly detected, it is only detected during Theiler Stage 20.

The screenshot shows the FcaBedrock v3.11.4 application window. On the left, a sidebar contains links: 'Open data set', 'Define attributes' (highlighted), 'Output format', 'Convert', 'Help', and 'About'. The main area is divided into several sections. The 'Define attributes' section has buttons for 'Load attributes from bedrock file', 'Save attributes to bedrock file', and 'Autodetect metadata'. Below these is a 'Scan progress' bar at 100% and a list of attributes: 'Id: 0. Gene' (highlighted), 'Id: 1. Theiler_Stage', 'Id: 2. EMAP', 'Id: 3. Tissue', and 'Id: 4. Strength'. To the right, there are input fields for 'Position' (0), 'Name' (Gene), 'Ignore this attribute' (checkbox), 'Description' (Default attribute description), and 'Count of categories' (6773). Below these is a table with three columns: 'Input', 'Output', and 'Filtered'. The table contains rows for 'Bmf', 'Bmi1', 'Bmp10', 'Bmp2', 'Bmp3', 'Bmp4', 'Bmp5', and 'Bmp6'. The 'Bmp4' row has a checked checkbox in the 'Filtered' column. At the bottom right are 'Back' and 'Next' buttons.

Input	Output	Filtered
Bmf	Bmf	<input type="checkbox"/>
Bmi1	Bmi1	<input type="checkbox"/>
Bmp10	Bmp10	<input type="checkbox"/>
Bmp2	Bmp2	<input type="checkbox"/>
Bmp3	Bmp3	<input type="checkbox"/>
Bmp4	Bmp4	<input checked="" type="checkbox"/>
Bmp5	Bmp5	<input type="checkbox"/>
Bmp6	Bmp6	<input type="checkbox"/>

Figure 6.20: Restricting ‘Gene’ to ‘Bmp4’, in FcaBedrock.

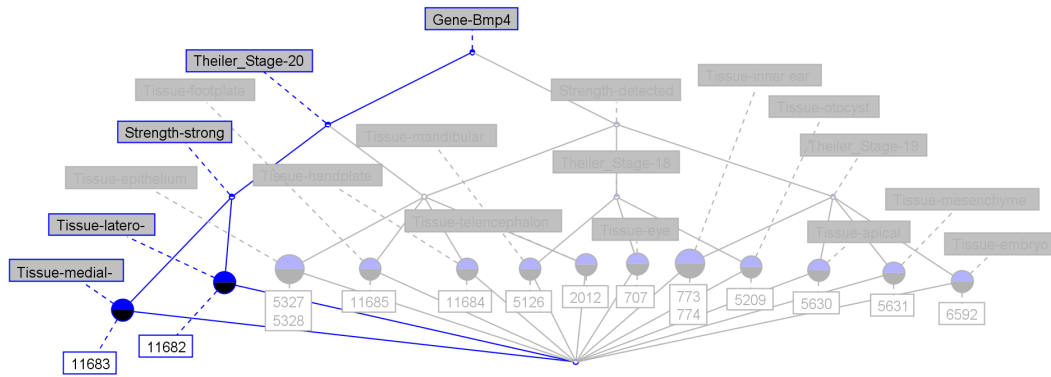


Figure 6.21: A concept lattice of Bmp4's moderate or strong detection in tissues.

6.4.4 Summary

FcaBedrock employs guided automation to appropriate data for FCA. During the conversion process, the user is in full control of the conversion process. The autodetection of metadata for each dataset is a distinctive feature, as no schema needs to be defined about the underlying data. It features object restriction, attribute restriction and conceptual scaling capabilities. It is the only approach able to appropriate data for FCA from both the EMAGE database and UCI ML Repository case studies, provided that the datasets are exported as many column or three column CSV files.

Chapter 7

Evaluation of the FCA Data Appropriation Approaches

7.1 Introduction

In this chapter, the different approaches for appropriating data for Formal Concept Analysis from structured sources are evaluated, namely:

- CUBIST's scalingless approach.
- CUBIST's scaleful approach.
- CUBIST's interactive approach.
- FcaBedrock's guided automation approach.

To conduct this evaluation, the DESMET methodology [50] was used in order to determine the most effective approach to appropriate data for FCA in a user-driven, automated setting.

Numerous methodologies and tools exist, to evaluate software. Such methods are described in detail in [39, 51, 50, 42, 61], while published papers in this area of study have been systematically reviewed by Jadhav and Sonar [46]. Furthermore, [46] concludes that a standard of generic software evaluation

7.2 Essential Features for Enabling Data Appropriation for FCA

criteria does not exist. Thus, the DESMET methodology is used to evaluate the FCA data appropriation approaches in this work.

Evaluators or academic researchers can use the DESMET methodology when investigating new methods [50]. DESMET is used to evaluate the FCA data appropriation approaches presented in chapter 6. It identifies nine methods of evaluation, with the three most important evaluation methods being formal experiments, quantitative case studies, and feature analysis evaluations [51, 50]. A feature analysis evaluation, based on the features identified in applications such as [9, 28, 20, 27], and the FCA data appropriation approaches of chapter 6, is carried out in this work.

The feature analysis evaluation ascertained how the FCA approaches, investigated in this work, facilitated the appropriation of data in a user-driven, automated setting. In section 7.2, the essential features for enabling data appropriation for FCA are identified. In section 7.3, the features identified in section 7.2 are evaluated for each approach, after which a ranking is assigned to each approach. The chapter concludes at section 7.4.

7.2 Essential Features for Enabling Data Appropriation for FCA

Let us reiterate the key research question of this thesis, from section 1.1:

“How can data from structured sources, consisting of various data types, be acquired and appropriated for FCA in a semi-automated, user-driven environment?”

The key research question considers an automated, user-driven environment, when FCA data appropriation is conducted using data from structured sources. Thus, applications facilitating FCA data appropriation should employ some degree of automation, always within reasonable limits. The end-user should be capable of driving the appropriation process, when utilizing the automated

7.3 An Assessment of the FCA Data Appropriation Approaches

capabilities and features of such an application, without having expertise in FCA, or in the data retrieval mechanisms of the underlying data source.

Formal contexts of even modest size can generate an exponentially large number of formal concepts (refer to section 4.3 for a reminder). This has been demonstrated in the experiments of chapter 6, where unreadable and unmanageable concept lattices were produced due to the overwhelming amount of formal concepts that had to be visualized. Therefore, object restriction, attribute restriction and conceptual scaling are crucial features, which applications that can effectively appropriate data for FCA, should possess.

Lastly, the approaches employed in such applications should be generalizable, to eliminate the threat of external credibility. External credibility pertains to the confirmability and transferability of findings and conclusions. It is defined by Onwuegbuzie and Leech [63] as “the degree that the findings of a study can be generalized across different populations of persons, settings, contexts, and times”. Therefore, such an application should also be able to work with different datasets and data sources.

7.3 An Assessment of the FCA Data Appropriation Approaches

7.3.1 An Assessment of CUBIST’s Scalingless Approach

Dau’s CUBIST scalingless approach [27] facilitates the creation of unscaled formal contexts out of RDF triple stores. It does not feature any sort of guided automation; the process is manual and iterative. Furthermore, it requires a good level of understanding of SPARQL and RDF triple stores, making its use by non-SPARQL experts impractical. Furthermore, it can cater for all the datasets provided by the use-case partners of the CUBIST project.

The CUBIST scalingless approach supports object restriction and attribute restriction capabilities, by incorporating such restrictions in the SPARQL

7.3 An Assessment of the FCA Data Appropriation Approaches

queries themselves. Again, a good level of understanding of SPARQL is required to achieve this. It does not support conceptual scaling.

Lastly, the CUBIST scalingless approach is generalizable, but limited within the context of RDF triple stores; its heavy reliance on SPARQL and hardcoded object and attribute identifiers in the SPARQL queries, prohibit its use with other structured sources, such as relational databases.

7.3.2 An Assessment of CUBIST’s Scaleful Approach

Orphanides’s [28] CUBIST scaleful approach extends the CUBIST scalingless approach by also providing conceptual scaling capabilities. However, the support for conceptual scaling is limited. For example, it is not possible to set a user-specified number of bins when scaling a continuous attribute; the number of bins is hardcoded to 4. This prohibits the creation of meaningful ranges for an attribute that might require more ranges, or less, for example.

Contrasting to the CUBIST scalingless approach, which does not feature guided automation, the CUBIST scaleful approach features partial automation, by allowing the end-user to choose whether an unscaled or scaled formal context should be created, and choosing binning techniques for continuous attributes.

However, the CUBIST scaleful approach suffers from the same drawbacks as the CUBIST scalingless approach; the analyses are driven by the SPARQL queries, rather than the end-user, and SPARQL expertise is required.

Lastly, as with the CUBIST scalingless approach, this approach is generalizable, but limited within the context of RDF triple stores. Without major modifications, it is not possible to utilize this approach with other structured sources, such as relational databases.

7.3.3 An Assessment of CUBIST’s Interactive Approach

The CUBIST interactive approach [27] incorporated the scalingless and scaleful CUBIST approaches in a web application called Cubix [60], CUBIST’s state-of-the-art visualization tool.

7.3 An Assessment of the FCA Data Appropriation Approaches

Contrasting to the previous two CUBIST approaches, knowledge of SPARQL or RDF triple stores is not necessary. Rather, the approach is an interactive one, allowing the end-user to explore and analyze the data, through a process of guided automation.

The CUBIST interactive approach supports object restriction and attribute restriction, by allowing the user to interactively select which attributes to exclude from the analysis, or restrict the analysis to specific attribute values. Furthermore, the support for conceptual scaling is limited. For example, it is not possible to set a user-specified number of bins when scaling a continuous attribute; the number of bins is, again, hardcoded to 4.

Finally, the CUBIST interactive approach is only generalizable within the context of the use cases of the CUBIST use case partners. It is theoretically possible to utilize this approach with other RDF data sources, but this has not been proven. Lastly, it is not possible to utilize this approach with other structured sources.

7.3.4 An Assessment of FcaBedrock’s Guided Automation Approach

FcaBedrock¹ is an OSS tool for creating formal contexts for FCA, placing sole emphasis on the FCA data appropriation process [9, 10].

FcaBedrock abstracts the issue of support for various structured data sources away, by utilizing many column CSV and three column CSV files as its input data source. As a result, no knowledge or expertise of the underlying data source is required. CSV is a popular, traditional export option, which is fully supported in most popular relational databases, vertical databases, and RDF triple stores. For this reason, FcaBedrock was the only approach which managed to appropriate data for FCA using data from both the EMAGE database and UCI ML Repository case studies.

¹<https://github.com/trashr0x/fcabedrock>

7.3 An Assessment of the FCA Data Appropriation Approaches

Through its metadata autodetection functionality, the end-user does not need to have an understanding of the underlying schema of the data source. The metadata are reusable and can be shared with other users of the tool, thus allowing subsequent analyses to take place, and different interpretations to be made.

FcaBedrock fully supports object restriction, attribute restriction, and conceptual scaling. The user is able to choose how each attribute should be converted, and how each of its attribute values should be treated. It supports both discrete and hierarchical scaling, and currently supports three binning strategies: equal frequency binning, equal width binning, and free binning. In free binning, assuming n attribute values, n bins are created, each representing a distinct attribute value. As opposed to the other approaches, where all binning strategies are hardcoded to produce a maximum of 4 bins, in FcaBedrock the user is able to specify the number of bins. Furthermore, it is the only approach which allows exporting a formal context in both the Burmeister (.cxt) and FIMI (.dat) file formats.

Moreover, FcaBedrock is the only software evaluated in this chapter that is actively maintained, is open source, and freely available as a standalone application.

Finally, FcaBedrock is generalizable, provided that data from any structured data source are extracted as many column CSV or three column CSV files. This eliminates the problem of hardcoding solutions, and making them tightly bound to the underlying structured data source they aim to investigate.

7.3.5 Overall Ranking and Feature Overview

Table 7.1 provides a summary of the essential features supported by each FCA data appropriation approach. Instead of inspecting the table and manually assigning a rank to each approach, we will use FCA to aid us in the decision making process. Thus, we convert the table into a CSV file and load it in FcaBedrock. We let FcaBedrock autodetect the metadata of the dataset and

7.3 An Assessment of the FCA Data Appropriation Approaches

	CUBIST Scalingless Approach	CUBIST Scaleful Approach	CUBIST Interactive Approach	FcaBedrock
FCA data appropriation	Yes	Yes	Yes	Yes
FCA or other expertise required	Yes	Yes	No	No
Data from multiple structured data sources	No	No	No	Yes
Object restriction	Yes	Yes	Yes	Yes
Attribute restriction	Yes	Yes	Yes	Yes
Conceptual scaling	No	Partial	Partial	Yes
User-driven	No	No	Yes	Yes
Guided automation	No	Partial	Yes	Yes
Generalizable	No	No	No	Yes

Table 7.1: Essential features support of each FCA data appropriation approach.

7.3 An Assessment of the FCA Data Appropriation Approaches

Figure 7.1: Converting the feature overview per FCA approach to a formal context, in FcaBedrock.

treat all attributes as categorical (figure 7.1). We convert the dataset to a formal context and visualize it in ConExp. The resulting concept lattice is depicted in figure 7.2.

The concept lattice in figure 7.2 provides a hierarchical overview of how each FCA data appropriation approach fares against the essential features list identified in section 7.2. It is interesting to notice how ‘negative’ features are displayed on the left hand side of the lattice, while the ‘positive’ features are displayed on the right hand side of the lattice, with the partially supported features displayed in the middle. A quick inspection of the lattice demonstrates how FcaBedrock is the clear winner, featuring all the possible ‘positive’ features in the lattice. The CUBIST interactive approach comes second due to its partial conceptual scaling support, its lack of support for multiple structured data sources, and its non-generalizability for data other than RDF. The CUBIST scaleful approach comes third, due its partial automation and conceptual scaling capabilities. Finally, the CUBIST scalingless approach comes last, due to it only featuring the ‘negative’ features in the lattice.

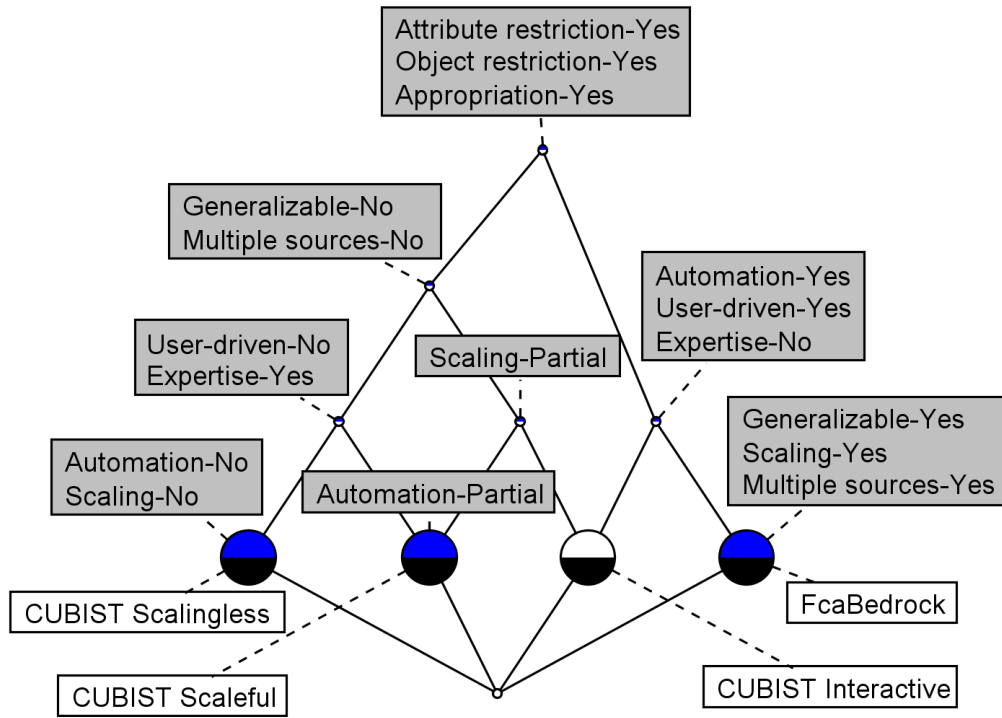


Figure 7.2: Converting the feature overview per FCA approach to a concept lattice, in ConExp.

Consequently, a ranking can now be assigned to each approach:

- 1st place: FcaBedrock.
- 2nd place: CUBIST interactive approach.
- 3rd place: CUBIST scaleful approach.
- 4th place: CUBIST scalingless approach.

7.4 Concluding Summary

In this chapter, the identified FCA data appropriation approaches were evaluated using the DESMET methodology. An essential feature list was identified, and a feature analysis evaluation was conducted, to ascertain how the FCA approaches, investigated in this work, facilitated the appropriation of data in a user-driven, automated setting.

FcaBedrock was assessed as being the most appropriate approach for FCA data appropriation as, out of the four approaches, it possessed the most essential features identified during the feature analysis evaluation. The CUBIST interactive approach and the CUBIST scaleful approach were assessed as being the 2nd and 3rd most appropriate approaches, respectively. CUBIST's scalingless approach was assessed as being the least appropriate approach.

Chapter 8

Conclusion and Future Work

8.1 A Review of the Chapters in this Thesis

Chapter 1 presented the research rationale, the research objectives, and key research question of the thesis.

Chapter 2 presented the research methodology used to conduct the research, in particular the multiple case study research methodology. The e-Mouse Atlas of Gene Expression (EMAGE) database and the UCI Machine Learning (ML) Repository were used as the two critical case studies for this research, providing the real-life context for the investigation and evaluation of data appropriation from structured sources for Formal Concept Analysis (FCA).

Chapter 3 provided an overview of structured data, and the various data types usually encountered in such data. The overview was provided to familiarize the reader with those concepts, as FCA data appropriation involves the retrieval of such data from structured data sources.

Chapter 4 presented Formal Concept Analysis, and explained how FCA data appropriation has as a prerequisite the conversion of data from structured sources into formal contexts, and subsequently concept lattices.

Chapter 5 investigated existing and new FCA data appropriation approaches. Data from the EMAGE database and the UCI Machine Learning Repository were used to demonstrate the approaches. The unsuitable approaches were

identified, and an explanation was provided on why they were not further evaluated in this work. The remaining approaches were experimented with in chapter 6.

Chapter 6 outlined the empirical experiments carried out for each of the approaches, identified as suitable in chapter 5, to assess their suitability as FCA data appropriation approaches. Datasets from the two critical case studies were used. The experiments carried out were used to evaluate the approaches in chapter 7.

In chapter 7, the approaches were evaluated using the DESMET research methodology. An essential feature list was created, by identifying which features an ideal FCA data appropriation approach should possess, using DESMET's feature analysis evaluation method. The approaches were evaluated against the essential feature list, and a ranking was assigned to each approach.

This chapter provided a review of the chapters in this thesis. In the next sections, an explanation is given into how the results of the research addressed the research objectives of this work. Furthermore, the contribution to knowledge that the research has produced is provided. Finally, the chapter and thesis conclude with proposals for future work.

8.2 Addressing of Research Objectives

This work addressed and accomplished the research objectives, stated in section 1.2, as follows:

Objective #1: To understand FCA data appropriation issues and how they are dealt with. This work used existing literature to understand and identify issues related to FCA data appropriation. In chapter 1, the lack of automation and diverse interpretation of data, leading to inconsistent and incomparable analyses, were identified. In chapter 3, the need of distinguishing between data types, during the conversion process, was stated. Chapter 4 demonstrated how data booleanization, data discretization and conceptual scaling are necessary to conduct meaningful FCA analyses. Finally, chapter 5 identified how knowledge

of database query languages, to retrieve data from structured data sources, is more than often a prerequisite and can be problematic, only allowing ‘experts’ to conduct FCA. Chapter 5 also demonstrated various approaches to address these issues, such as employing guided automation, making the approach generalizable, and requiring no expertise of the underlying schema of the structured data source, in order to access and appropriate the data for FCA.

Objectives #2 & #3: To build on existing FCA data appropriation approaches and to develop semi-automated, user-driven, FCA data appropriation techniques. This work investigated existing FCA data appropriation approaches in chapter 5. It demonstrated the advantages and disadvantages of each approach and then explained how FcaBedrock improves upon those approaches, by employing a novel approach of guided automation.

Objective #4: To apply existing and new FCA data appropriation approaches to the UCI ML Repository and EMAGE case studies. This work applied the existing and new FCA data appropriation approaches to the EMAGE database and UCI ML Repository case studies. In chapter 5, examples from the two case studies were created, to demonstrate how each approach works in practice. In chapter 6, experiments were carried out for each approach, to assess their usefulness as FCA data appropriation approaches, using data from the EMAGE database and UCI ML Repository case studies. The results of these experiments were then used to evaluate the approaches in chapter 7.

Objective #5: To compare and evaluate the usefulness and effectiveness of the different FCA data appropriation approaches. This work compared and evaluated the effectiveness of the different FCA data appropriation approaches in chapters 6 and 7. Specifically, in chapter 6, experiments were conducted for each approach, in order to assess their usefulness as FCA data appropriation techniques. Chapter 7 evaluated those approaches, using DESMET as the research methodology. Through a feature analysis evaluation, FcaBedrock emerged as the most suitable approach for appropriating data for FCA, in a user-driven, automated setting.

8.3 Research Knowledge Contribution

The key contributions to knowledge, of this research, are as follows:

Knowledge Contribution #1: This work has expanded the available literature on appropriating data from structured sources for Formal Concept Analysis. Chapter 1 remarked that there is less work describing how techniques for appropriating data for FCA, such as data discretization and data booleanization, can be applied in an automated, reproducible way [3]. Furthermore, the available literature on appropriating data from structured sources for Formal Concept Analysis is limited. This thesis, including publications derived from this work such as [9, 10, 8, 14], have extended the published body of work on this subject. A comprehensive list of publications derived from this work can be found in Appendix A.

Knowledge Contribution #2: This work compared the different approaches used to appropriate data for Formal Concept Analysis. The approaches employed to appropriate data for FCA, in chapter 5, were compared in chapter 7. This was achieved by comparing their effectiveness and applicability in appropriating data for Formal Concept Analysis. These comparisons can act as a guide, or reference, for FCA analysts and researchers looking to adopt one or more of these approaches in their work.

Knowledge Contribution #3: This work evaluated the different approaches used to appropriate data for Formal Concept Analysis. This work outlined in detail the different approaches used to appropriate data for Formal Concept Analysis, in chapter 5. In chapter 6, the identified approaches were thoroughly investigated, with experiments carried out using data from the EMAGE database and UCI Machine Learning Repository, representing the two critical case studies of this thesis. This facilitated the evaluation of these approaches in chapter 7, ascertaining their effectiveness and suitability as FCA data appropriation techniques. This evaluation makes it possible for new or existing FCA researchers, entering this field of study, to comprehend what has been accomplished thus far.

8.4 Conclusion

The aim of this research was to establish how can data from structured sources, consisting of various data types, be acquired and appropriated for Formal Concept Analysis (FCA) in a semi-automated, user-driven environment.

The research in this thesis leads to the conclusion that the approach of guided automation, employed by FcaBedrock, addresses this question. It has been demonstrated how approaches which do not employ an automated, user-driven workflow, will typically require a significant level of expertise of the structured sources that data are appropriated from, and will usually lack generalizability. Through user guidance and graphical interaction, guided automation is made possible, thus eliminating these concerns from the FCA data appropriation process.

8.5 Future Work

FcaBedrock employs a process of guided automation to appropriate data from structured sources for Formal Concept Analysis. It does so by accepting many column CSV or three column CSV files as input. This eliminates the hurdles associated with accessing and querying a structured data source, as evidenced in this thesis. However, alternative ways of retrieving data from these data sources could be explored. For example, using engineering practices inspired from the rise of low-code and no-code development [87, 35, 76, 58], an intuitive wizard could be developed, allowing the FcaBedrock end-user to drag-and-drop a set of tables from a relational database or triple store, by only having to provide database connection credentials. From then on, FcaBedrock can autodetect the metadata of these tables by parsing the database schema, after which the rest of the process remains identical to the current one and ensures that the process of user-driven, guided automation, is maintained.

Another plan is to provide the end-user the ability to apply the well-known idea of minimum support [97] on the formal contexts produced by

FcaBedrock, possibly through the incorporation of In-Close2 [5] in FcaBedrock. We previously described object restriction and attribute restriction as context reduction techniques. Minimum support is, instead, a formal concept reduction technique: by specifying a minimum number of objects and/or attributes a formal concept should contain, formal concepts containing numbers of objects or attributes smaller than the defined minimums are filtered out from the formal context. Thus, FcaBedrock's data appropriation techniques can be complemented by, and extended with, formal context appropriation techniques.

Triadic Concept Analysis (TCA) is another interesting avenue to explore. It is an extension of FCA that was introduced by Lehmann and Wille [57, 88] and incorporates the notion of *conditions*. Conditions are made possible through a *triadic context*, which is a three-dimensional cross table that is defined as a quadruple $\mathbb{T} = (G, M, B, Y)$, where G, M and B are sets and $Y \subseteq G \times M \times B$ is a ternary relation between those three sets. The elements of G, M and B are called objects, attributes, and conditions, respectively. For $g \in G, m \in M$ and $b \in B$, $(g, m, b) \in Y$ is read as: the object g has the attribute m under condition b . Such triadic contexts can be viewed as a family of formal contexts, with each formal context representing one condition, by slicing the triadic context vertically across the conditions [32]. To this end, FcaBedrock could be extended to support TCA by allowing the definitions of conditions by which a formal context would be created for each of the conditions.

References

- [1] Abadi, D. J., Marcus, A., Madden, S. R., and Hollenbach, K. (2009). SW-Store: a vertically partitioned DBMS for Semantic Web data management. In *VLDB*, volume 18, pages 385–406. Springer-Verlag.
- [2] Anderson, B. and Hardin, J. M. (2014). Credit scoring in the age of big data. In *Encyclopedia of Business Analytics and Optimization*.
- [3] Andrews, S. (2009a). Data conversion and interoperability for FCA. In Croitoru, M., Fortin, J., and Jäschke, R., editors, *CS-TIW 2009*, pages 42–49.
- [4] Andrews, S. (2009b). In-Close, a fast algorithm for computing formal concepts. In Rudolph, S., Dau, F., and Kuznetsov, S. O., editors, *ICCS 2009*, volume 843. CEUR WS.
- [5] Andrews, S. (2011). In-Close2, a high performance formal concept miner. In Andrews, S., Polovina, S., Hill, R., and Akhgar, B., editors, *Conceptual Structures for Discovering Knowledge - Proceedings of the 19th International Conference on Conceptual Structures (ICCS)*, pages 50–62. Springer.
- [6] Andrews, S. (2015). A Best-of-Breed approach for designing a fast algorithm for computing fixpoints of Galois Connections. *Information Sciences*, 295(20):633–649.
- [7] Andrews, S. and McLeod, K. (2013). Gene co-expression in mouse embryo tissues. *International Journal of Intelligent Information Technologies (IJIT)*, 9(4):55–68.
- [8] Andrews, S. and Orphanides, C. (2010a). Analysis of large data sets using formal concept lattices. In Kryszkiewicz, M. and Obiedkov, S., editors, *Proceedings of Concept Lattices and their Applications (CLA) 2010*, pages 104–115. University of Sevilla.
- [9] Andrews, S. and Orphanides, C. (2010b). FcaBedrock, a formal context creator. In Croitoru, M., Ferre, S., and Lukose, D., editors, *ICCS 2010, LNCS*, volume 6208/2010 of *LNCS*. Springer.
- [10] Andrews, S. and Orphanides, C. (2010c). Knowledge discovery through creating formal contexts. In Xhafa, F., Demetriadis, S., Caballe, S., and Abraham, A., editors, *Proceedings of the First International Workshop on Computational Intelligence in Networks and Systems (CINS 2010)*, pages 455–460.

-
- [11] Andrews, S. and Orphanides, C. (2012). Knowledge discovery through creating formal contexts. In Li, J., editor, *International Journal of Space-Based and Situated Computing*, volume 2(2), pages 123–138. Inderscience.
- [12] Andrews, S. and Orphanides, C. (2013). Discovering knowledge in data using Formal Concept Analysis. In Bessis, N., editor, *International Journal of Distributed Systems and Technologies (IJDST)*, volume 4(2), pages 31–50. IGI Global.
- [13] Andrews, S., Orphanides, C., and Polovina, S. (2010). Visualising computational intelligence through converting data into formal concepts. In Xhafa, F., Barolli, L., Nishino, H., and Aleksy, M., editors, *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, pages 302–307. IEEE.
- [14] Andrews, S., Orphanides, C., and Polovina, S. (2011). Visualising computational intelligence through converting data into formal concepts. In Bessis, N. and Xhafa, F., editors, *Next Generation Data Technologies for Collective Computational Intelligence*, volume 352 of *Studies in Computational Intelligence*, pages 139–165. Springer.
- [15] Arevalo, G., Berry, A., Huchard, M., Perrot, G., and Sigayret, A. (2007). Performances of Galois Sub-hierarchy-building algorithms. In Kuznetsov, S. O. and Schmidt, S., editors, *ICFCA 2007, LNAI 4390*, pages 166–180.
- [16] Backus, J. W. (1959). The syntax and semantics of the proposed international algebraic language of the Zurich ACM-GAMM Conference. In *Proceedings of the International Conference on Information Processing*, pages 125–132. UNESCO.
- [17] Baldock, R. and Davidson, D. (2008). *Anatomy ontologies for bioinformatics: principles and practise*, pages 249–265. Springer-Verlag.
- [18] Becker, H. S. (1996). The epistemology of qualitative research. In *Ethnography and human development: Context and meaning in social inquiry*, pages 53–71.
- [19] Becker, P. (2006). ToscanaJ user manual. available: http://www.wormuth.info/ICFCA04/ToscanaJ_User_Manual.pdf, 2006.
- [20] Becker, P. and Correia, J. H. (2005). The ToscanaJ suite for implementing Conceptual Information Systems. In Ganter, B., Stumme, G., and Wille, R., editors, *Formal Concept Analysis: Foundations and Applications*, volume 3626 of *LNCS*, pages 324–348. Springer.
- [21] Belohlavek, R. and Trnecka, M. (2013). Basic Level in Formal Concept Analysis: Interesting concepts and psychological ramifications. In Rossi, F., editor, *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence IJCAI 2013*, pages 1233–1239. AAAI Press.
- [22] Birkhoff, G. (1940). Lattice theory. In *American Mathematical Society, 1940*, volume 25.

-
- [23] Boole, G. (1854). An investigation of the laws of thought. pages 265–275. London: Walton & Maberly, 1854.
 - [24] Boulicaut, F. and Besson, J. (2008). Actionability and formal concepts: A data mining perspective. In Medina, R. and Obiedkov, S., editors, *ICFCA 2008*, volume 4933 of *LNCS (LNAI)*, pages 14–31. Springer-Verlag, Berlin/Heidelberg.
 - [25] Burmeister, P. (2000). Conimp - ein programm zur formalen begriffsanalyse. In Stumme, G. and Wille, R., editors, *Begriffliche Wissensverarbeitung. Methoden und Anwendungen*, pages 25–56.
 - [26] Clarke, E. J. and Barton, B. A. (2000). Entropy and mdl discretization of continuous variables for bayesian belief networks. In *International Journal of Intelligent Systems*, volume 15, pages 61–92.
 - [27] Dau, F. (2013). Towards scalingless generation of formal contexts from an ontology in a triple store. *International Journal of Conceptual Structures and Smart Applications (IJCSSA)*, 1(1):18–38.
 - [28] Dau, F. and Andrews, S. (2014). Combining Business Intelligence with Semantic Technologies: The CUBIST project. In *International Conference on Conceptual Structures*, pages 281–286. Springer.
 - [29] Dau, F. and Sertkaya, B. (2011). An extension of ToscanaJ for FCA-based data analysis over triple stores. In *Proceedings of CUBIST Workshop 2011*.
 - [30] Dougherty, J., Kohavi, R., and Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In Prieditis, A. and Russell, S. J., editors, *Work*, pages 194–202. Morgan Kaufmann.
 - [31] Fayyad, U. and Irani, K. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 1022–1027.
 - [32] Felde, M. and Stumme, G. (2021). Triadic exploration and exploration with multiple experts. In *International Conference on Formal Concept Analysis (ICFCA) 2021*, volume 12733 of *Lecture Notes in Computer Science (LNAI)*, pages 175–191.
 - [33] Flyvbjerg, B. (2006). Five misunderstandings about case-study research. In *Qualitative Inquiry*, volume 12, pages 219–245.
 - [34] Frank, A. and Asuncion, A. (2022). UCI Machine Learning Repository: <https://archive.ics.uci.edu/ml>.
 - [35] Fryling, M. (2019). Low code app development. In *Journal of Computing Sciences in Colleges 2019*, volume 34(6).
 - [36] Ganter, B. (1984). Two basic algorithms in concept analysis. FB4-Preprint 831, TH Darmstadt.
 - [37] Ganter, B. and Wille, R. (1999). *Formal Concept Analysis: Mathematical Foundations*. Springer-Verlag, 1st edition.

-
- [38] Goethals, B. and Zaki, M. (2004). Advances in Frequent Itemset Mining Implementations: Report on FIMI'03. In *SIGKDD Explorations Newsletter*, volume 6(1), pages 109–117. ACM New York.
- [39] Grau, G., Carvallo, J. P., Franch, X., and Quer, C. (2004). DesCOTS: a software system for selecting COTS components. In *Euromicro Conference 2004*, pages 118–126.
- [40] Han, J., Cheng, H., Xin, D., and Yan, X. (2007). Frequent pattern mining: current status and future directions. In *Data Mining and Knowledge Discovery*, volume 15, pages 55–86.
- [41] Harris, D. and Harris, S. (2012). *Digital design and computer architecture*. 2nd edition.
- [42] Hlupic, V. and Mann, A. S. (1995). SimSelect: a system for simulation software selection. In *Simulation Conference Proceedings, 1995*, pages 720–727. IEEE.
- [43] Ignatov, D. I. (2017). Introduction to Formal Concept Analysis and its applications in information retrieval and related fields. In *Information Retrieval - 8th Russian Summer School, RuSSIR 2014, Nizhniy Novgorod, Russia*, volume 505, pages 42–141. Springer.
- [44] Ignatov, D. I. and Kuznetsov, S. O. (2009). Frequent itemset mining for clustering near duplicate web documents. In *ICCS 2009*, volume 5662 of *LNCS (LNAI)*, pages 185–200. Springer-Verlag, Berlin/Heidelberg.
- [45] Imberman, S. and Domanski, D. (2001). Finding association rules from quantitative data using data booleanization. In *Proceedings of the Americas' Conference on Information Systems (AMCIS) 2001*.
- [46] Jadhav, A. S. and Sonar, R. M. (2009). Evaluating and selecting software packages: A review. In *Information and software technology*, 51(3), pages 555–563.
- [47] Jin, R., Breitbart, Y., and Muoh, C. (2009). Data discretization unification. In *Knowledge and Information Systems*, volume 19(1), pages 1–29.
- [48] Kaytoue-Uberall, M., Duplessis, S., and Napoli, A. (2008). Using Formal Concept Analysis for the extraction of groups of co-expressed genes. volume 14 of *CCIS*, pages 439–449, Berlin/Heidelberg. Springer-Verlag.
- [49] Kelliher, F. (2011). Interpretivism and the pursuit of research legitimisation: an integrated approach to single case design. *Leading Issues in Business Research Methods*, 1.
- [50] Kitchenham, B., Linkman, S., and Law, D. (1997). DESMET: a methodology forevaluating software engineering methods and tools. In *Computing & Control Engineering Journal*, 8(3), pages 120–126.
- [51] Kitchenham, B. A. (1996). Evaluating software engineering methods and tools part 1: The evaluation context and evaluation methods. In *ACM SIGSOFT Software Engineering Notes*, 21(1), pages 11–14.

-
- [52] Knuth, D. E. (1964). Backus Normal Form vs. Backus Naur Form. In *Communications of the ACM*, volume 7(12), pages 735–736.
 - [53] Krajca, P., Outrata, J., and Vychodil, V. (2008). Parallel recursive algorithm for FCA. In Belohavlek, R. and Kuznetsov, S., editors, *Proceedings of Concept Lattices and their Applications*.
 - [54] Kreutz, D., Yu, J., Verissimo, P., Magalhaes, C., and Ramos, F. (2017). The KISS principle in software-defined networking: An architecture for Keeping It Simple and Secure.
 - [55] Kuznetsov, S. O. (1999). Learning of simple Conceptual Graphs from positive and negative examples. In Zytkow, J. M. and Rauch, J., editors, *PKDD'99, Lecture Notes in Computer Science*, volume 1704, pages 384–391. Springer.
 - [56] Kuznetsov, S. O. and Obiedkov, S. A. (2002). Comparing performance of algorithms for generating concept lattices. In *Journal of Experimental and Theoretical Artificial Intelligence*, volume 14, pages 189–216.
 - [57] Lehmann, F. and Wille, R. (1995). A triadic approach to Formal Concept Analysis. In *International Conference on Conceptual Structures (ICCS) 1995. Conceptual Structures: Applications, Implementation and Theory*, volume 954 of *Lecture Notes in Computer Science (LNCS)*, pages 32–43.
 - [58] Luo, Y., Liang, P., Wang, C., Shahin, M., and Zhan, J. (2021). Characteristics and challenges of low-code development: The practitioners' perspective. In *CM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM) 2021*. ACM, New York, NY, USA.
 - [59] MacHale, P. D. (1985). *George Boole – His Life and Work*. Boole Press.
 - [60] Melo, C., Orphanides, C., McLeod, K., Aufaure, M.-A., Andrews, S., and Burger, A. (2013). A conceptual approach to gene expression analysis enhanced by visual analytics. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 1314–1319.
 - [61] Mohamed, A., Wanyama, T., Ruhe, G., Eberlein, A., and Far, B. (2004). COTS evaluation supported by knowledge bases. In *Advances in Learning Software Organizations*, pages 43–54. Springer Berlin Heidelberg.
 - [62] Nisbet, R., Elder, J., and Miner, G. (2009). Statistical analysis and data mining applications 2009. pages 51–73. Elsevier.
 - [63] Onwuegbuzie, A. J. and Leech, N. L. (2007). Validity and qualitative research: An oxymoron? In *Quality and Quantity*, volume 41(2), pages 233–249.
 - [64] Orphanides, C. (2010). FcaBedrock, a formal context creator. In *Final Year Project, BSc (Hons) Computing, Department of Computing, Sheffield Hallam University*.
 - [65] Orphanides, C. (2011). Exploring the applicability of Formal Concept Analysis on market intelligence data. In Dau, F., editor, *CEUR Workshop Proceedings 2011*, volume 753. CEUR.

-
- [66] Orphanides, C., Akhgar, B., and Bayerl, P. S. (2016). Discovering knowledge in online drug transactions using Conceptual Graphs and Formal Concept Analysis. In *2016 European Intelligence and Security Informatics Conference (EISIC)*, pages 100–103. Institute of Electrical and Electronics Engineers (IEEE).
- [67] Passin, T. B. (2004). *Explorer's Guide to the Semantic Web*. Manning Publications Co.
- [68] Prediger, S. (1997). Logical Scaling in Formal Concept Analysis. In Lukose, D., Delugach, H., Keeler, M., Searle, L., and Sowa, J., editors, *Conceptual Structures: Fulfilling Peirce's Dream, Proceedings of the Fifth International Conference on Conceptual Structures (ICCS) 1997*, number 1257 in Lecture Notes in Computer Science (LNCS), pages 332–341. Springer.
- [69] Priss, U. (2006). Formal Concept Analysis in Information Science. In Cronin, B., editor, *Annual Review of Information Science and Technology (ASIST)*, volume 40, pages 521–543.
- [70] Raymond, E. (2004). *The Art of Unix Programming*. Addison-Wesley.
- [71] Richardson, L., Venkataraman, S., Stevenson, P., Yang, Y., Burton, N., Rao, J., Fisher, M., Baldock, R. A., Davidson, D. R., and Christiansen, J. H. (2010). EMAGE mouse embryo spatial gene expression database: 2010 update. *Nucleic Acids Research*, 38.
- [72] Richardson, L., Venkataraman, S., Stevenson, P., Yang, Y., Moss, J., Graham, L., Burton, N., Hill, B., Rao, J., Baldock, R. A., and Armit, C. (2014). EMAGE mouse embryo spatial gene expression database: (2014 update). *Nucleic acids research*, 42.
- [73] Rioult, F., Robardet, C., Blachon, S., Crémilleux, B., Gandrillon, O., and Boulicant, J.-F. (2003). Mining concepts from large SAGE gene expression matrices. In Boulicant, J.-F. and Dzeroski, S., editors, *Workshop on Knowledge Discovery in Inductive Databases 2003*, pages 107–118.
- [74] Rudin, W. (1976). *Principles of Mathematical Analysis*. International Series In Pure And Applied Mathematics. McGraw-Hill, New York, 3rd edition.
- [75] Runeson, P. and Host, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering*, 2:131–164.
- [76] Sahay, A., Indamutsa, A., Di Ruscio, D., and Pierantonio, A. (2020). Supporting the understanding and comparison of low-code development platforms. In *Proceedings of the 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA) 2020*, pages 171–178.
- [77] Sandelowski, M. (2000). Focus on research methods-whatever happened to qualitative description? In *Research in nursing and health*, volume 4 of 23, pages 334–340.

-
- [78] Sawase, K., Nobuhara, H., and Bede, B. (2009). Visualizing huge image databases by Formal Concept Analysis. In Barjiela, A. and Pedrycz, W., editors, *Human-Centric Information Processing Through Granular Modelling*, volume 182 of *Studies in Computational Intelligence*. Springer, Berlin/Heidelberg.
 - [79] Schell, C. (1992). The value of the case study as a research strategy. In *Manchester, UK: University of Manchester, Manchester Business School*, pages 1–15.
 - [80] Simon, S. (2008). What is categorical data? cmh.edu/stats/definitions/categorical.htm.
 - [81] Stumme, G. (1999). Hierarchies of Conceptual Scales. In *Proceedings of the Workshop on Knowledge Acquisition, Modeling and Management (KAW'99)*, pages 78–95.
 - [82] Tan, P.-N., Steinbach, M., and Kumar, V. (2013). *Introduction to Data Mining, 2013*. Pearson New International Edition.
 - [83] Theiler, K. (2013). *The house mouse: atlas of embryonic development*. Springer Science & Business Media.
 - [84] UK Standing Committee for Quality Assessment (UKSCQA) (2018). Degree classification: Transparent, consistent and fair academic standards. page 10. Universities UK.
 - [85] Venkataraman, S., Stevenson, P., Yang, Y., Richardson, L., Burton, N., Perry T., P., Smith, P., Baldock R., A., Davidson D., R., and Christiansen, J. H. (2008). EMAGE - Edinburgh Mouse Atlas of Gene Expression: 2008 update. In *Nucleic Acids Research*, volume 36(1).
 - [86] Wagner, H. (1973). Begriff. In Krings, H., Baumgartner, H. M., and Wild, C., editors, *Handbuch philosophischer Grundbegriffe*, pages 191–209. Munich.
 - [87] Waszkowski, R. (2019). Low-code platform for automating business processes in manufacturing. In *IFAC-PapersOnLine 52*, volume 10, pages 376–381.
 - [88] Wille, R. (1995a). The basic theorem of Triadic Concept Analysis. In *Order*, volume 12, pages 149–158.
 - [89] Wille, R. (1995b). Begriffsdanken: Von der griechischen philosophie bis zur kunstlichen intelligenz heute. pages 77–109. Diltthey-Kastanie, Ludwig-Georgs-Gymnasium, Darmstadt.
 - [90] Wille, R. (2005). Formal Concept Analysis as Mathematical Theory of Concepts and Concept Hierarchies. In Ganter, B., Stumme, G., and Wille, R., editors, *Formal Concept Analysis: Foundations and Applications, LNCS*, volume 3626, pages 1–33. Springer.
 - [91] Wolff, K. E. (1993). A first course in Formal Concept Analysis: How to understand line diagrams. In *Advances in Statistical Software*, volume 4, pages 429–438.

- [92] Yang, Y. and Webb, G. (2002). A comparative study of discretization methods for Naive-Bayes classifiers. In Yamaguchi, T., Hoffman, A., Motoda, H., and Compton, P., editors, *Proceedings of the Pacific Rim Knowledge Acquisition Workshop (PKAW) 2002*, pages 159–173.
- [93] Yevtushenko, S. (2000a). *Concept Explorer. The user guide*. Available: <https://conexp.sourceforge.net/users/documentation/>.
- [94] Yevtushenko, S. (2000b). System of data analysis "Concept Explorer". (in Russian). In *Proceedings of the 7th national conference on Artificial Intelligence KII-2000*, pages 127–134.
- [95] Yin, R. K. (2014). *Case Study Research Design and Methods*. Sage, 5th edition.
- [96] Zainal, Z. (2007). Case study as a research method. In *Jurnal Kemanusiaan bil.*
- [97] Zaki, M. J. and Hsiao, C.-J. (2005). Efficient algorithms for mining closed itemsets and their lattice structure. In *IEEE Transactions on Knowledge and Data Mining*, volume 17 of 4, pages 462–478. IEEE Computer Society.

Appendix A

Relevant Publications

A list of publications derived from the work carried out as part of this thesis, can be seen below. As at the time of writing, the list is up-to-date. The publications are sorted in ascending chronological order.

1. Andrews S., and Orphanides, C. (2010a). *FcaBedrock, a Formal Context Creator*. In: Proceedings of the International Conference on Conceptual Structures (ICCS) 2010. pp. 181–184.
2. Andrews S., and Orphanides, C. (2010b). *Analysis of Large Data Sets Using Formal Concept Lattices*. In: Kryszkiewicz, M. and Obiedkov, S. (eds.) Proceedings of the 7th International Conference on Concept Lattices and Their Applications. Seville, University of Seville. pp. 104–115.
3. Andrews, S., and Orphanides, C. (2010c). *Knowledge Discovery Through Creating Formal Contexts*. In: Proceedings of the International Conference on Intelligent Networking and Collaborative Systems 2010. IEEE Computer Society. pp. 455–460.
4. Andrews S., Orphanides, C., and Polovina, S. (2010). *Visualising Computational Intelligence Through Converting Data Into Formal Concepts*. In: Proceedings of the International Conference on P2P, Parallel, Grid, Cloud and Internet Computing 2010. IEEE. pp. 302–307.

-
5. Andrews S., Orphanides, C., and Polovina, S. (2011). *Visualising Computational Intelligence Through Converting Data Into Formal Concepts*. In: Next Generation Data Technologies for Collective Computational Intelligence. Springer, Berlin, Heidelberg. pp. 139–165.
 6. Orphanides, C. (2011). *Exploring the Applicability of Formal Concept Analysis on Market Intelligence Data*. In: Proceedings of the 1st CUBIST Workshop. CEUR-WS. pp. 43–50.
 7. Andrews, S., and Orphanides, C. (2012). *Knowledge Discovery Through Creating Formal Contexts*. In: International Journal of Space-Based and Situated Computing, Vol. 2(2). pp. 123–138.
 8. Melo, C., Orphanides, C. McLeod, K., Aufaure, M.A., Andrews, S., and Burger, A. (2013). *A Conceptual Approach to Gene Expression Analysis Enhanced by Visual Analytics*. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing 2013. Association of Computing Machinery. pp. 1314–1319.
 9. Andrews, S., and Orphanides, C. (2013). *Discovering Knowledge in Data Using Formal Concept Analysis*. In: International Journal of Distributed Systems and Technologies (IJDST), Vol. 4(2). pp. 31–50.
 10. Nwagwu, H., and Orphanides, C. (2015). *Visual Analysis of a Large and Noisy Dataset*. In: International Journal of Conceptual Structures and Smart Applications (IJCSSA). Vol. 3(2). IGI Global. pp. 12–24.
 11. Orphanides, C., and Akhgar, B. and Bayerl, P. S. (2016). *Discovering Knowledge in Online Drug Transactions Using Conceptual Graphs and Formal Concept Analysis*. In: Proceedings of the European Intelligence and Security Informatics Conference (EISIC) 2016. IEEE Computer Society. pp. 100–103.

FcaBedrock, a formal context creator

ANDREWS, S. <<http://orcid.org/0000-0003-2094-7456>> and ORPHANIDES, C.

Available from Sheffield Hallam University Research Archive (SHURA) at:

<http://shura.shu.ac.uk/2104/>

This document is the author deposited version. You are advised to consult the publisher's version if you wish to cite from it.

Published version

ANDREWS, S. and ORPHANIDES, C. (2010). FcaBedrock, a formal context creator. In: 18th International Conference on Conceptual Structures, Kuching, Malaysia, 26-31st July, 2010. (Submitted)

Copyright and re-use policy

See <http://shura.shu.ac.uk/information.html>

FcaBedrock, a Formal Context Creator

Simon Andrews and Constantinos Orphanides

Conceptual Structures Research Group, Communication and Computing Research
Centre, Sheffield Hallam University, Sheffield, UK
`s.andrews@shu.ac.uk`, `corphani@my.shu.ac.uk`

Abstract. FcaBedrock employs user-guided automation to convert c.s.v. data sets into Burmeister .cxt and FIMI .dat context files for FCA.

1 Introduction

Data often exists in the form of flat-files of comma separated values. For FCA to be carried out, these data must be converted into Formal Contexts. Many tools exist to carry out analysis of Formal Contexts but few exist that carry out this preparatory task. Elba performs this task for data-base tables to supply ToscanaJ with Formal Contexts [3], but FcaBedrock deals with flat-file data, producing Formal Context files that can be used by a number of tools and programs. Moreover, FcaBedrock has been developed to convert large data sets into Formal Contexts. It has now been made available at *Sourceforge*¹. FcaBedrock discretizes and Booleanizes data; taking each many-valued attribute and converting it into as many Boolean attributes as it has values and converting continuous values using ranges [4]. Data can be interpreted in many ways leading to inconsistent analysis and problems in measuring the performance of FCA algorithms [1,5]. FcaBedrock solves these problems by documenting data conversions in re-usable, editable, meta-data files called Bedrock files (Figure 1).

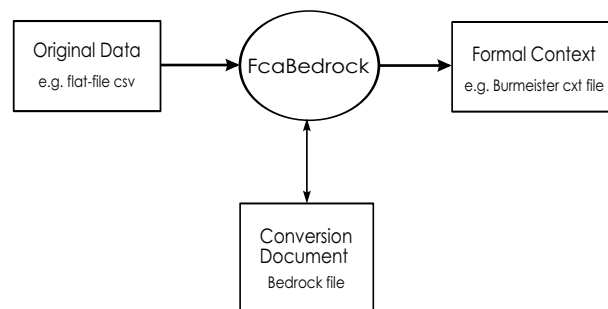


Fig. 1. FcaBedrock Process

¹ <http://sourceforge.net/projects/fcabedrock/>

2 Operation

Figure 2 shows FcaBedrock. There are fields for the names and types of the original data attributes, whether they are to be converted, their categories and the corresponding category values found in the data file. These can be entered by the user, input via a Bedrock file or auto-detected from a data-file. The names of the categories and the category file values are not always the same, so FcaBedrock uses both; the category values are required for converting data and the category names appear in the Context file. The example shown is the *Mushroom* data set from the UCI Machine Learning Repository [2].

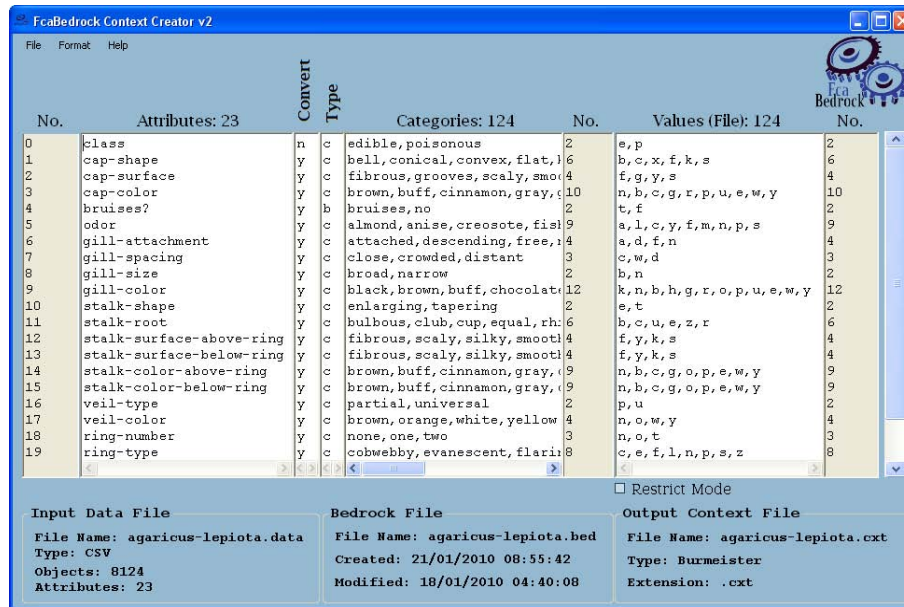


Fig. 2. FcaBedrock

3 Capabilities

The following is a list of some of the capabilities of FcaBedrock (for a more detailed description, see the software documentation at *Sourceforge*). Figure 3 illustrates some of features using the *Adult* data set from UCI [2].

- Data types converted: categorical, continuous and Boolean.
- Input file formats: Many column csv, three-column csv (triples).
- Output file formats: Burmeister (.cxt), FIMI format (.dat)².
- Auto-detection of meta-data from data-file.

² <http://fimi.cs.helsinki.fi/>

- Interpretation of data:
 - Restrict conversion to user-defined values.
 - Exclude from conversion user-defined values.
 - Freedom over treatment of missing values.

No.	Attributes: 15	Convert Type	Categories: 103	No.	Restrict Values (File): 5	No.
0	age	y o	<,20,40,60,>	4		0
1	workclass	y c	Private,Self-emp-not-inc,	8		0
2	fnlwgt	n c		0		0
3	education	y c	Bachelors,Some-college,11	16	Bachelors, Masters, Doc	3
4	education-num	n c		0		0
5	marital-status	y c	Married-civ-spouse,Divorc	7		0
6	occupation	y c	Tech-support,Craft-repair,	14		0
7	relationship	y c	Wife,Own-child,Husband,No	6		0
8	race	y c	White,Asian-Pac-Islander,	5		0
9	sex	y c	Female,Male	2	Female	1
10	capital-gain	n c		0		0
11	capital-loss	n c		0		0
12	hours-per-week	n c		0		0
13	native-country	y c	United-States,Cambodia,Eng	41		0
14	class	n c	>50K,<=50K	2	>50K	1

Fig. 3. Creating an *Adult* sub-context using restrict-to values

The following is an example adapted from the UCI *Adult* data set, using a data file called *mini-adult.data* with eight instances and five attributes (*age*, *education*, *employment*, *sex* and *US-citizen*) plus a *salary* class. File 2 shows a corresponding output from FcaBedrock in the ext format.

```

39, Bachelors, Clerical, Male, Yes, <=50K
50, Bachelors, Managerial, Female, Yes, <=50K
38, HS-grad, Unskilled, Male, Yes, <=50K
53, 11th, Unskilled, Male, Yes, <=50K
28, Bachelors, Professional, Female, Yes, >50K
37, Masters, Managerial, Female, No, <=50K
49, ?, Clerical, Female, No, <=50K
52, HS-grad, Managerial, Male, Yes, >50K

```

File 1. *mini-adult.data*

4 Evaluation

An initial version FcaBedrock was evaluated by a class of final-year Computing undergraduates at Sheffield Hallam University (SHU). Results of this evaluation fed into the development of the version now at *Sourceforge*. This version was then evaluated by successfully converting a number of data sets into Formal Contexts, including the *Mushroom*, *Adult*, *Internet Advertisements*, *Flags* and *Tic-tac-toe* data sets from UCI and several internal student information and supermarket data sets at SHU. The Context files produced were successfully and consistently processed by two Formal Concept generators.

B	7	employment-Unskilled
	age-<30	sex-Male
8	age-30to<40	sex-Female
15	age-40to<50	US-citizen
	age->=50	.X..X...X...X.X
0	education-Bachelors	...XX...X...XX
1	education-Masters	.X.....X...XX.X
2	education-11th	...X..X...XX.X
3	education-HS-grad	X...X.....X..XX
4	employment-Clerical	.X...X...X...X.
5	employment-Managerial	..X.....X...X.
6	employment-Professional	...X...X.X...X.X

File 2. mini-adult.cxt, *Burmeister* context file.

There are several other file formats used in FCA, obtainable from a cxt file produced by FcaBedrock using the conversion tool, *FcaStone* [7]. A version of FcaBedrock is being developed that takes RDF-S and OWL as input [6]. This work will form a core part of CUBIST (“Combining and Uniting Business Intelligence with Semantic Technologies”), awarded under the European Union’s 7th Framework Programme, 5th ICT call, topic 4.3: Intelligent Information Management; STREP Project No.: FP7 257403.

References

1. Andrews, S.: Data Conversion and Interoperability for FCA. In: CS-TIW 2009, pp. 42–49 (2009), http://www.kde.cs.uni-kassel.de/ws/cs-tiw2009/proceedings_final_15July.pdf
2. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA (2007), <http://www.ics.uci.edu/~mllearn/MLRepository.html>
3. Becker, P., Correia, J.H.: The ToscanaJ Suite for Implementing Conceptual Information Systems. In: Ganter, B., Stumme, G., Wille, R. (eds.) Formal Concept Analysis. LNCS (LNAI), vol. 3626, pp. 324–348. Springer, Heidelberg (2005)
4. Ganter, B., Wille, R.: Conceptual Scaling. In: Roberts, F. (ed.) Applications of Combinatorics and Graph Theory to the Biological and Social Sciences. IMA, vol. 17, pp. 139–168. Springer, Heidelberg (1989)
5. Kuznetsov, S.O., Obiedkov, S.A.: Comparing Performance of Algorithms for Generating Concept Lattices. Journal of Experimental and Theoretical Artificial Intelligence 14, 189–216 (2002)
6. Passin, T.B.: Explorer’s Guide to the Semantic Web, Manning, Greenwich, CT (2004)
7. Priss, U.: FcaStone - FCA File Format and Interoperability Software. In: Croitoru, M., Jaschke, R., Rudolph, S. (eds.) CS-TIW 2008, pp. 33–43 (2008)

Analysis of Large Data Sets using Formal Concept Lattices

Simon Andrews and Constantinos Orphanides

Conceptual Structures Research Group
Communication and Computing Research Centre
Faculty of Arts, Computing, Engineering and Sciences
Sheffield Hallam University, Sheffield, UK
`s.andrews@shu.ac.uk` `corphani@my.shu.ac.uk`

Abstract. Formal Concept Analysis (FCA) is an emerging data technology that has applications in the visual analysis of large-scale data. However, data sets are often too large (or contain too many formal concepts) for the resulting concept lattice to be readable. This paper complements existing work in this area by describing two methods by which useful and manageable lattices can be derived from large data sets. This is achieved through the use of a set of freely available FCA tools: the context creator *FcaBedrock* and the concept miner *In-Close*, that were developed by the authors, and the lattice builder *ConExp*. In the first method, a sub-context is produced from a data set, giving rise to a readable lattice that focuses on attributes of interest. In the second method, a context is mined for ‘large’ concepts which are then used to re-write the original context, thus reducing ‘noise’ in the context and giving rise to a readable lattice that lucidly portrays a conceptual overview of the large set of data it is derived from.

1 Introduction

It has been shown that a variety of data sets can be converted into formal contexts [7,2] by a process of discretising and Booleanising the data. However, data sets of only modest size can produce contexts containing hundreds of thousands of formal concepts [9], making the resulting concept lattices unreadable and unmanageable. Perhaps more pertinent than size, however, is the density of and ‘noise’ in a context; factors that increase the number of formal concepts. There is also an issue in computing large numbers of formal concepts; much of the existing software are not capable of carrying out this task on a large scale. Tools such as ToscanaJ [5] and Concept Explorer (ConExp) [14] exist that compute and visualise concept lattices but are not designed to do so for large numbers of concepts.

This paper describes two ways in which concept lattices can be produced from data sets: 1) by creating sub-contexts by restricting the conversion of the data to information of interest, and 2) by removing relatively small concepts from a context to reduce ‘noise’, so that a readable, yet still meaningful, concept lattice can be produced.

2 Analysis of Sub-Contexts from Data Sets

*FcaBedrock*¹ is a freely available tool developed by the authors that converts *csv* format data files into formal context *cxt* files and FIMI data format files [3]. It reads a data file and automatically converts each many-valued data attribute in the file to formal attributes. The process is guided by the user in deciding how the data set should be interpreted. The user can specify, for example, discrete ranges for continuous data attributes and what names should be given in the *cxt* file to the formal attributes. The user can also create sub-contexts by restricting the conversion to only the data attributes of interest for a particular analysis. Such meta-data, used to guide the conversion process, is stored in a separate file called a *Bedrock* file. These files can be loaded into *FcaBedrock* to repeat the conversion, or allow changes in the meta-data to be made to produce different sub-contexts for alternative analyses.

2.1 Attribute Exclusion

To illustrate the production of concept lattices from data sets using sub-contexts, the well-known *Mushroom* and *Adult* data sets from the *UCI Machine Learning Repository* [4] will be used. The Mushroom data set contains data of 8124 edible and poisonous mushrooms of the families *agaricus* and *lepiota*. It is many-valued categorical data with attributes describing properties such as stalk shape, cap colour and habitat. As an example of the *agaricus* family of mushrooms, Figure 1 is of a meadow *agaricus*.

Fig. 1. A meadow *agaricus* mushroom (source www.50birds.com/gmushrooms1.htm)



The data set was converted by *FcaBedrock* using all of the attributes and their categories. The resulting *cxt* file was processed by a formal concept miner developed by one of the authors, called *In-Close*² [1], generating over 220,000 concepts; far too many to visualise.

However, let us say we are interested in the relationship between mushroom habitat and population type. Figure 2 shows the meta-data for the Mushroom data set loaded into *FcaBedrock*. The *Convert* column was used to select only habitat and population to convert. In this way, a Mushroom sub-context was created containing formal attributes only for habitat and population. There were 13 formal attributes in all, corresponding to the 7 categories of habitat: *grasses*,

¹ <https://sourceforge.net/projects/fcabedrock>

² <https://sourceforge.net/projects/inclose>

leaves, meadows, paths, urban, waste and woods, and 6 categories of population type: *abundant, clustered, numerous, scattered, several and solitary*.

No.	Attributes: 23	Convert	Type	Categories: 13	No.	Values (File): 13	No.
7	gill-spacing	n	c	close, crowded, distant	3	c, w, d	3
8	gill-size	n	c	broad, narrow	2	b, n	2
9	gill-color	n	c	black, brown, buff, choc	12	k, n, b, h, g, r, o, p, u, e,	12
10	stalk-shape	n	c	enlarging, tapering	2	e, t	2
11	stalk-root	n	c	bulbous, club, cup, equa	6	b, c, u, e, z, r	6
12	stalk-surface-above-	n	c	fibrous, scaly, silky, s	4	f, y, k, s	4
13	stalk-surface-below-	n	c	fibrous, scaly, silky, s	4	f, y, k, s	4
14	stalk-color-above-ri	n	c	brown, buff, cinnamon, g	9	n, b, c, g, o, p, e, w, y	9
15	stalk-color-below-ri	n	c	brown, buff, cinnamon, g	9	n, b, c, g, o, p, e, w, y	9
16	veil-type	n	c	partial, universal	2	p, u	2
17	veil-color	n	c	brown, orange, white, ye	4	n, o, w, y	4
18	ring-number	n	c	none, one, two	3	n, o, t	3
19	ring-type	n	c	cobwebby, evanescent, f	8	c, e, f, l, n, p, s, z	8
20	spore-print-color	n	c	black, brown, buff, choc	9	k, n, b, h, r, o, u, w, y	9
21	population	y	c	abundant, clustered, nu	6	a, c, n, s, v, y	6
22	habitat	y	c	grasses, leaves, meadow	7	g, l, m, p, u, w, d	7

Input Data File
File Name: agaricus-lepiota.data
Type: CSV
Objects: 8124
Attributes: 23

Bedrock File
File Name: agaricus-lepiota.bed
Created: 25/01/2010 12:19:16
Modified: 14/01/2010 15:00:50

Output Context File
File Name: mushMabiPop.cxt
Type: Burmeister
Extension: .cxt

Fig. 2. Mushroom habitat/population sub-context being created by FcaBedrock

This cxt file was then processed by ConExp to produce the concept lattice in Figure 3. In ConExp, the size of the node in the lattice can be made proportional to the number of own objects (mushrooms), so it can be seen that, for example, clusters of mushrooms are found in similar numbers in woods, leaves and waste ground, but not in other habitats. Solitary mushrooms are most likely to be found in woods, although they can occasionally be found in paths, urban areas and grassland.

A similar approach is applied by TocscanaJ [5], where an individual categorical attribute or a pair of continuous attributes are scaled to produce a lattice. A further attribute can be added as a nested lattice. The nested results are not always easy to interpret, however, sometimes requiring some visual cross-referencing of diagrams, and adding further lattice ‘nests’ does not appear possible or practical.

2.2 Object Exclusion

A second analysis was carried out, this time using the Adult data set [4]. This data set is US Census data of 32,561 adults, with attributes such as age, edu-

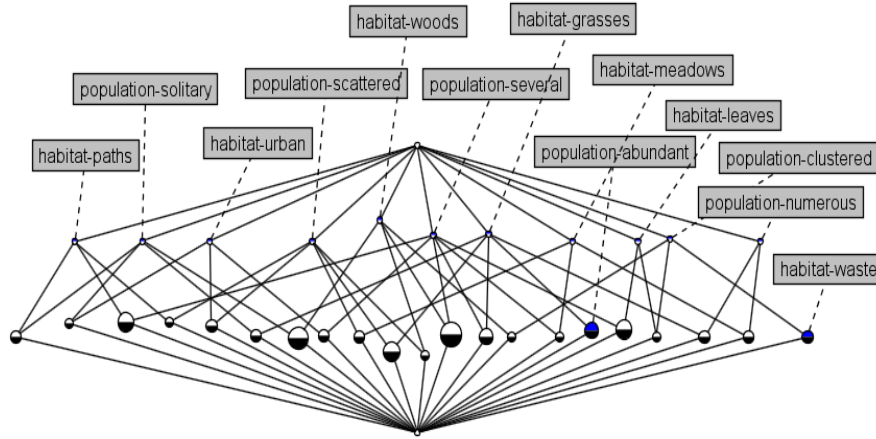


Fig. 3. Mushroom habitat/population lattice in ConExp

cation and employment type. The formal context produced when all the (suitable) attributes were converted contained over 100,000 concepts. However, let us say that in this analysis we are interested in comparing how pay is effected by gender in adults who have had a higher education. To carry out this analysis, FcaBedrock was used (Figure 4) to convert only the sex (*male*, *female*), class (pay $\leq \$50k$, pay $> \$50k$) and education attributes. Furthermore, FcaBedrock's attribute value restriction feature was used to convert only those objects (adults) with the education attribute value *Bachelors*, *Masters* or *Doctorate* (three out of the 16 possible categories of education in the data set). This resulted in a sub-context with 7 formal attributes and 7,491 objects.

The resulting concept lattice in ConExp is shown in Figure 5, containing 37 concepts. The number of objects (and the percentage of the whole) is being displayed for the concepts of interest. Because only objects with Bachelors, Masters or Doctorate education categories were included in the conversion, there were 13 education categories (such as 10th grade and high school graduate) left with zero objects associated with them. Such unsupported formal attributes are normally labeled at the infimum of the concept lattice, but these labels can be hidden in ConExp.

Although a little cluttered with lines, the lattice is still readable, aided by a ConExp feature whereby information regarding a concept is displayed when a node is pointed to with the mouse. An examination of the concepts allows us to compare the percentage of males and females who earn more than \$50k, for each type of higher education. With a Bachelor's degree, 21% of females and 50% of males earned more than \$50k. This 'gender gap' was maintained at Master's level, with 33% of females and 65% of males earning more than \$50k,

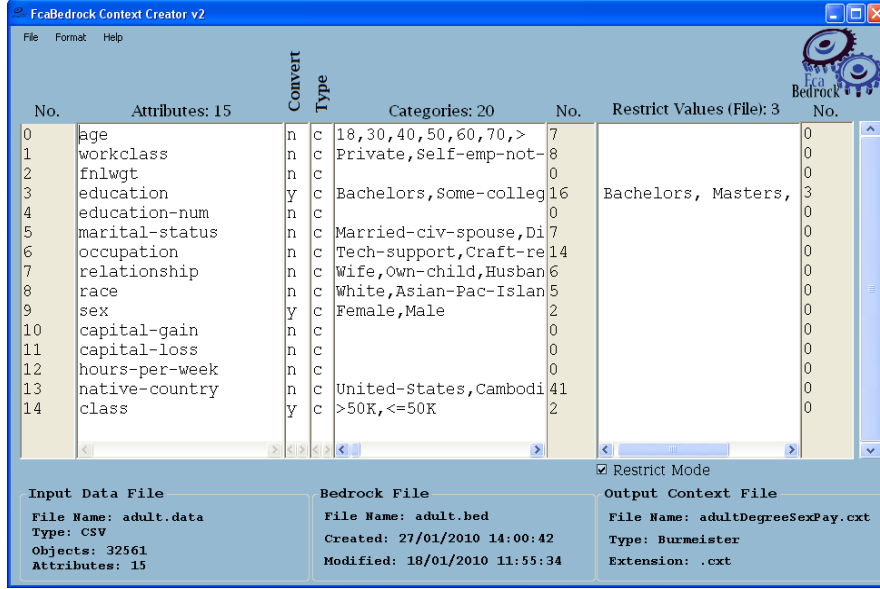


Fig. 4. Adult Degree/sex/pay sub-context being created by FcaBedrock

but narrowed slightly at Doctorate level, with 58% of females and 78% of males earning more than \$50k.

3 Reducing ‘Noise’ in a Context

The approaches described so far rely on reducing the size of the context. The next approach is to focus on the size of the concepts, using the well-known idea of minimum support [15] to filter out relatively small concepts (noise) from the data. This is achieved by specifying a minimum number of objects and/or attributes for a concept. Noise is therefore simply the concepts containing numbers of attributes or objects smaller than the user-defined minimums. This approach has been shown to be useful in gene-expression analysis [8] although the process described appeared to involve a degree of manual manipulation of the data and bespoke programming, and the analysis stopped short of visualising the results in a concept lattice.

In contrast, the approach described here is a semi-automated form of lattice ‘iceberging’ [13] to reduce noise using minimum support to such an extent that a manageable and meaningful concept lattice can be produced from the remaining concepts, thus giving a broad conceptual overview of the data. The reduction of noise is achieved by mining a context for concepts that satisfy a minimum support and then *re-writing* the context using only those concepts.

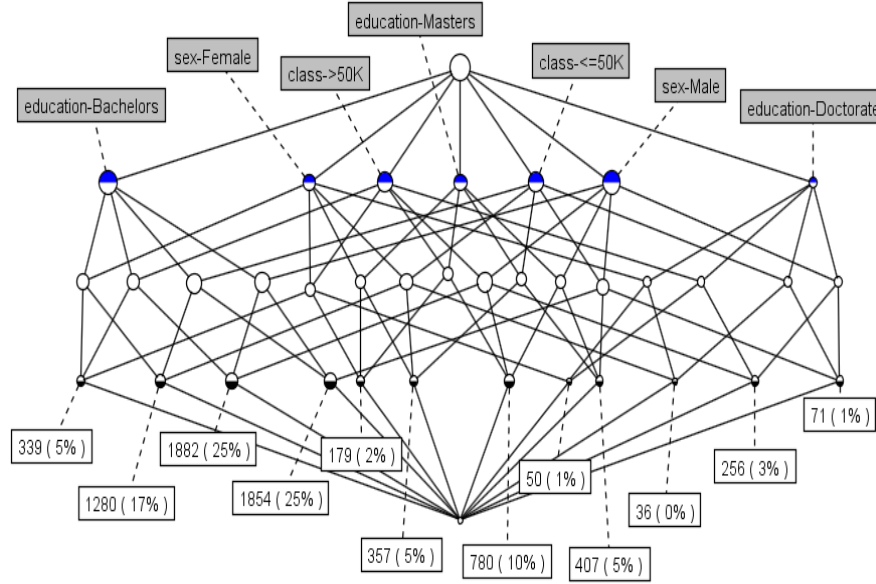


Fig. 5. Adult Degree/sex/pay lattice in ConExp

In-Close does this automatically. After mining concepts that satisfy a minimum support, In-Close uses them to output a ‘quiet’ (or ‘clean’) version of the original cxt file. This can then be used to produce a readable concept lattice. Figures 6 and 7 show a small example of applying a minimum support of two attributes and two objects. The two concepts large enough to satisfy the minimum support are $(\{a0, a1, a2\}, \{o0, o1\})$ and $(\{a2, a3\}, \{o1, o2\})$. These are mined and used to create the ‘quiet’ version. The lattices of the small ‘noisy’ and ‘quiet’ examples are shown in Figures 8 and 9 respectively.

In contrast to traditional iceberging, where a lattice is truncated by removing concepts that do not have a defined minimum number of objects, a complete hierarchy is maintained here in the resulting lattice, because where the large concepts ‘overlap’, other concepts are found during a *second pass* of concept mining (with no minimum support) when producing the concept lattice. In this way, possibly significant concepts that would not have satisfied the original minimum support are retained. In this simple example, the connecting concepts are $(\{a2\}, \{o0, o1, o2\})$ and $(\{a0, a1, a2, a3\}, \{o1\})$. These will be generated, along with the ones that satisfied the original minimum support. It should be noted that this means that it is the second, usually larger number of concepts that has to be noted when deciding on the level at which a manageable lattice is produced.

	a0	a1	a2	a3	a4	a5
o0	×	×	×			
o1	×	×	×	×		×
o2			×	×		
o3	×				×	

Fig. 6. A small ‘noisy’ context

	a0	a1	a2	a3	a4	a5
o0	×	×	×			
o1	×	×	×	×		
o2			×	×		
o3						

Fig. 7. A ‘quiet’ version of the ‘noisy’ context

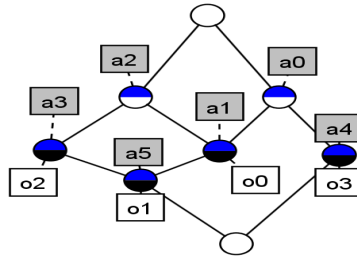


Fig. 8. Lattice in ConExp of the small ‘noisy’ context

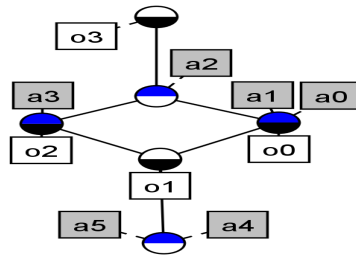


Fig. 9. Lattice in ConExp of the small ‘quiet’ context

3.1 A Student Survey Example

To illustrate this method, an analysis is carried out here on a set of student survey data [6] consisting of demographic and ‘problem’ data from 587 university undergraduates. The ‘problem’ data consists of ‘yes/no’ responses to 36 problems (or reasons for problems) that a student may have experienced during their studies; such as missing too many lectures, performing below their expectations, finding it difficult to settle or having high outside commitments. This is a good example of a noisy data set: there were 145 formal attributes when all the original attributes were converted by FcaBedrock, and the rather subjective ‘yes/no’ data gave rise to a context that was quite dense (20%) and noisy. Using In-Close, there were found to be 22,760,243 concepts.

However, let us say that we were only interested in analysing the ‘problem’ data. By only converting these attributes and excluding the demographic ones, a context containing 339,672 concepts was produced; a significant reduction but still far too many for a readable lattice. To reduce this number of concepts further, In-Close was used with minimum support specified for the intent and extent. The minimum size of intent was set to four. Although this may seem an arbitrary choice, it was decided that this would represent a sensible number if we were interested in combinations of problems that are experienced by students. The minimum support for the extent (number of students) was then varied to obtain large concepts that were small enough in number to obtain a readable lattice. Experimenting with In-Close, it appeared that a minimum support of between 80 and 70 students would produce a manageable lattice with between 32 and 160 nodes.

The lattice shown in Figure 10 was produced using the context generated by In-Close with a minimum support of 80 students (minimum of four attributes). In this case, the nodes in ConExp had a fixed radius to make them more prominent. The attributes related to the following problems asked in the student survey:

- course: Have sometimes found course stressful
- stress: Stress problems
- leave: Considered changing/leaving at some stage
- different: Course different from expectation
- examinations: Examination performance below expectations
- commitments: Outside commitments high
- distractions: Too many distractions that affect ability to study

The lattice appeared to give some useful insight into commonly connected problems experienced by undergraduates. It seemed that a stressful course was a common problem (217 students) in combination with general stress problems, considering leaving a course, finding a course different to expectations and not performing as expected in examinations. Problems that were less common, but still related, seemed to be having high outside commitments and a large number of distractions; all students who reported these problems found their course stressful.

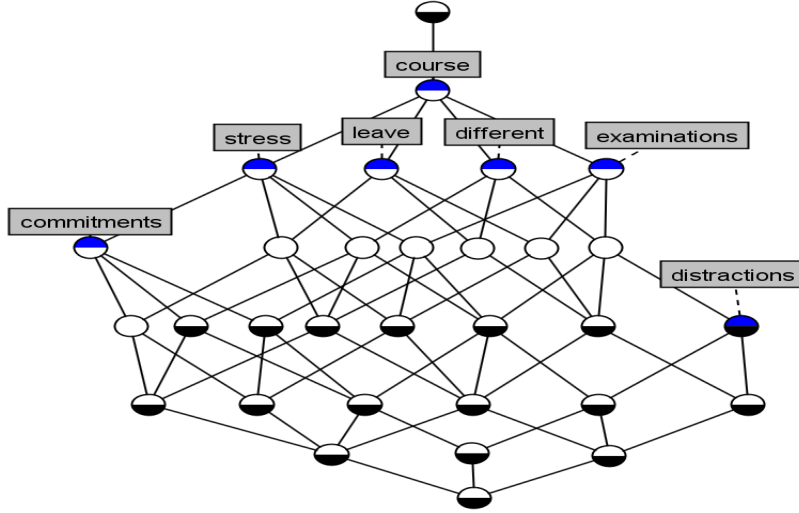


Fig. 10. Student problem lattice in ConExp

3.2 Comparing Quiet Sub-Contexts

To further illustrate the usefulness of this method, a second analysis of the Mushroom data set was carried out, this time to investigate differences between poisonous and edible mushrooms. The attribute value restriction of FcaBedrock was used to divide the mushroom context into an edible mushroom sub-context and a poisonous mushroom sub-context. There were 4,208 objects (mushrooms) and 92,543 concepts in the edible mushroom sub-context and 3,916 objects and 86,198 concepts in the poisonous mushroom sub-context. To provide a significant number of attributes to compare, 10 was chosen as the minimum support for intent. The process of reducing noise using minimum support by In-Close was carried out, resulting in an edible mushroom concept lattice containing 2,848 objects and 17 concepts, and a poisonous mushroom concept lattice containing 3,344 objects and 14 concepts (Figures 11 and 12).

Similarities were identified as attributes expressed in both lattices and were moved to the right of each lattice. Differences were identified as attributes expressed in only one lattice and were moved to the left, thus giving a clear visualisation for comparison. Examining the lattices suggest combinations of features that indicate if a mushroom is safe to eat. For example, it seems that smooth-stalked mushrooms are likely to be safer to eat than silky-stalked ones. The lattices also suggest that mushrooms with pendant rings are more wholesome than those with evanescent rings. The combination of a foul smell, bulbous root and a chocolate coloured spore print would seem to be a strong indication of danger. The fact that bruised mushrooms are likely to be edible could be be-

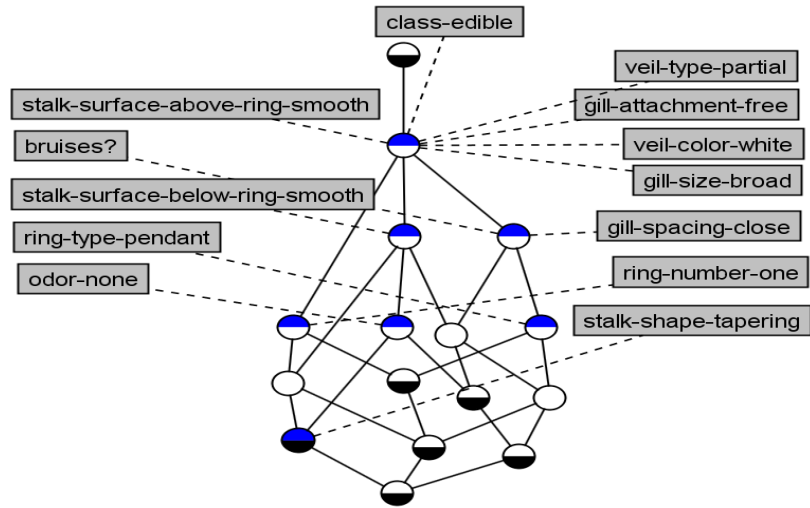


Fig. 11. Edible mushroom lattice in ConExp

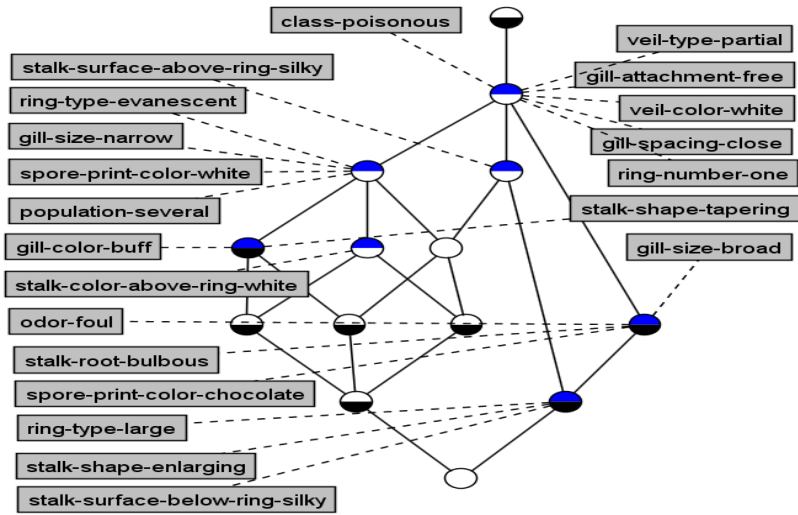


Fig. 12. Poisonous mushroom lattice in ConExp

cause foraging animals may damage mushrooms that are near to ones they are eating.

4 Conclusion

Although large data sets may be difficult to deal with computationally, it is the number of formal concepts derived from a data set that is the key factor in determining if a concept lattice will be useful as a visualisation. Typical data sets contain too many concepts. It has been shown here that readable lattices can be produced from real data sets with a straightforward process of creating sub-contexts and reducing noise, using freely available software. Whilst the analyses presented here have not been rigorously corroborated with domain expertise or statistical analysis of the data, we have nonetheless demonstrated that understandable results are obtainable from existing data sources where this would not normally have been the case. Formal contexts that would normally be intractable for visualisation have been processed in a disciplined manner to provide meaningful results. This has been achieved by 1) focusing on information of interest and by 2) reducing ‘noise’ in the context, thus revealing readable lattices that lucidly portray conceptual meaning in large data sets.

The cxt format, commonly used in FCA, was used successfully here to inter-operate between the tools. Further integration of the tools would make the analysis easier to perform. This would also give scope for improvements such as discarding the attributes that are left with no objects after the second pass of concept mining, rather than manually hiding these labels in ConExp. Integration will also open up the possibility of more dynamic analysis, where a user can quickly see changes in the lattice when altering the parameters of analysis. Some of the operations, such as determining the level of minimum support, would particularly benefit from this greater degree of automation.

There may be limitations in the technique of noise reduction presented here; for example, if two otherwise identical attributes that have high support differ only in one object (possibly even due to error of data entry), concepts for the first and second attribute will be included in the resulting lattice. Such anomalies might be removed by applying the notion of concept stability, where the stability of a concept is defined by the extent to which its attributes are dependent on its objects (a stable concept is not affected by attribute ‘noise’ in object descriptions) [10] or by applying so-called fault-tolerant FCA, in allowing a bound number of exceptions (false values) in a concept [11]. A comparison of techniques of noise reduction and concept clustering would be useful.

Acknowledgement This work is part of the CUBIST project (“Combining and Uniting Business Intelligence with Semantic Technologies”), funded by the European Commission’s 7th Framework Programme of ICT, under topic 4.3: Intelligent Information Management.

References

1. Andrews, S.: In-Close, A Fast Algorithm for Computing Formal Concepts. In: Rudolph, S., Dau, F., Kuznetsov, S.O. (eds.) ICCS'09, <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-483/> (2009)
2. Andrews, S.: Data Conversion and Interoperability for FCA. In: CS-TIW 2009, pp. 42-49, http://www.kde.cs.uni-kassel.de/ws/cs-tiw2009/proceedings_final_15July.pdf (2009)
3. Andrews, S., Orphanides, C.: *FcaBedrock, a Formal Context Creator*. In: Croitoru, M., Ferre, S. and Lukose, D. (eds.) ICCS 2010, LNAI 6208. Springer-Verlag, Berlin/Heidelberg (2010)
4. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository [http://www.ics.uci.edu/\\$sim\\$mlearn/MLRepository.html](http://www.ics.uci.edu/simmlearn/MLRepository.html). Irvine, CA: University of California, School of Information and Computer Science (2007)
5. Becker, P., Correia, J.H.: The ToscanaJ Suite for Implementing Conceptual Information Systems. In: Ganter, B., et al. (eds.) Formal Concept Analysis, LNCS (LNAI), vol. 3626, pp. 324-348. Springer-Verlag, Berlin-Heidelberg (2005)
6. Burley, K.: Data Mining Techniques in Higher Education Research: The Example of Student Retention. A thesis submitted in partial fulfillment of the requirements of Sheffield Hallam University for the degree of Doctor of Education. Sheffield Hallam University (2006)
7. Ganter, B., Wille, R.: Conceptual Scaling. In: Roberts, F. (ed.) Applications of Combinatorics and Graph Theory to the Biological and Social Sciences. IMA, vol. 17, pp. 139-168, Springer, Berlin-Heidelberg-New York (1989)
8. Kaytoue-Uberall, M., Duplessis, S., Napoli, A.: Using Formal Concept Analysis for the Extraction of Groups of Co-expressed Genes. In: Le Thi, H.A., Bouvry, P., Pham Dinh, T. (eds.) MCO 2008. CCIS vol. 14, pp. 439-449. Springer-Verlag, Berlin/Heidelberg (2008)
9. Krajca, P., Outrata, J., Vychodil, V.: Parallel Recursive Algorithm for FCA. In: Belohlavek, R., Kuznetsov, S.O. (eds.) CLA 2008, pp. 71-82. Palacky University, Olomouc (2008)
10. Kuznetsov, S., Obiedkov, S., Roth, C.: Reducing the Representation Complexity of Lattice-Based Taxonomies. In: Priss, U., Polovina, S., Hill, R. (eds.) ICCS 2007, LNAI 4604, pp. 241-254. Springer-Verlag, Berlin/Heidelberg (2007)
11. Pensa, R. G., Boulicaut, J-F.: Towards Fault-Tolerant Formal Concept Analysis. In: Banidini, S., Manzoni, S. (eds.) AI*IA 2005, LNAI 3673, pp. 212-223, Springer-Verlag, Berlin/Heidelberg (2005)
12. Priss, U.: FcaStone - FCA File Format and Interoperability Software. In: Croitoru, M., Jaschké, R., Rudolph, S. (eds.) CS-TIW 2008, pp. 33-43 (2008)
13. Stumme, G., Taouil, R., Bastide, Y., Lakhal, L., Conceptual clustering with iceberg concept lattices. In: Proceedings of GIFachgruppentreffen Maschinelles Lernen01, Universitat Dortmund, vol. 763. (2001)
14. Yevtushenko, S.A.: System of data analysis "Concept Explorer". (In Russian). Proceedings of the 7th national conference on Artificial Intelligence KII-2000, p. 127-134, Russia, 2000.
15. Zaki, M. J., Hsiao, C-J.: Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure. In: IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 4, pp. 462-478. IEE Computer Society (2005)

Knowledge discovery through creating formal contexts

ANDREWS, Simon <<http://orcid.org/0000-0003-2094-7456>> and
ORPHANIDES, Constantinos

Available from Sheffield Hallam University Research Archive (SHURA) at:

<http://shura.shu.ac.uk/3710/>

This document is the author deposited version. You are advised to consult the publisher's version if you wish to cite from it.

Published version

ANDREWS, Simon and ORPHANIDES, Constantinos (2010). Knowledge discovery through creating formal contexts. In: XHAFA, F, ABRAHAM, A, DEMETRIADIS, S and CABALLE, S, (eds.) First international workshop on computational intelligence in networks and systems (CINS 2010). IEEE Computer Society, 455-460.

Copyright and re-use policy

See <http://shura.shu.ac.uk/information.html>

Knowledge Discovery through Creating Formal Contexts

Simon Andrews* and Constantinos Orphanides†

Conceptual Structures Research Group, Communication and Computing Research Centre
Faculty of Arts, Computing, Engineering and Sciences, Sheffield Hallam University, Sheffield, UK

*s.andrews@shu.ac.uk †corphani@my.shu.ac.uk

Abstract—Knowledge discovery is important for systems that have computational intelligence in helping them learn and adapt to changing environments. By representing, in a formal way, the context in which an intelligent system operates, it is possible to discover knowledge through an emerging data technology called Formal Concept Analysis (FCA). This paper describes a tool called FcaBedrock that converts data into Formal Contexts for FCA. The paper describes how, through a process of guided automation, data preparation techniques such as attribute exclusion and value restriction allow data to be interpreted to meet the requirements of the analysis. Creating Formal Contexts using FcaBedrock is shown to be straightforward and versatile. Large data sets are easily converted into a standard FCA format.

Keywords—Formal Context; Formal Concept Analysis; guided automation; visualisation; Concept Lattice

I. INTRODUCTION

Formal Concept Analysis (FCA) was introduced in the 1990s by Rudolf Wille and Bernhard Ganter [17] building on applied lattice and order theory developed by Birkhoff and others in the 1930s and was initially developed as a subsection of Applied Mathematics, based on the mathematisation of concepts and concepts hierarchy. The following description of FCA is that used in [4].

A *Formal Concept* is constituted by its *extension*, comprising of all objects which belong to the Concept, and its *intension*, comprising of all attributes (properties, meanings) which apply to all objects in the extension. The set of all objects and attributes together with their relation to each other form a *Formal Context*, which can be represented by a cross-table.

Airlines	Latin America	Europe	Canada	Asia Pacific	Middle East	Africa	Mexico	Caribbean	USA
Air Canada	×	×	×	×	×		×	×	×
Air New Zealand		×		×					×
Nippon Airways		×		×					×
Ansett Australia				×					
Austrian Airlines		×	×	×	×	×			×

The cross-table above is a Formal Context representing the destinations of five airlines, where the elements on the left are formal objects (airlines) and the elements

at the top are formal attributes (destinations). If an object has a specific attribute, it is indicated by placing a cross in the corresponding cell of the table. An empty cell indicates that the corresponding object does not have the corresponding attribute. In the Airlines Context, Air Canada flies to Latin America but does not fly to Africa.

A central notion of FCA is a duality called a ‘*Galois connection*’. This connection is often observed between two types of items that relate to each other. A Galois connection implies that “if one makes the set of one type larger, they will be related to a smaller set of the other type, and vice versa” [14]. In the airlines example, the combination of destinations, Asia Pacific, Europe and USA, are flown to by four airlines. If Middle East is added to the list of destinations, the number of airlines reduces to two.

The definition of a Formal Concept is extended by the idea of closure: the extension contains all objects that have the attributes in the intension, and the intension contains all attributes shared by the objects in the extension. In the example of the two airlines that fly to Asia Pacific, Europe, USA and the Middle East, it can be seen from the table that Canada is also flown to by the same two airlines. Adding Canada to the list of destinations completes (closes) that particular Formal Concept.

A strength of FCA is that the Galois connections between the Formal Concepts can be visualised in a *Concept Lattice* (Figure 1), which is an intuitive way of discovering hitherto undiscovered information in data and portraying the natural hierarchy of Concepts that exist in a Formal Context.

Each node in the lattice is a Formal Concept. The objects (airlines) are labeled below the nodes, while the attributes (destinations) are labeled above the nodes. Extracting information from a lattice is straightforward. In order to see which attributes are featured by an object, one begins from the node where the object is located and moves upwards. Any attributes one meets along the way are the attributes featured by that object. For example, if one heads upwards in the lattice from Air New Zealand (object), one will collect the attributes USA, Europe and Asia Pacific. This can be interpreted

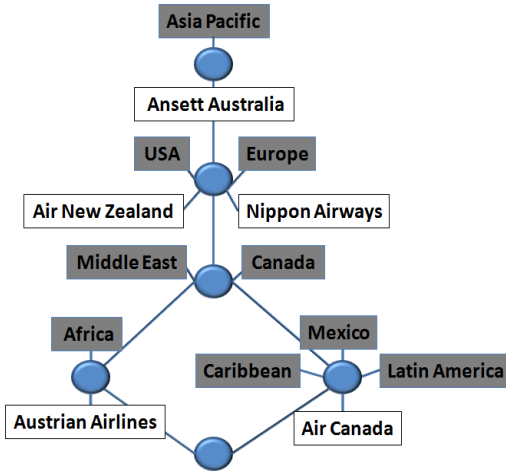


Figure 1. A Lattice corresponding to the Airlines Context

as ‘Air New Zealand flies to USA, Europe and Asia Pacific’. Similarly, in order to see which objects have a specific attribute, one heads downwards in the lattice. Any objects one meets along the way are objects which feature that particular attribute. For example, heading downwards from Canada (attribute), one will collect the objects Austrian Airlines and Air Canada. This can be interpreted as ‘Canada is flown to by Austrian Airlines and Air Canada’. Although the Airline Context is only a small example of FCA, it is evident that the Concept Lattice provides information that is not evident from looking at the table alone. It has been shown that Formal Concept Analysis (FCA) can be usefully applied to large sets of data [9], [10], [16] and that FCA has applications in data mining [6], [15]. Algorithms exist that are capable of processing large Formal Contexts in reasonable time [2], [11]. However, data is rarely in a form immediately suitable for FCA tools and applications. Data often exists in the form of flat-file tables of comma separated values (csv). For FCA to be carried out, these data must be converted into Formal Contexts. The existing, typically many-valued, attributes must be converted into Formal Attributes. This can be an involved and time consuming task, usually requiring a programming element. The task is, essentially, a process of discretizing and Booleanizing data; taking each many-valued attribute and converting it into as many Boolean attributes as it has values. The incorporation of many-valued attributes into FCA is well known [7], [18] and continuous data can be used for FCA by being hierarchically scaled [14] or discretized as disjoint ranges of values [10]. However, there is less work describing how these techniques can be applied in an automated, reproducible way.

Dealing with existing data sets raises a number

issues: Free-text data (such as peoples’ names and addresses) is not usually suitable for Booleanizing; missing values are common and must be dealt with in some way; data classes are sometimes included in the original data, but it may be more appropriate to think of these as sub-contexts rather than attribute values; continuous data can be discretized but the method to use and the boundary values need to be decided; attributes with two values may sometimes be interpreted as being Boolean and thus converted using a single Formal Attribute; attribute values that are unused may be interpreted as not being required in the Formal Context.

The possibility of such diverse interpretation of data can, without careful documentation, lead to inconsistent and incomparable analysis. Several analyses may each use a different interpretation of the same original data set. This can also lead to problems in measuring the performance of FCA algorithms [12].

A tool to address these issues was proposed at CSTIW’09 [3] where an early prototype of what follows was demonstrated.

II. FCABEDROCK OVERVIEW

FcaBedrock is an open-source tool for automating the conversion of existing, flat-file csv data sets into Formal Contexts¹ (Figure 2)

FcaBedrock uses a process of guided automation. The user supplies the tool with appropriate meta-data for conversion, such as the names of the attributes and their values and with decisions as to what to convert and how to convert it. After reading in the original data file, this meta-data is used by FcaBedrock to create a Formal Context file in a standard format for FCA. The meta-data is stored in a separate text document called a Bedrock file. This can be used for subsequent conversions and acts as a record of the interpretation made of the data set. Bedrock files are loaded into FcaBedrock allowing the reproduction of Context files and allowing changes in the interpretation to be made. Different interpretations can be made and constraints applied to a data set allowing different analyses. Each Formal Context produced can be documented with its corresponding Bedrock file. Multiple data files with the same attributes can be converted using the same Bedrock file.

FcaBedrock currently supports two input data file formats: many column csv and three column csv. FcaBedrock outputs the Formal Context in the widely *Burmeister* FCA format.

Many column csv is a traditional flat-file format for many-valued data. Each row represents an instance (object) and each column a many-valued attribute.

¹<https://sourceforge.net/projects/fcabedrock/>

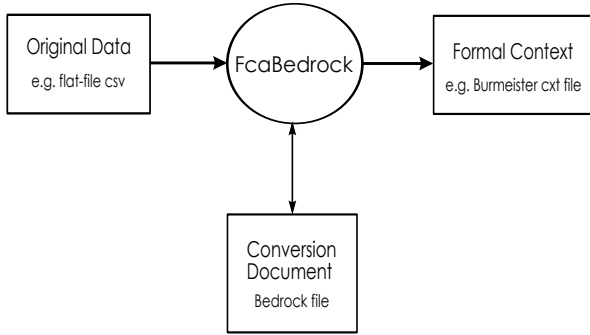


Figure 2. FcaBedrock Process

The three column format is becoming popular as a standardized format and is the approach used, for example, in the Resource Description Format (RDF) [1], [13].

FcaBedrock assumes that, when reading a three column data file, the first value is a Formal Object, the second is a many-valued attribute and the third is an attribute value. The first values are used by FcaBedrock as object names when outputting a Formal Context in the *Burmeister* format (see below). If auto-detection is used, the second values are detected as attribute names. The order in which triples appear in a data file is not significant.

The *Burmeister* Formal Context format (cxt) is a popular format in FCA. A cxt file begins with the number of objects and number of attributes and then lists the objects followed by the attributes. It then stores the body of the Formal Context as a grid using crosses for *True* values and dots for *False* values.

The sections that follow describe how FcaBedrock operates. Examples are given to illustrate how existing data can be interpreted for FCA using public data sets from the UCI Machine Learning Repository [5].

III. ATTRIBUTE VALUES

Figure 3 shows FcaBedrock. There are windows for the names and types of the original data attributes, whether they are to be converted or not, their categories and the corresponding category values found in the data file. All these must initially be entered by the user (unless FcaBedrock’s auto-detection feature is being used - see Section VIII), although copy and paste is available and useful if the names of the categories and the category values are the same. This is often but not always the case, so FcaBedrock uses both; the category values are required for the computation but the category names appear in the Context file. The example shown is of the well-known *Mushroom* data set from UCI [5], where arbitrary letters are used in

the data file to represent attribute categories. The *cap-surface* attribute, for example, has four possible categories, *fibrous*, *grooves*, *scaly* and *smooth*, represented by the values *f*, *g*, *y* and *s*, respectively, in the data file. In another UCI example, the *Adult* data set [5] has category values in the data file that each contain a leading space character. The spaces are better omitted from the category names but are required for converting the data.

IV. ATTRIBUTE TYPES

FcaBedrock uses three types of attribute: categorical, continuous and Boolean. The type is set by the user entering *c* for categorical, *o* for continuous or *b* for Boolean in the Type window.

A. Categorical Attributes

The predominant attribute type is categorical. This is the typical many-valued attribute and is converted by creating one formal attribute for each of the attribute categories. In the conversion process, each row of comma separated values read in from the data file is split into a list of individual values. Each value is compared with the category values listed for the corresponding attribute in the Values (File) window. Attribute zero corresponds to the first column in the data file and so on. A match is recorded as a *True* value in the Formal Context.

In the *Mushroom* example shown in Figure 3, all but one of the attributes are categorical. The *gill-spacing* attribute, for example has three categories: *close*, *crowded*, and *distant*. FcaBedrock will create three corresponding Formal Attributes and name them by concatenating the attribute and category names, thus *gill-spacing-close*, *gill-spacing-crowded*, and *gill-spacing-distant*.

B. Boolean Attributes

A Boolean attribute can be interpreted as a single Formal Attribute. Typically, a Boolean attribute in a data set is one that has two categories that represent *True* and *False*. In the *Mushroom* example, the Boolean type is being used for the *bruises?* attribute. The attribute has two categories, *bruises* and *no*, represented in the data file by the values *t* and *f*. For a Boolean attribute, FcaBedrock uses the first category value as the *True* value, *t* in this example. During the conversion process, it compares the corresponding value read in from the data file with the *True* value. If they match, this is recorded as a *True* value in the Formal Context. FcaBedrock names the Formal Attribute using the original attribute name, without concatenating the name of the *True* category. So, in this example, the original *Bruises?* attribute becomes a single Formal Attribute also called *Bruises?*.

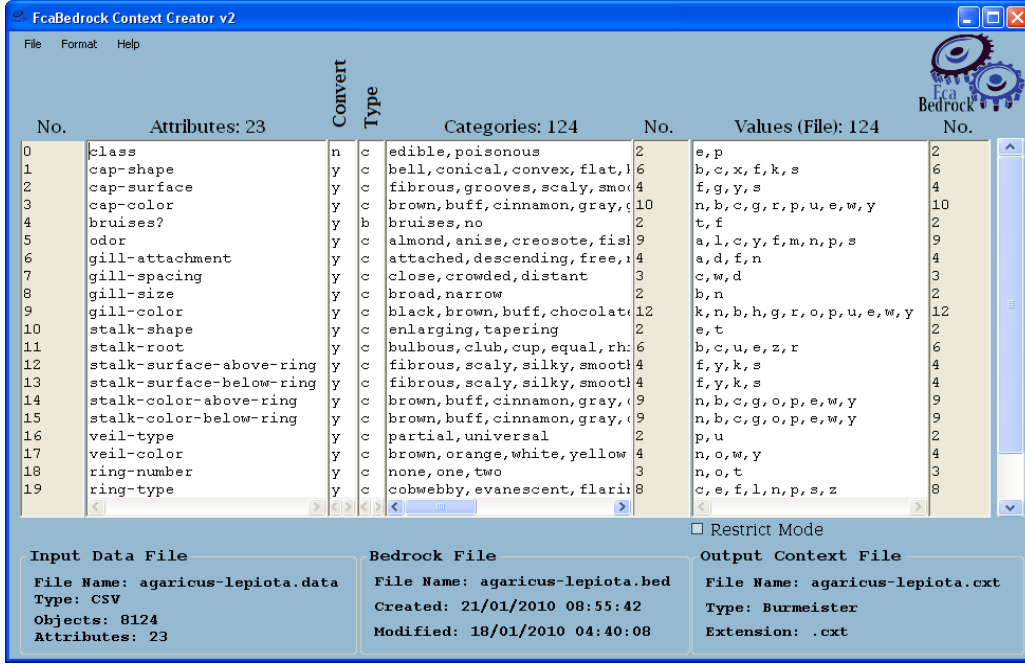


Figure 3. FcaBedrock

C. Continuous Attributes

Continuous attributes are dealt with in FcaBedrock by discretizing the data using user defined ranges. Although hierarchical scaling (e.g. >0 , >10 , >20 ...) is often used in FCA to describe continuous attributes [14], this can lead to dense contexts and, consequently, a large number of large concepts. Hence, in FcaBedrock, a continuous attribute is divided into a number of ranges with disjoint boundaries.

Figure 4 shows the meta-data of the *Adult* data set in FcaBedrock. The *age* attribute is to be converted as a continuous attribute. The boundaries of the ranges are entered as comma separated values. In this case, the boundaries are $<$, 20, 40, 60 and $>$. FcaBedrock will apply these boundaries as the ranges *less than 20*, *20 to less than 40*, *40 to less than 60* and *greater than or equal to 60*, giving a total of four ranges (categories) for the five boundaries. The $<$ and $>$ boundaries are optional; literal end-boundary values will mean that any values outside the overall range will be ignored in the conversion process. The Formal Attribute names are created by concatenating values appropriately; in this case *age<20*, *age20to<40*, *age40to<60* and *age>=60*. During the conversion process, FcaBedrock compares a continuous value read in from the data file with the appropriate boundary values, assigning a *True* value to the corresponding Formal Attribute in the Context. Fractional values are allowed; *height1.567to<2.890*, for

example, would be possible as a Formal Attribute.

V. GUIDED AUTOMATION

A. Exclusion of Attributes and Attribute Values

It may be necessary or desirable to exclude attributes from a Formal Context for a number of reasons: the attribute may be unsuitable for conversion (if it is a free-text attribute for example) or it may be that a particular attribute is not of interest in the analysis being undertaken. The user of FcaBedrock can decide which attributes in the data set to include in the Formal Context using the Convert window. Entering a *y* (for *yes*) will mean that the corresponding attribute will be included in the conversion. Entering an *n* (for *no*) will mean that the corresponding attribute will be excluded from the conversion; no Formal Attributes will be created for this attribute. Individual attribute categories can be excluded from the Formal Context by simply not including them in the list.

Note that these attribute and category exclusions do not exclude *objects* from the Formal Context; all the rows (objects) of the data file will be included in the conversion.

B. Attribute Value Restriction

Sub-contexts can be created by restricting the conversion to user-specified attribute values. By specifying one or more category values of one or more

No.	Attributes: 15	Convert	Type	Categories: 103	No.	Values (File): 103	No.
0	age	y	o	<,20,40,60,>	4	<,20,40,60,>	4
1	workclass	y	c	Private,Self-emp-not-inc,8	8	Private, Self-emp-not-inc	8
2	fnlwgt	n	c		0		0
3	education	y	c	Bachelors,Some-college,11	16	Bachelors, Some-college	16
4	education-num	n	c		0		0
5	marital-status	y	c	Married-civ-spouse,Divorced	7	Married-civ-spouse, Div	7
6	occupation	y	c	Tech-support,Craft-repair,14	14	Tech-support, Craft-rep	14
7	relationship	y	c	Wife,Own-child,Husband,No	6	Wife, Own-child, Husband	6
8	race	y	c	White,Asian-Pac-Islander,5	5	White, Asian-Pac-Island	5
9	sex	y	c	Female,Male	2	Female, Male	2
10	capital-gain	n	c		0		0
11	capital-loss	n	c		0		0
12	hours-per-week	n	c		0		0
13	native-country	y	c	United-States,Cambodia,Eng	41	United-States, Cambodia	41
14	class	n	c	>50K,<=50K	2	>50K, <=50K	2

Figure 4. *Adult* meta-data in FcaBedrock

attributes, the Formal Context will only contain objects which those values. For example, taking the *Mushroom* data set, a sub-context containing only poisonous mushrooms can be created simply by specifying the category value p for the *class* attribute. This can be repeated for any number of attributes. So, for example, a mushroom sub-context could be created containing only poisonous mushrooms that have brown, buff or cinnamon caps.

C. Missing Values

Data sets often contain missing values. Typically these are interpreted as *False* values in FCA, although *missing* more correctly means *unknown*. A symbol such as $?$ is often used in a data set to indicate a missing value and this symbol can be used in FcaBedrock if missing values are of interest in the analysis. If the missing value symbol is included as a category value of an attribute then a corresponding missing value Formal Attribute will be created.

D. Auto-Detection of Meta-Data

If a data file is read without first loading or creating a Bedrock file, FcaBedrock can detect attribute values and create meta-data automatically. It will assume that all attributes are categorical. It will add each new value it finds in a column in the data file to a corresponding list of category values. It will assume that each attribute is to be converted. It will set the category names to the category values. If it finds more than 100 values for an attribute, FcaBedrock will list the first 100 values and indicate that the detection of values for the attribute has been truncated. This is most likely to occur when the attribute being detected is a continuous

one, free-text or some form of ID value. The meta-data obtained through auto-detection can then be edited to provide the required interpretation. If a three-column format input file is being used, attribute names will also be detected.

VI. A MINI-ADULT EXAMPLE

The following is an example adapted from the UCI *Adult* data set. The example uses a data file (see File 1) called *mini-adult.data* with just eight instances and five attributes: *age*, *education*, *employment*, *sex* and *US-citizen*. There are two classes indicating a salary above or below \$50k.

```

39, Bachelors, Clerical, Male, Yes, <=50K
50, Bachelors, Managerial, Female, Yes, <=50K
38, HS-grad, Unskilled, Male, Yes, <=50K
53, 11th, Unskilled, Male, Yes, <=50K
28, Bachelors, Professional, Female, Yes, >50K
37, Masters, Managerial, Female, No, <=50K
49, ?, Clerical, Female, No, <=50K\
52, HS-grad, Managerial, Male, Yes, >50K

```

File 1. *mini-adult.data*

The *mini-adult* meta-data in FcaBedrock are shown in Figure 5. The output *Burmeister* context file, *mini-adult.cxt*, is shown in File 2. The Concept Lattice is shown in figure 6, visualised by inputting the cxt file to a tool called Concept Explorer (ConExp) [19]. Note that the object labels refer to the number of the adult instance (row) in the data file.

VII. CONCLUSION

FcaBedrock has been used successfully to convert large data sets into Formal Contexts: The *Mushroom*

No.	Attributes: 6	Convert Type	Categories: 15	No.	Values (File): 15	No.
0	age	y o	<,30,40,50,>	4	<,30,40,50,>	4
1	education	y c	Bachelors,Masters,11th,HS-	4	Bachelors, Masters, 11t	4
2	employment	y c	Clerical,Managerial,Profe	4	Clerical, Managerial, P	4
3	sex	y c	Male,Female	2	Male, Female	2
4	US-citizen	y b	Yes,No	2	Yes, No	2
5	class	n c	>50K, <=50K	2	>50K, <=50K	2

Figure 5. *mini-adult* meta-data in FcaBedrock

```

B      education-Bachelors
      education-Masters
8      education-11th
15     education-HS-grad
      employment-Clerical
0      employment-Managerial
1      employment-Professional
2      employment-Unskilled
3      sex-Male
4      sex-Female
5      US-citizen
6      .X..X...X...X.X
7      ...XX...X...XX
age<30 .X....X...XX.X
age30to<40 ...X..X...XX.X
age40to<50 X...X...X...XX
age>=50 .X...X...X...X.
      ...X...X...X.
      ...X...X...X.X

```

File 2. *mini-adult.cxt*, *Burmeister* context file.

data set has 8124 objects and around 125 Formal Attributes (depending on the interpretation). The *Adult* data set has 32561 objects and around 106 Formal Attributes and the *Internet Advertisements* data set has 3279 objects and around 1570 Formal Attributes.

FcaBedrock is straightforward to use: the meta-data required is usually easy to obtain (data sets usually come with descriptive documentation) and enter, particularly with the use of the repeat and auto-detection functions. There are only three types of attribute to consider and each type is straightforward to convert.

FcaBedrock is a versatile tool. It currently supports two standard input formats and two popular output formats (the use of *FcaStone* makes other FCA formats possible, too). FcaBedrock gives the user control of the conversion process and freedom to interpret data as seen fit, including the creation of sub-contexts, to suit the needs of analysis. Multiple data sets of the same type (data samples, for example) can easily be converted into Formal Contexts for comparative analysis.

The Formal Contexts created by FcaBedrock are clearly documented by corresponding Bedrock files. These files allow the reproduction of Formal Contexts.

Different interpretations of the same data are possible though the editing of meta-data in FcaBedrock, with a corresponding Bedrock file being created for each interpretation, if desired.

FcaBedrock is an open source project. It may be developed further to support other input and output formats. Additional functionality may be added, such as auto-creation of ranges for continuous attributes as part of auto-detection, grouping of categories and hierarchical scaling of continuous attributes. A more ambitious goal is to incorporate categorisation of free-text values into formal attributes; capturing similarities in free-text values such as addresses from the same town or names in alphabetical order. Such enhancements to the process of Context creation will open further possibilities for FCA to be carried out on existing data sets.

REFERENCES

- [1] Abadi, D.J., Marcus, A., Madden, S.R., Hollenbach, K.: SW-Store: a vertically partitioned DBMS for Semantic Web data management. In: VLDB, vol. 18, pp. 385-406. Springer-Verlag (2009)
- [2] Andrews, S.: In-Close, A Fast Algorithm for Computing Formal Concepts. In: Rudolph, S., Dau, F., Kuznetsov, S.O. (eds.) ICCS 2009, <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-483/paper1.pdf> (2009)
- [3] Andrews, S.: Data Conversion and Interoperability for FCA. In: CS-TIW 2009, pp. 42-49, http://www.kde.cs.uni-kassel.de/ws/cs-tiw2009/proceedings_final_15July.pdf (2009)
- [4] Andrews, S., Orphanides, C., Polovina, S.: Visualising Computational Intelligence through converting Data into Formal Concepts. In the First International Workshop on Emerging Data Technologies for Collective Intelligence (EDTCI-2010), to be held in Fukuoka, Japan, Nov. 4-6, 2010. IEEE Conference Publishing Services, in press
- [5] Asuncion, A., Newman, D.J.: UCI Machine Learning Repository [http://www.ics.uci.edu/\\$sim\\$mlearn/MLRepository.html](http://www.ics.uci.edu/simmlearn/MLRepository.html). Irvine, CA: University of California, School of Information and Computer Science (2007)
- [6] Boulicaut, J-F., Besson, J.: Actionability and Formal Concepts: A Data Mining Perspective. In: Medina, R., Obiedkov, S. (eds.) ICFA 2008. LNCS (LNAI), vol. 4933, pp. 14-31. Springer-Verlag, Berlin/Heidelberg (2008)
- [7] Ganter, B., Wille, R.: Formal Concept Analysis - Mathematical Foundations. Springer-Verlag (1999)
- [8] Goethals, B., Zaki, M.: Advances in Frequent Itemset Mining Implementations: Report on FIMI'03. In: SIGKDD Explorations Newsletter, vol. 6(1), pp. 109-117. ACM New York (2004)

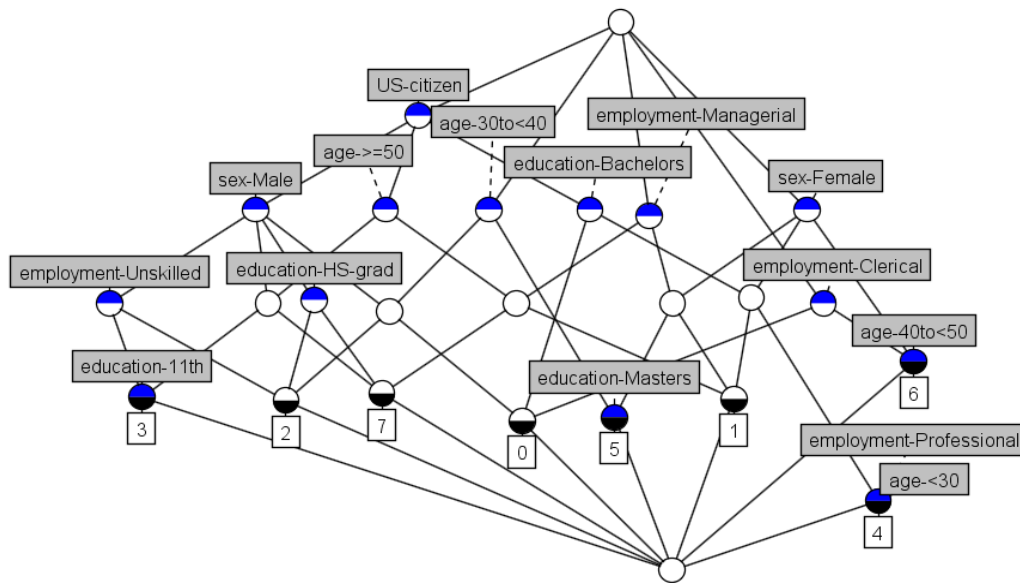


Figure 6. mini-adult Concept Lattice

- [9] Ignatov, D.I., Kuznetsov, S.O.: Frequent Itemset Mining for Clustering Near Duplicate Web Documents. In: Rudolph, S., Dau, F., Kuznetsov, S.O. (eds.) ICCS 2009. LNCS (LNAI), vol. 5662, pp. 185-200. Springer-Verlag, Berlin/Heidelberg (2009)
- [10] Kaytoue-Uberall, M., Duplessis, S., Napoli, A.: Using Formal Concept Analysis for the Extraction of Groups of Co-expressed Genes. In: Le Thi, H.A., Bouvry, P., Pham Dinh, T. (eds.) MCO 2008. CCIS vol. 14, pp. 439-449. Springer-Verlag, Berlin/Heidelberg (2008)
- [11] Krajca, P., Outrata, J., Vychodil, V.: Parallel Recursive Algorithm for FCA. In: Belohlavek, R., Kuznetsov, S.O. (eds.) CLA 2008, pp. 71-82. Palacky University, Olomouc (2008)
- [12] Kuznetsov, S.O., Obiedkov, S.A.: Comparing Performance of Algorithms for Generating Concept Lattices. In: Journal of Experimental and Theoretical Artificial Intelligence, vol. 14, pp. 189-216 (2002)
- [13] Passin, T. B.: Explorer's Guide to the Semantic Web. Manning, Greenwich, CT (2004)
- [14] Priss, U.: Formal Concept Analysis in Information Science. In: Cronin, B. (ed.), Annual Review of Information Science and Technology, vol. 40, p. 521-543 (2006)
- [15] Rioult, F., Robardet, C., Blachon, S., Crémilleux, B., Gandrillon, O., Boulicant, J-F.: Mining concepts from large SAGE gene expression matrices. In: Boulicant, J-F., Dzeroski, S. (eds.) Workshop on Knowledge Discovery in Inductive Databases 2003, pp. 107-118. Catat-Dubrovnik, Croatia (2003)
- [16] Sawase, K., Nobuhara, H., Bede, B.: Visualizing Huge Image Databases by Formal Concept Analysis. In: Barjiela, A., Pedrycz, W. (eds.) Human-Centric Information Processing Through Granular Modelling. Studies in Computational Intelligence, vol. 182. Springer, Berlin/Heidelberg (2009)
- [17] Ganter, B., Wille, R. (1998) *Formal Concept Analysis: Mathematical Foundations*, Springer-Verlag, Berlin. Translated by C. Franzke.
- [18] Wolff, K.E.: A First Course in Formal Concept Analysis: How to Understand Line Diagrams. In: Faulbaum, F. (ed.) SoftStat 1993. Advances in Statistical Software vol. 4, pp. 429-438 (1993)
- [19] Yevtushenko, S. (2006). *ConExp*. Available at: <http://sourceforge.net/projects/conexp>

Visualising computational intelligence through converting data into formal concepts

ANDREWS, S. <<http://orcid.org/0000-0003-2094-7456>>, ORPHANIDES, C. and POLOVINA, S. <<http://orcid.org/0000-0003-2961-6207>>

Available from Sheffield Hallam University Research Archive (SHURA) at:
<http://shura.shu.ac.uk/2720/>

This document is the author deposited version. You are advised to consult the publisher's version if you wish to cite from it.

Published version

ANDREWS, S., ORPHANIDES, C. and POLOVINA, S. (2010). Visualising computational intelligence through converting data into formal concepts. In: XHAFI, F., BAROLLI, L., NISHINO, H. and ALEKSY, M., (eds.) Proceedings of the 2010 international conference on P2P, parallel, grid, cloud and internet computing (3GPCIC). IEEE Computer Society, 302-307.

Copyright and re-use policy

See <http://shura.shu.ac.uk/information.html>

Visualising Computational Intelligence through converting Data into Formal Concepts

Simon Andrews*, Constantinos Orphanides† and Simon Polovina*

Conceptual Structures Research Group, Communication and Computing Research Centre
Faculty of Arts, Computing, Engineering and Sciences, Sheffield Hallam University, Sheffield, UK

*{s.andrews, s.polovina}@shu.ac.uk †corphani@my.shu.ac.uk

Abstract—Formal Concept Analysis (FCA) is an emerging data technology that complements collective intelligence such as that identified in the Semantic Web by visualising the hidden meaning in disparate and distributed data. The paper demonstrates the discovery of these novel semantics through a set of FCA open source software tools *FcaBedrock* and *In-Close* that were developed by the authors. These tools add computational intelligence by converting data into a Boolean form called a Formal Context, prepare this data for analysis by creating focused and noise-free sub-Contexts and then analyse the prepared data using a visualisation called a Concept Lattice. The Formal Concepts thus visualised highlight how data itself contains meaning, and how FCA tools thereby extract data’s inherent semantics. The paper describes how this will be further developed in a project called CUBIST, to provide in-data-warehouse visual analytics for RDF-based triple stores.

Keywords—Formal Concept Analysis, FCA, Formal Context, Formal Concept, visualisation, Concept Lattice, data warehousing, in-warehouse analytics, attributes, objects, Galois connection, Semantic Web, RDF, distributed data, disparate data

I. INTRODUCTION

As its core, the Semantic Web comprises of design principles, collaborative working groups and a variety of enabling technologies [16]. It includes formal specifications such as the *Resource Description Framework (RDF)*, *Web Ontology Language (OWL)* and a variety of data interchange formats, such as *RDF/XML* and *N-Triples* [14]. These technologies provide a formal description of concepts, terms and relationships that capture and integrate meaning with distributed data within a given domain. New data technologies are emerging that can analyse, annotate and visualise such data and promote collective computational intelligence. In this vein, a data analysis method that has been rapidly developed during the past two decades and focuses on knowledge presentation, information management and identifying conceptual structures among semantic data is *Formal Concept Analysis (FCA)*.

II. FORMAL CONCEPT ANALYSIS

Formal Concept Analysis is a term that was introduced by Rudolf Wille in 1984 and builds on “applied lattice and order theory that was developed by Birkhoff and others in

the 1930’s” [18] and was initially developed as a subsection of Applied Mathematics, based on the mathematisation of concepts and concepts hierarchy.

A *Formal Concept* is constituted by its *extension*, comprising of all objects which belong to the Concept, and its *intension*, comprising of all attributes (properties, meanings) which apply to all objects in the extension [18]. The set of all objects and attributes together with their relation to each other form a *Formal Context*, which can be represented by a cross-table [12].

Airlines	Latin America	Europe	Canada	Asia Pacific	Middle east	Africa	Mexico	Caribbean	USA
Air Canada	×	×	×	×	×		×	×	×
Air New Zealand		×		×					×
Nippon Airways		×		×					×
Ansett Australia				×					
Austrian Airlines		×	×	×	×	×			×

The cross-table above is a Formal Context representing the destinations of five airlines, where the elements on the left are formal objects (airlines) and the elements at the top are formal attributes (destinations). If an object has a specific attribute, it is indicated by placing a cross in the corresponding cell of the table. An empty cell indicates that the corresponding object does not have the corresponding attribute. In the Airlines Context, Air Canada flies to Latin America but does not fly to Africa.

A central notion of FCA is a duality called a ‘*Galois connection*’. This connection is often observed between two types of items that relate to each other. A Galois connection implies that “if one makes the set of one type larger, they will be related to a smaller set of the other type, and vice versa” [12]. In the airlines example, the combination of destinations, Asia Pacific, Europe and USA, are flown to by four airlines. If Middle East is added to the list of destinations, the number of airlines reduces to two.

The definition of a Formal Concept is extended by the idea of closure: the extension contains all objects that have the attributes in the intension, and the intension contains

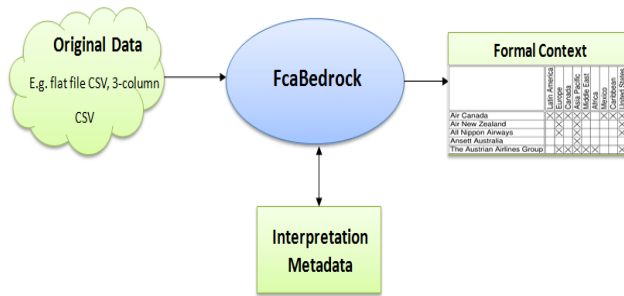


Figure 2. FcaBedrock Process.

they are converted by creating a Formal Attribute for each of the attribute categories [2].

B. *FcaBedrock*

FcaBedrock is a tool for creating Formal Contexts for FCA [3] by converting many-valued data into Boolean data (many-valued attributes into Formal Attribute). By using a process of guided automation, the tool obtains the metadata required for conversion, such as attribute names and attribute types. The metadata are stored, can be edited, used for subsequent conversions and act as a record of how data was interpreted (Figure 2). FcaBedrock currently takes many-column CSV and 3-column CSV files as input, but a version is being developed that also takes RDF-S and OWL formats [11]. The tool can convert categorical (nominal), Boolean and continuous attribute types into Formal Attributes. As opposed to classical FCA, FcaBedrock converts continuous attributes by using disjoint ranges, rather than progressive scaling [19], since ranges such as $0-9$, $10-19$, $20-29$ lead to a less dense Context than the corresponding scales <10 , <20 , <30 and make the size of Concepts in a data set more manageable. Large data sets are easily converted into two popular FCA formats, Burmeister (.cxt) [13] and FIMI³ (.dat).

Creating Formal Contexts using FcaBedrock is straightforward, versatile and reproducible. The tool has a variety of features, which give the user a high degree of control over the conversion process, such as the ability to repeat metadata for similar attributes and the ability to auto-detect the metadata, directly from the input file, if desired. FcaBedrock also provides features for data analysis (or data preparation); it allows the exclusion of attributes from a conversion if they are not of particular interest in, or appropriate for, the analysis. Furthermore, the conversion can be restricted to only those objects with user-specified values.

V. CONCEPT AND LATTICE GENERATION

Several tools exist for visualizing lattices, such as the *ToscanaJ* kit [5], a complete suite of tools for creating

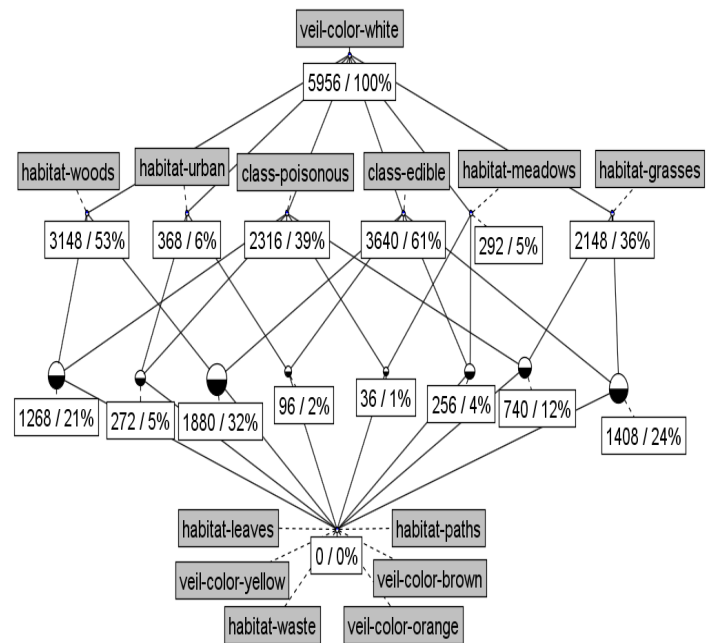


Figure 3. Visualising a *Mushroom* sub-Context in ConExp.

and using Conceptual Information Systems and *ConExp*⁴, a tool for analysing Formal Contexts, exploring dependencies between attributes, counting Formal Concepts and building Concept lattices using Contexts in popular FCA formats as input.

However, as mentioned earlier, the actual usefulness of the lattices produced by lattice visualization software heavily relies on the size of the Formal Contexts given as input. Formal Contexts with a large number of Formal Concepts can result in performance issues and the production of unmanageable, unreadable lattices. FcaBedrock’s data preparation features address these issues, by allowing the creation of sub-Contexts, based on exclusion and restriction criteria set by the user. This means of focusing the analysis can result in significantly smaller and easier to visualise lattices. Figure 3 shows an example of applying the attribute exclusion and restriction features of FcaBedrock to the *Mushroom* data set [4]. The data set originally comprises of 8124 objects, 23 attributes and 125 Formal Attributes⁵. 20 attributes were excluded from the conversion, except from the *class* (poisonous and edible), *veil-color* and *habitat* attributes. Furthermore, the habitat attribute was restricted to *woods*, *urban*, *meadows* and *grasses*. The sub-Context returned 5956 objects and 13 Formal Attributes.

The lattice produced some interesting information. For example, in the sample, mushrooms that live in woods,

⁴<http://sourceforge.net/projects/conexp/>

⁵The number of Formal Attributes varies according to user interpretation of the original data.

³<http://fimi.cs.helsinki.fi/>

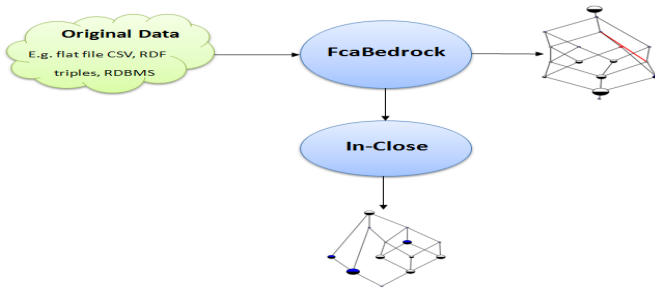


Figure 4. Visualising Formal Contexts using FcaBedrock and In-Close.

meadows, grass and urban areas all have white veils. The proportions of edible to poisonous mushrooms in the various habitats would suggest that woods and meadows are the best places for mushroom pickers to go.

VI. DEALING WITH NOISE

The above example has shown how the production of smaller and easier to visualise lattices is possible by using FcaBedrock. However, this has been achieved by significantly reducing the size of the Formal Context by restricting the data conversion. An alternative approach, that involves all of the data, is to focus the analysis on large Concepts. Small Concepts (noise) can be filtered out of a Context using the In-Close program (Figure 4). In-Close accepts Formal Contexts in the Burmeister (.cxt) format as input, and computes its Concepts [1]. In-Close allows the user to exclude from the computation Concepts with fewer than user-specified numbers of attributes and objects (so-called *minimum support*). After the computation of Concepts, In-Close outputs the same Burmeister file, but with only those Concepts that have the minimum support set by the user. By setting the minimum support high enough so that a relatively small number of Concepts are produced, ‘noise-free’ Contexts can be produced which are easily managed and readable in lattice visualization software.

Figures 5 and 6 show the results in ConExp of minimum support applied to the *Mushroom* data. To compare features of edible and poisonous mushrooms in the sample, two sub-Contexts were created using FcaBedrock; one containing all the edible mushrooms and the other containing all the poisonous ones. Each sub-Context was processed by In-Close to produce a manageable number of Concepts, by setting appropriately large values for minimum support. In each case, the minimum number of attributes in a Concept was set to ten. For the edible mushrooms, the minimum number of objects in a Concept was set to 1900, resulting in a Context containing 17 Concepts. For the poisonous mushrooms, the minimum number of objects in a Concept was set to 885, resulting in a Context containing 14 Concepts. These sizes, although somewhat arbitrary, were set to provide lattices with a similar, readable, number of nodes.

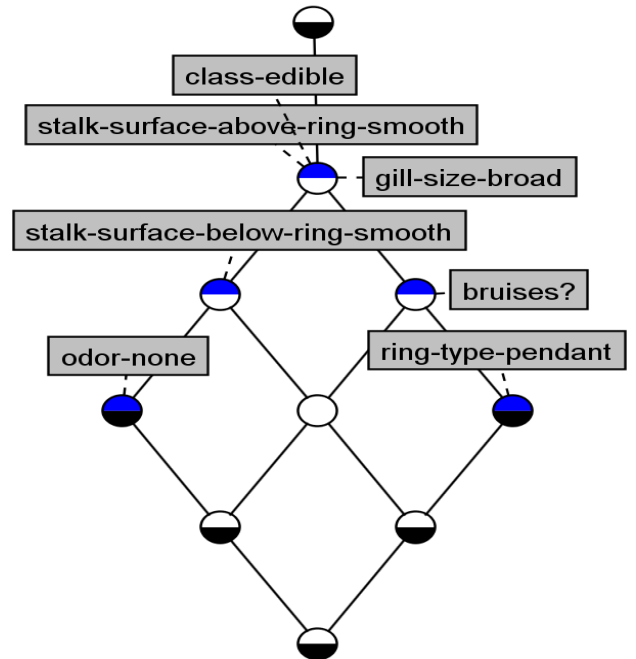


Figure 5. Edible Mushroom Concept Lattice.

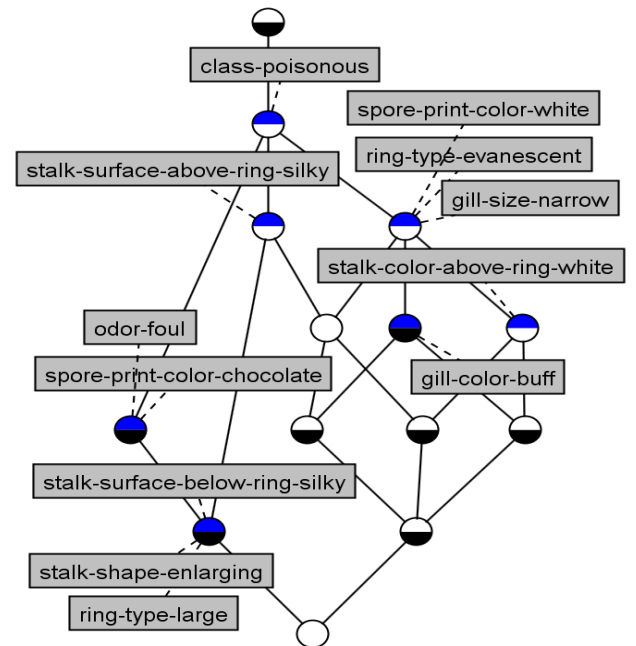


Figure 6. Poisonous Mushroom Concept Lattice.

In each case, the resulting Concepts involved most of the mushrooms in the corresponding ‘noisy’ sub-Context (3334 out of 3916 poisonous mushrooms and 2848 out of 4208 edible mushrooms). The lattices in ConExp were further simplified by hiding attributes that were commonly

supported in both (the purpose here was to highlight the difference between the sub-Contexts, not the similarities). For example, *veil-type-partial* and *ring-number-one* were present in both edible and poisonous Concepts and were removed. This process led to the edible Concept lattice being reduced to ten Concepts. Differences between the two sub-Contexts were now clear. For example, a smooth stalk would seem to indicate that a mushroom is good to eat, whereas those with silky stalks should be avoided. Those with a combination of white spores, an evanescent ring and narrow gills are probably dangerous; safer to try mushrooms with broader gills. Less surprising is the fact that foul smelling mushrooms should probably be left alone; those with no smell are safer. Noting that the number of objects (mushrooms) involved in a Concept decreases as one navigates downwards in a lattice, having a pendant ring is probably only a corroboratory factor in deciding on the wholesomeness of a mushroom. The significance of a mushroom having bruises is interesting, perhaps indicating that edible mushrooms are more likely to show damage from foraging animals.

VII. CUBIST

The further development of this work will form a core part of CUBIST (“Combining and Uniting Business Intelligence with Semantic Technologies”), a research project awarded under the European Unions 7th Framework Programme, 5th ICT call, topic 4.3: Intelligent Information Management; STREP Project No.: FP7 257403. CUBIST aims to develop an approach for Business Intelligence that augments Semantic Technologies with BI capabilities and provides conceptually relevant and user-friendly FCA-based visual analytics. CUBIST will find applications within the Semantic Web and specifically the use of RDF. CUBIST aims to deliver high performance in-warehouse interactive visual analytics for information warehouses and triple stores.

A. Semantic Web and RDF

In the development of the Semantic Web, the use of the RDF schema and triples is proposed, rather than using traditional XML [6]. In XML, the same information can be represented using various structures that all have the same meaning to a person reading them [6], [11], [14]. However, each document can produce different XML trees when parsed by a machine [6]. These are problems which the RDF schema tries to resolve, as RDF gives some standard ways of writing statements so that however it occurs in a document, the same effects can be produced in RDF terms.

FcaBedrock is being developed to accept RDF/XML files as input. RDF uses the *subject-predicate-object* logic, which is the same logic used for the 3-column CSV format currently supported by FcaBedrock. Functionality is to be added for deriving data encoded in RDF vocabularies such

as *Friend of a Friend*⁶ (FOAF) and in authoring ontologies languages such as OWL. By using FcaBedrock as a semantic data preparation tool, FCA can find further applications in the Semantic Web and make knowledge representation, information management and visualizing conceptual structures among semantic data possible.

B. Triple Stores

Triple Stores are column-oriented data warehouses for storing and retrieving RDF metadata using query languages for RDF, such as the *SPARQL Protocol and RDF Query Language*⁷. There is a growth of interest in industry because, using sophisticated indexing techniques, high performance data analysis is possible over billions of triples [15], [17]. As part of CUBIST, FcaBedrock and In-Close will be developed to capture and process disparate and distributed data and be integrated into such triple stores, with the objective of providing ways to visualise and explore hitherto undiscovered BI using FCA (Figure 7).

VIII. CONCLUSION

Using freely available software tools it is possible to visualise hidden meaning in data. FCA is shown to be applicable to large-scale data. As well as the analysis of the Mushroom data set presented here, useful analysis (not presented here) has been carried out on the *Adult* and *Internet Advertisement* data sets from UCI [4] and an internal set of student-related data at Sheffield Hallam University. FcaBedrock is to be presented at the 19th International Conference on Conceptual Structures [3], ICCS 2010, and work to further demonstrate the applicability of the techniques described here is being prepared for other venues, such as the International Conference on Concept Lattices and Their Applications. Further work is required to integrate the processes described here, and to provide intuitive and responsive interfaces to them. Interest at a European level has been demonstrated by the funding of CUBIST, in which this further work can be carried out. CUBIST is bringing together European data warehousing companies, universities with expertise in FCA and commercial use-case partners to develop powerful, insightful and intuitive RDF-based FCA Visual Analytics for BI. Disparate data will come from a range of structured and unstructured sources, providing rich and complex challenges for CUBIST to provide collective intelligence through the use of FCA as an emerging data technology.

REFERENCES

- [1] Andrews, S. (2009). *In-Close, A Fast Algorithm for Computing Formal Concepts*. Available: <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-483/paper1.pdf>. Last accessed 06 May 2010.

⁶<http://www.foaf-project.org/>

⁷<http://www.w3.org/TR/rdf-sparql-query/>

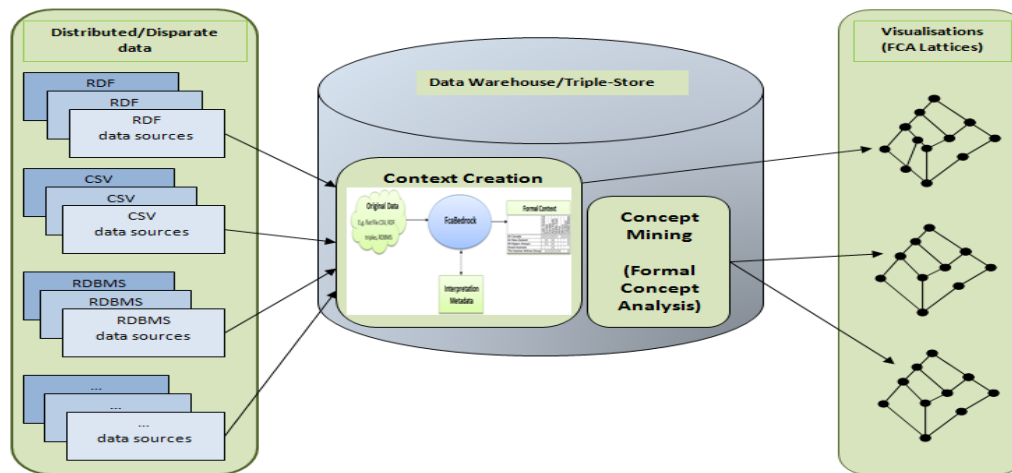


Figure 7. Incorporation of FcaBedrock and In-Close into BI-enabled Triple Stores.

- [2] Andrews, S. (2009). *Data conversion and interoperability for FCA*. In: *Conceptual Structures Tools Interoperability Workshop*, ICCS 2009: Moscow.
- [3] Andrews, S. and Orphanides, C. (2010). *FcaBedrock, a Formal Context Creator*. Submitted to: Croitoru, M., Ferre, S. and Lukose, D. (eds.) *ICCS 2010*, [www.iccs.info] (accepted paper).
- [4] Asuncion, A. and Newman, D.J. (2007). *UCI Machine Learning Repository* [http://www.ics.uci.edu/~mllearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science.
- [5] Becker, P. and Correia, J.H. (2005). *The ToscanaJ Suite for Implementing Conceptual Information Systems*. In *Formal Concept Analysis*, LNCS, Vol. 3626, pp. 324-348, Springer Berlin / Heidelberg.
- [6] Berners-Lee, T. (1998). *Why RDF model is different from the XML model*. Available: <http://www.w3.org/DesignIssues/RDF-XML>. Last accessed 10 May 2010.
- [7] Imberman, S. and Domanski, B. (1999). *Finding Association Rules from Quantitative Data using Data Booleanization*. Available: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.14.4447&rep=rep1&type=pdf>. Last accessed 11 May 2010.
- [8] Jin, R. Breitbart, Y. and Muoh, C. (2009). *Data discretization unification*. In: *Knowledge and Information Systems*. 19(1), pp. 1-29.
- [9] Krajca, P., Outrata, J., Vychodil, V. (2008) *Parallel Recursive Algorithm for FCA*. In: Belohlavek, R., Kuznetsov, S.O. (eds.), *Proceeding of the Sixth International Conference on Concept Lattices and their Applications*, pp. 71-82, Palacky University, Olomouc.
- [10] Kaytoue-Uberall, M., Duplessis S. and Napoli, A. (2008). *Using Formal Concept Analysis for the Extraction of Groups of Co-expressed Genes*. In: Le Thi, H.A., Bouvry, P., Pham Dinh, T. (eds.) *MCO 2008*. CCIS vol. 14, pp. 439-449. Springer-Verlag, Berlin/Heidelberg.
- [11] Passin, T. B. (2004). *Explorers Guide to the Semantic Web*. Manning, Greenwich: CT.
- [12] Priss, U. (2008). *Formal Concept Analysis in Information Science*. In: Cronin, B. (ed.). *Annual Review of Information Science and Technology*, ASIST, vol. 40.
- [13] Priss, U. (2008) *FcaStone - FCA File Format and Interoperability Software*. In: Croitoru, M., Jaschké, R., Rudolph, S. (eds.), *Conceptual Structures and the Web, Proceedings of the Third Conceptual Structures and Tool Interoperability Workshop*, pp. 33-43.
- [14] Semantic Web. (2010). *The Semantic Web*. Available: http://semanticweb.org/wiki/Main_Page. Last accessed 28 Apr 2010.
- [15] Slezak, D., Wroblewski, J., Eastwood, V. and Synak, P. (2008). *Brighthouse: an analytic data warehouse for ad-hoc queries*. In: *Proceedings of the VLDB Endowment*. vol. 1(2), pp. 1337-1345. ACM Digital Library.
- [16] World Wide Web Consortium. (2010). *Design Issues*. Available: <http://www.w3.org/DesignIssues/>. Last accessed 08 May 2010.
- [17] White, P.W. and French, C.D. (1998). *Database system with methodology for storing a database table by vertically partitioning all columns of the table*. US Patent 5,794,229, August 11, 1998.
- [18] Wille, R. (2005). *Formal Concept Analysis as Mathematical Theory of Concepts*. In: Ganter, B., Stumme, G. and Wille, R. (eds.) *Formal Concept Analysis: Foundations and Applications*. Berlin: Springer. pp. 1-6.
- [19] Wolff, K.E. (1993). *A First Course in Formal Concept Analysis*. Available: http://www.fbmh.h-da.de/home/wolff/Publikationen/A_First_Course_in_Forma_Concept_Analysis.pdf. Last accessed 03 May 2010.

Abstract Formal Concept Analysis (FCA) is an emerging data technology that complements collective intelligence such as that identified in the Semantic Web, by visualising the hidden meaning in disparate and distributed data. The paper demonstrates the discovery of these novel semantics through a set of FCA open source software tools, *FcaBedrock* and *In-Close*, that were developed by the authors. These tools add computational intelligence by converting data into a Boolean form called a Formal Context, prepare this data for analysis by creating focused and manageable sub-contexts and then analyse the prepared data using a visualisation called a Concept Lattice. The Formal Concepts thus visualised highlight how data itself contains meaning, and how FCA tools thereby extract data's inherent semantics. The paper describes how this will be further developed in a project called CUBIST, to provide in-data-warehouse visual analytics for RDF-based triple stores.

Key words: Formal Concept Analysis, FCA, Formal Context, Formal Concept, visualisation, Concept Lattice, data warehousing, in-warehouse analytics, attributes, objects, Galois connection, Semantic Web, RDF, distributed data, disparate data

Visualising Computational Intelligence through converting Data into Formal Concepts

Simon Andrews, Constantinos Orphanides, and Simon Polovina

1 Introduction

As its core, the Semantic Web comprises of design principles, collaborative working groups and a variety of enabling technologies [28]. It includes formal specifications such as the *Resource Description Framework (RDF)*, *Web Ontology Language (OWL)* and a variety of data interchange formats, such as *RDF/XML* and *N-Triples* [24]. These technologies provide a formal description of concepts, terms and relationships that capture and integrate meaning with distributed data within a given domain. New data technologies are emerging that can analyse, annotate and visualise such data and promote collective computational intelligence. In this vein, a data analysis method that has been rapidly developed during the past two decades for knowledge representation, information management and identifying conceptual structures in semantic data is *Formal Concept Analysis (FCA)*.

2 Formal Concept Analysis

Formal Concept Analysis (FCA) was introduced in the 1990s by Rudolf Wille and Bernhard Ganter [12], building on applied lattice and order theory developed by Birkhoff and others in the 1930s. It was initially developed as a subsection of Applied Mathematics based on the mathematisation of concepts and concepts hierarchy, where a concept is constituted by its *extension*, comprising of all objects which belong to the concept, and its *intension*, comprising of all attributes (properties, meanings) which apply to all objects of the extension [30]. The set of objects and

Simon Andrews · Constantinos Orphanides · Simon Polovina
Conceptual Structures Research Group, Communication and Computing Research Centre
Faculty of Arts, Computing, Engineering and Sciences
Sheffield Hallam University, Sheffield, UK
e-mail: {s.andrews, c.orphanides, s.polovina}@shu.ac.uk

attributes, together with their relation to each other, form a *Formal Context*, which can be represented by a cross table [21].

Airlines	Latin America	Europe	Canada	Asia Pacific	Middle east	Africa	Mexico	Caribbean	USA
Air Canada	×	×	×	×	×		×	×	×
Air New Zealand		×		×					×
Nippon Airways		×		×					×
Ansett Australia				×					
Austrian Airlines		×	×	×	×	×			×

The cross-table above shows a formal context representing destinations for five airlines. The elements on the left side are formal objects; the elements at the top are formal attributes. If an object has a specific property (formal attribute), it is indicated by placing a cross in the corresponding cell of the table. An empty cell indicates that the corresponding object does not have the corresponding attribute. In the Airlines context above, Air Canada flies to Latin America (since the corresponding cell contains a cross) but does not fly to Africa (since the corresponding cell is empty). However, an empty cell might also mean that it is unknown whether the corresponding object has the corresponding attribute [31].

In mathematical terms, a formal context is defined as a triple $\mathbb{K} := (G, M, I)$, with G being a set of objects, M a set of attributes and I a relation defined between G and M . The relation I is understood to be a subset of the cross product between the sets it relates, so $I \subseteq G \times M$. If an object g has an attribute m , then $g \in G$ relates to m by I , so we write $(g, m) \in I$, or gIm . For a subset of objects $A \subseteq G$, a derivation operator $'$ is defined to obtain the set of attributes, common to the objects in A , as follows:

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

Similarly, for a subset of attributes $B \subseteq M$, the derivation operator $'$ is defined to obtain the set of objects, common to the attributes in B , as follows:

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

Now, a pair (A, B) is a *Formal Concept* in a given formal context (G, M, I) only if $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. The set A is the extent of the concept and the set B is the intent of the concept. A formal concept is, therefore, a closed set of object/attribute relations, in that its extension contains all objects that have the attributes in its intension, and the intension contains all attributes shared by the objects in its extension. In the Airlines example, it can be seen from the cross-table that Air Canada and Austrian Airlines fly to both USA and Europe. However, this does not

constitute a formal concept because both airlines also fly to Asia Pacific, Canada and the Middle East. Adding these destinations completes (closes) the formal concept:

$$(\{Air\ Canada, Austrian\ Airlines\}, \{Europe, USA, Asia\ Pacific, Canada, Middle\ East\}).$$

Another central notion of FCA is a duality called a ‘Galois connection’, which is often observed between items that relate to each other in a given domain, such as objects and attributes. A Galois connection implies that “if one makes the sets of one type larger, they correspond to smaller sets of the other type, and vice versa” [21]. Using the formal concept above as an example, if Africa is added to the list of destinations, the set of airlines reduces to $\{Austrian\ Airlines\}$.

The Galois connections between the formal concepts of a formal context can be visualized in a *Concept Lattice* (Figure 1), which is an intuitive way of discovering hitherto undiscovered information in data and portraying the natural hierarchy of concepts that exist in a formal context.

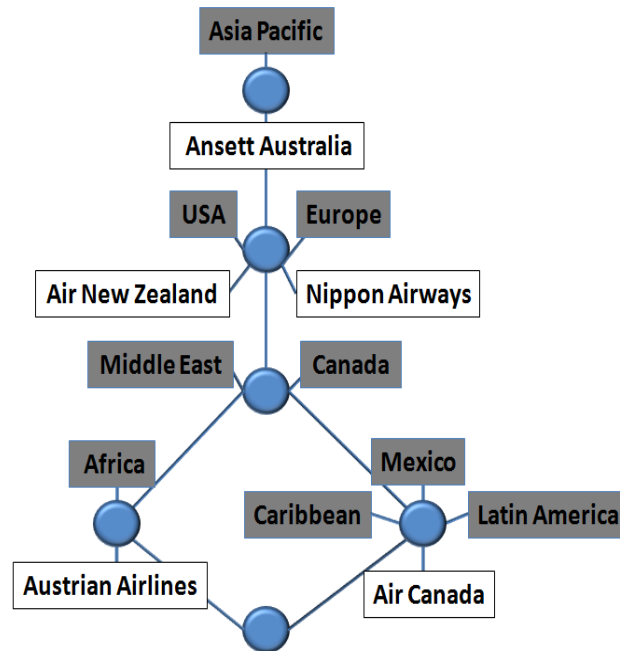


Fig. 1 A Lattice corresponding to the Airlines context.

A concept lattice consists of the set of concepts of a formal context and the subconcept-superconcept relation between the concepts [21]. The nodes in Figure 1 represent formal concepts. Formal objects are noted slightly below and formal attributes slightly above the nodes, which they label.

A concept lattice can provide valuable information when one knows how to read it. As an example, the node which is labeled with the formal attribute ‘Asia Pacific’ shall be referred to as *Concept A*. To retrieve the extension of Concept A (the objects which feature the attribute ‘Asia Pacific’), one begins from the node where the attribute is labeled and traces all paths which lead down from the node. Any objects one meets along the way are the objects which have that particular attribute. Looking at the lattice in Figure 1, if one takes the attribute ‘Asia Pacific’ and traces all paths which lead down from the node, one will collect all the objects. Thus Concept A can be interpreted as ‘All airlines fly to Asia Pacific’. Similarly, the node which is labeled with the formal object ‘Air New Zealand’ shall be referred to as *Concept B*. To retrieve the intension of Concept B (the attributes of ‘Air New Zealand’), one begins by the node where the object is labeled and traces all paths which lead up from the node. Any attributes one meets along the way, are the attributes of that particular object. Looking at the lattice once again, if one takes the object ‘Air New Zealand’ and traces all paths which lead up from the node, one will collect the attributes USA, Europe, and Asia Pacific. This can be interpreted as ‘The Air New Zealand airline flies to USA, Europe and Asia Pacific’. As a further example, the formal concept involving Air Canada and Austrian Airlines, from above, can be clearly seen in the concept lattice as the third node down from the top of the lattice.

Although the Airline context is a small example of FCA, visualising the formal context clearly shows that concept lattices provide richer information than by looking at the cross-table alone.

2.1 Formal Concept Analysis Scaling

In data terms, formal contexts are Boolean data. Of course data predominantly exists in non-Boolean form, so FCA introduces non-Boolean attributes via many-valued contexts. In FCA, *conceptual scaling* is used to transform many-valued contexts into single-valued contexts. Each non-Boolean attribute is given a *scale*, each scale being a context itself. In [31], an example is given of a table containing the sex and age of eight persons. In order to convert this many-valued context into a formal context, two scales (contexts) were created; the first scale represented their gender, containing ‘m’ for male and ‘f’ for female as possible values. The second scale represented their age, but instead of having the actual ages for each person, five formal attributes (the so called scale attributes) were produced; ‘<18’, ‘<40’, ‘<=65’, ‘>65’ and ‘>=80’. The first object is 21 years old, so the object has the ‘<40’ and ‘<=65’ formal attributes, as the object is both younger than 40 and younger than 65 (but not younger than 18, nor older than 65 or 80). The seventh object is 90 years old, so the object has the ‘>65’ and ‘>=80’ formal attributes, as the object is both older than 65 and 80 (but not younger than 18, 40 or 65). The two scales were then merged to form the complete formal context representing ‘the age and gender of eight persons’ (Figure 2). These notions are central to the process of formal context creation.

	sex	age
ADAM	m	21
BETTY	f	50
CHRIS	?	66
DORA	f	88
EVA	f	17
FRED	m	?
GEORGE	m	90
HARRY	m	60

	sex		age				
	m	f	<18	<40	<=65	>65	>=80
ADAM	X			X	X		
BETTY		X			X		
CHRIS						X	
DORA		X				X	X
EVA		X	X	X	X		
FRED	X						
GEORGE	X					X	X
HARRY	X				X		

Fig. 2 The transformation of Data into contexts (after [31]).

Figure 3 shows the age and gender context as a concept lattice, visualised in an open source FCA tool called *Concept Explorer* (ConExp) [32].

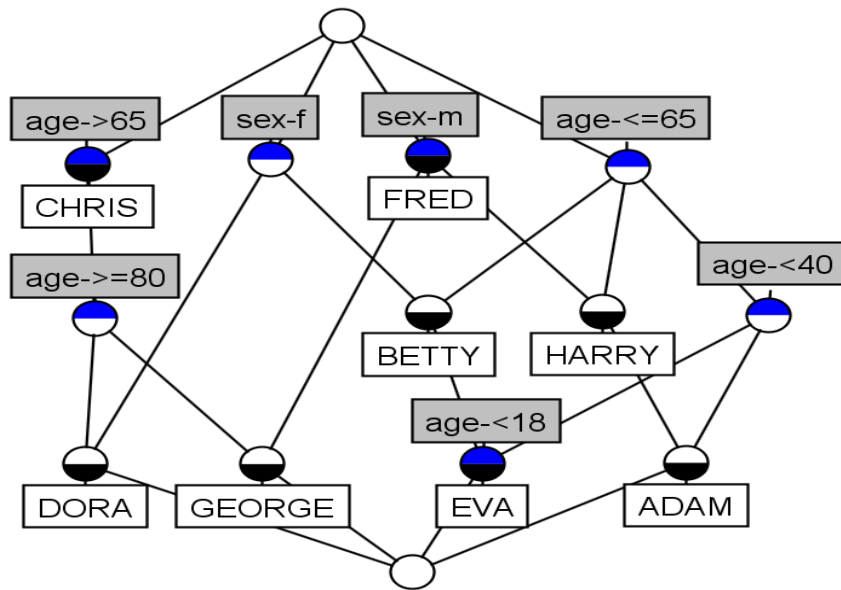


Fig. 3 The Concept Lattice for the 'age and gender' context (after [31]).

2.2 Formal Context Formats

To allow interoperability between FCA tools and applications, a number of file formats have been developed to represent formal contexts [23], the most popular being the *Burmeister* format, used for example by ConExp. As such, it is a sensible choice for developing new FCA tools. Outside of the FCA community, similar data analysis is carried out in the data mining domain [33]. A popular file format for data in this area is the *Frequent Itemset Mining Implementations* (FIMI) format [10]. This is therefore an appropriate choice of data file format to cater for if the interoperability of tools is to be extended beyond FCA. The two formats are briefly described below.

2.2.1 Burmeister (.cxt)

A Burmeister file begins with the letter ‘B’, to denote it is a Burmeister file (although the authors are unsure of its exact significance; it has merely become a convention to place this letter at the beginning of these files). It is followed by the number of objects, followed by the number of attributes and then lists the objects, followed by the attributes. It then stores the body of the formal context as a grid, using crosses for True values and dots for False values. File 1 is a cxt file for the ‘age and gender’ context and was used by ConExp to generate the concept lattice in Figure 3. In this figure the upper half-node when shaded means that the concept has its own attributes (i.e. they are not inherited). The lower half-node when shaded means that the concept has its own objects.

```

B
8
7

ADAM
BETTY
CHRIS
DORA
EVA
FRED
GEORGE
HARRY
sex-m
sex-f
age-<18
age-<40
age-<=65
age->65
age->=80
X..XX..
.X..X..
.....X.
.X...XX

```

```
.XXXX..
X.....
X....XX
X...X..
```

File 1 age-gender.cxt, *Burmeister* context file.

2.2.2 FIMI (.dat)

The *Frequent Itemset Mining Implementations* (FIMI) [10] data format (.dat) is used in data mining, particularly in testing the efficiency of algorithms [13]. As opposed to Burmeister, a FIMI file only consists of rows of numbers; each row represents an object and each number represents a formal attribute. The ordering of the attributes is as one would expect from the formal context, taking the first column of the context to be attribute one and so on. File 2 represents the body of the ‘age and gender’ formal context in the FIMI format.

```
1 4 5
2 5
6
2 6 7
2 3 4 5
1
1 6 7
1 5
```

File 2 age-gender.dat, *FIMI* file.

3 Large Data Sets

Going beyond the simple example above, it has been shown that FCA can be usefully applied to large sets of data and that FCA has applications in data mining [19]. Rather than *X*-sized data it can handle *XX*-sized data. Programs, such as *In-Close* [1], exist which are capable of handling and processing large formal contexts. However, issues arise when trying to visualise the concept lattice. Formal contexts that have tens of attributes and thousands of objects can easily contain tens, if not hundreds of thousands of formal concepts, as work on parallel processing such volumes has identified [18]. The *Mushroom* data set [6], for example, has 23 attributes (properties of mushrooms) and 8124 objects (mushrooms). The formal context that results from the data set contains over 220,000 formal concepts. Lattice visualisation software does not exist that can compute lattices with such large numbers of nodes. Even if such tools existed, the results would be highly complex and unreadable, unless a sophisticated means of managing the lattice was employed.

Issues also arise when acquiring data for FCA; as described above, most tools for FCA require data to be in a formal context format. Clearly, the majority of typical data sets are not in this format, making accessibility to FCA difficult. To achieve interoperability with FCA tools and applications, data sets first need to be converted. In the process of conversion, different existing data set formats require different treatment, as do different data types. Creating the `age-gender.cxt` file was relatively convenient as the formal context already existed and was comparatively small. Converting large data sets into formal contexts and their corresponding `cxt` files would require additional effort to get them into a suitable form.

More problems arise when attributes and attribute values can be interpreted in different ways, having, as a result, the production of inconsistent conversions. For example, in choosing scales for continuous values or in dealing with missing values different approaches may be taken [2]. This can lead to apparently conflicting analyses of the same data and make the comparison of FCA tools and algorithms more difficult.

Another issue is the fact that disparate and distributed data do not, by definition, exist in a unified form. It is possible to regard a formal context as a unifying form, but existing data need to be converted into formal contexts, first, in order for FCA to be carried out.

As we shall explicate later on, data set conversion for FCA is conducted by *discretising* and *Booleanising* data; that is to take each many-valued attribute in a data set and convert it into as many Boolean attributes as it has values and by scaling continuous values using ranges. Although some issues of data discretisation and Booleanisation are understood (see FCA scaling, above), more work was required in this area to consider data interpretation, large scale data and automated conversion, and tools do not exist that facilitate these processes for FCA.

Thus, the task of converting data into formal contexts can be time consuming, is open to interpretation and usually requires a programming element to cope with large data sets. *FcaBedrock*, is a tool that has been developed to carry out this process [4], and is described later on.

4 Interpreting Data for FCA

Whereas FCA has used the term many-valued attribute to encompass such things as age and gender, such data may be distinguished by considering gender to be a *nominal* or *categorical* attribute [6], with the categories *male* and *female*, whilst age may be described as being a continuous attribute. Both are many-valued, but each needs to be treated differently in interpretation and conversion. The distinguishing characteristics of categorical and continuous are useful when considering these processes and how they might be automated. Furthermore, the process of discretising applies most clearly to continuous attributes, whereas the process of Booleanising is that of creating True/False attributes from many-valued ones, whether those values be categories or discretised values.

4.1 Data Discretisation

Data discretisation is defined as “a process of converting continuous data attribute values into a finite set of intervals with minimal loss of information” [17] and with simpler terminology as the process of scaling continuous values using ranges [4]. Data mining tasks often involve dealing with continuous attributes and this can decrease performance [17]. This can be resolved by producing discretised values of a continuous attribute and replacing it with the new values.

An example of data discretisation could be student grades for a module. Each student can have any grade value from 0 to 100 (including decimal values). Discretising the attribute ‘*grade*’ can be accomplished by producing ranges and assigning each student grade to one of the produced ranges; any grades smaller than 40% could be categorized as ‘*FAIL*’ and any grades equal or higher than 40% as ‘*PASS*’. This approach makes continuous attributes easier to handle and increases the performance of mining tasks.

Note that the notion of ‘categories’ may still be applied to a discretised continuous attribute, however in this instance referring to the ranges or boundary values used in the discretisation.

4.2 Data Booleanisation

Data Booleanisation is the process of taking each many-valued attribute in a data set and converting it into as many Boolean attributes as it has values [16]. This is also the approach used to convert many-valued attributes in FCA, as they are converted by creating a formal context attribute for each of the values [2].

The gender example above, is an instance of Booleanisation. Another example would be considering the attribute *eye-colour*, which might be converted into the formal attributes *eye-colour-blue*, *eye-colour-brown*, *eye-colour-green*, *eye-colour-grey*. Rarer colours could be captured by *eye-colour-other* and missing values by *eye-colour-missing*.

4.3 FcaBedrock

FcaBedrock is a tool for creating Formal contexts for FCA [4] by converting many-valued data into formal contexts (many-valued attributes into formal attributes). The tool is an open-source project at *Sourceforge* [5]. By using a process of *guided automation*, the tool obtains the metadata required for conversion, such as attribute names and attributes types, and converts corresponding data sets, using these metadata, into formal context files. The term *guided automation* refers to the fact that the process of conversion is automatic but the user first has to provide guidance as to how the data set should be interpreted. Further automation is provided in that

FcaBedrock can be used to examine a data file to automatically determine certain types of metadata (see below) and thus create a basic interpretation and corresponding formal context without use guidance.

A separate file, called a Bedrock file, is used by FcaBedrock to store the metadata. This can be edited directly in or outside of the tool, used for subsequent conversions and acts as a record of how a data set was interpreted (Figure 4).

The tool currently takes many-column CSV and 3-column CSV data files as input, but a version is being developed that also takes RDF-S and OWL formats [20]. Large data sets are easily converted by FcaBedrock into the two popular FCA formats, Burmeister (.cxt) [22] and FIMI (.dat).

Figure 5 shows FcaBedrock with metadata for the *Mushroom* data set (a publicly available data set from the UCI Machine Learning Repository [6]). The following can be seen: the attribute numbers; the names of the attributes; whether they will be converted as part of a formal context; their type (in this case, all but one of the mushroom attributes are categorical); their categories and the actual category values as they appear in the data file (in this case, nominal letters are used in the mushroom data file to represent the various categories of each attribute). The names of the categories were obtained from a data description document accompanying the data file. It is often the case that data sets are accompanied with a description of the data and such descriptions are a key source of metadata.

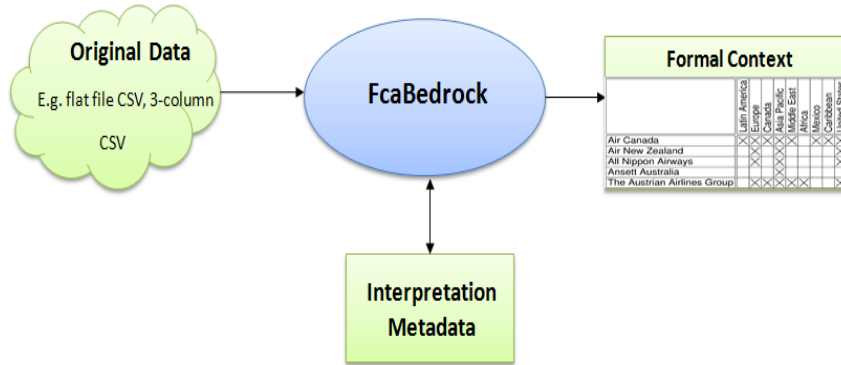
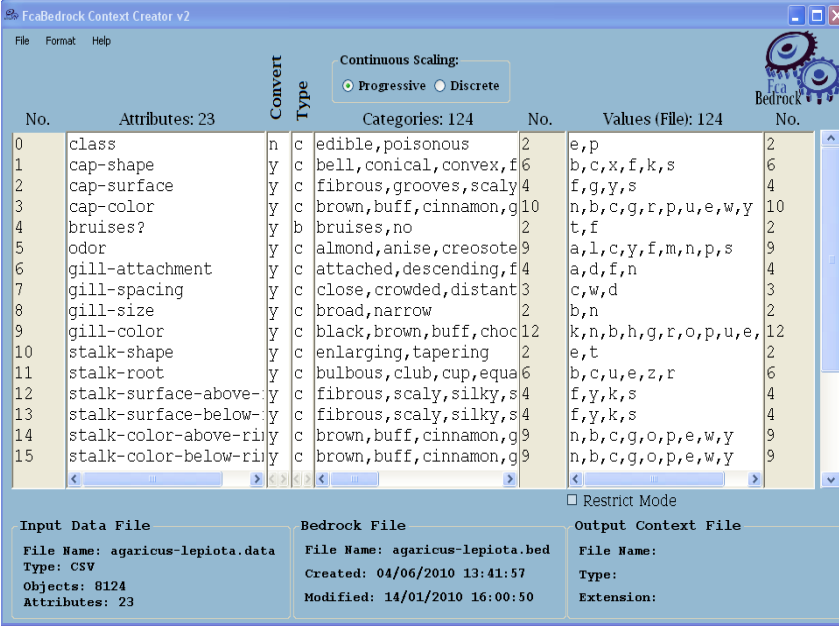


Fig. 4 FcaBedrock Process.

4.3.1 Attribute Types

FcaBedrock uses three types of attribute: categorical, Boolean and continuous. These types are represented in FcaBedrock by the letters 'c', 'b' and 'o', respectively.



No.	Attributes: 23	Convert Type	Categories: 124	No.	Values (File): 124	No.
0	class	n	c edible,poisonous	2	e,p	2
1	cap-shape	y	c bell,conical,convex,f	6	b,c,x,f,k,s	6
2	cap-surface	y	c fibrous,grooves,scaly	4	f,g,y,s	4
3	cap-color	y	c brown,buff,cinnamon,g	10	n,b,c,g,r,p,u,e,w,y	10
4	bruises?	y	b bruises,no	2	t,f	2
5	odor	y	c almond,anise,creosote	9	a,l,c,y,f,m,n,p,s	9
6	gill-attachment	y	c attached,descending,f	4	a,d,f,n	4
7	gill-spacing	y	c close,crowded,distant	3	c,w,d	3
8	gill-size	y	c broad,narrow	2	b,n	2
9	gill-color	y	c black,brown,buff,choc	12	k,n,b,h,g,r,o,p,u,e	12
10	stalk-shape	y	c enlarging,tapering	2	e,t	2
11	stalk-root	y	c bulbous,club,cup,equa	6	b,c,u,e,z,r	6
12	stalk-surface-above-	y	c fibrous,scaly,silky,s	4	f,y,k,s	4
13	stalk-surface-below-	y	c fibrous,scaly,silky,s	4	f,y,k,s	4
14	stalk-color-above-rii	y	c brown,buff,cinnamon,g	9	n,b,c,g,o,p,e,w,y	9
15	stalk-color-below-rii	y	c brown,buff,cinnamon,g	9	n,b,c,g,o,p,e,w,y	9

Input Data File	Bedrock File	Output Context File
File Name: agaricus-lepiota.data	File Name: agaricus-lepiota.bed	File Name:
Type: CSV	Created: 04/06/2010 13:41:57	Type:
Objects: 8124	Modified: 14/01/2010 16:00:50	Extension:
Attributes: 23		

Fig. 5 Metadata of the *Mushroom* [6] data set in FcaBedrock.

Categorical is the typical many-valued attribute and is converted by creating one formal attribute for each of the attribute categories. As an example, a categorical attribute ‘height’ with categories ‘short’, ‘normal’ and ‘tall’ will be converted by FcaBedrock by creating three corresponding formal attributes and naming them by concatenating the attribute and category names; thus ‘height-short’, ‘height-normal’ and ‘height-tall’.

Boolean attributes can be interpreted as a single formal attribute. In a data set, a Boolean attribute typically has two categories that represent True or False. For a Boolean attribute, FcaBedrock uses the first category value as the True value. However, both categorical and Boolean attribute types were dealt with keeping in mind freedom of interpretation. For example, an attribute ‘Married?’ with values ‘Yes’ and ‘No’ can be interpreted as Boolean, where only ‘Married?-Yes’ will appear in the formal context, or as categorical, where both ‘Married?-Yes’ and ‘Married?-No’ will appear in the formal context.

Continuous attributes are dealt with in FcaBedrock by either discretising the data using user defined ranges (e.g. 0-9, 10-19, 20-29), or by progressive scaling (e.g. <10, <20, <30), which is the approach to convert continuous attributes in classical FCA [31] (see above).

4.3.2 Auto-detection of Metadata

FcaBedrock can auto-detect metadata, directly from the input file, if desired. It will initially assume that all attributes are categorical. It will add each new value it finds in a data-column to a corresponding list of category values. It will assume that each attribute is to be converted. It will set the category names to the category values. If FcaBedrock determines that there is a high proportion of different numerical values for an attribute, it will suggest that the attribute is continuous and ask the user if ranges are to be automatically calculated. If so, FcaBedrock will discretise the attribute using five, equal-sized ranges, basing the division on the highest and lowest values detected. If not, FcaBedrock will list the first 100 values and indicate that the addition of new categories for the attribute has been truncated. For non-numerical values (when the attribute being detected is free-text or some form of ID value, for example), FcaBedrock will list the first 100 values and indicate that the addition of new categories for the attribute has been truncated. The metadata obtained through auto-detection can then be edited to provide the required interpretation.

Figure 6 shows auto-detected metadata for the *Adult* data set (also from UCI [6]), which contains US census data of 32,561 adult citizens. The Adult data set is a typical CSV file of data and does not contain the attribute names, only columns of their values. Thus the attribute names are set by FcaBedrock as their attribute number. The user can usually obtain attribute names from the data description that accompanies the data file. Note that several of the attributes have been determined by FcaBedrock to be continuous and suitable discretisation has been automatically applied.

4.3.3 Attribute Exclusion and Restriction

FcaBedrock can be also used as a data preparation, or preprocessing tool, by allowing the exclusion of attributes from a conversion if they are not of particular interest in, or appropriate for, the analysis undertaken. However, these attribute and category exclusions do not exclude objects from the formal context; all the rows (objects) of the data file will be included in the conversion.

On the other hand, sub-contexts can be created by restricting the conversion to user-specified attribute values. By specifying one or more category values of one or more attributes, the formal context will only contain objects with those values.

5 Concept and Lattice Generation

Lattice visualisation is made possible through several FCA tools, such as the *ToscanaJ* kit [8], a complete suit of tools for creating and using Conceptual Information Systems [7] and *ConExp* [32], a tool for analysing formal contexts, exploring

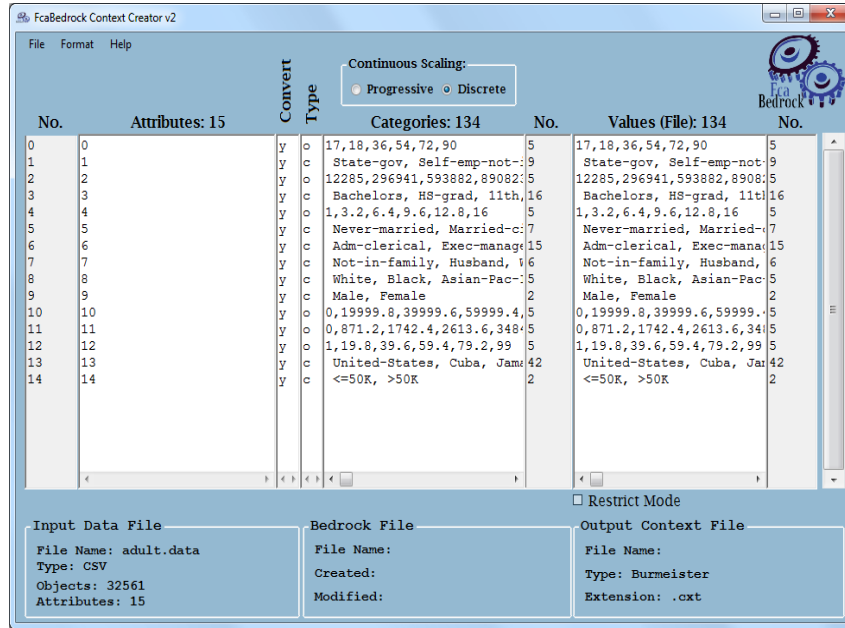


Fig. 6 Auto-detecting the metadata of the *Adult* data set [6] in FcaBedrock.

dependencies between attributes, counting formal concepts and building concept lattices using contexts, in popular FCA formats, as input.

However, as mentioned earlier, the actual usefulness of the lattices that lattice visualisation software produce, rely on the numbers of formal attributes and Formal concepts of the formal contexts given as input; formal contexts with a large number of formal concepts can result in performance issues and the production of unmanageable, unreadable lattices. These issues can be addressed by using FcaBedrock's data preparation features. By providing the ability to create sub-contexts, based on exclusion and restriction criteria set by the user, this means of focusing the analysis can result in smaller, more manageable lattices.

To illustrate an example of producing concept lattices from data sets by using sub-contexts, the *Mushroom* data set [6] will be used. The data set originally comprises of 8124 edible and poisonous mushrooms of the families *agaricus* and *lepiota*. It has 23 attributes describing properties such as edibility, stalk color and ring type (Figure 7).

Figure 8 shows an example of two mushrooms of the *agaricus* family; *Agaricus Arvensis* and *Agaricus Xanthodermus*. While they both resemble edible mushrooms on first sight, *Agaricus Xanthodermus* has poisonous properties. This is a good example of how mushroom identification can become confusing for non-experts who are unable to distinguish differences between similar-looking mushrooms.

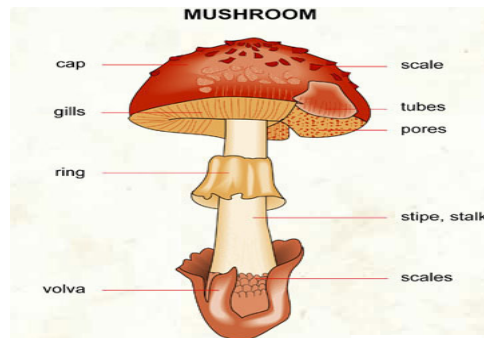


Fig. 7 Parts of a mushroom. (Source: <http://www.infovisual.info>)



Fig. 8 *Agaricus Arvensis* (to the left) and *Agaricus Xanthodermus* (to the right). Source: [<http://www.gracesmall.com/Tees/mushrooms.htm>]

Within this context, Figure 9 shows how the restriction and exclusion capabilities of FcaBedrock can be applied to create a sub-context of the *Mushroom* data set and help determining whether a mushroom is edible or not based on some of its properties. 20 attributes were excluded from the conversion, except from the *class* (poisonous and edible), *cap-color* and *habitat* attributes. Furthermore, the *cap-color* attribute was restricted to *red*, *pink*, *green*, *purple* and the *habitat* attribute was restricted to *woods*, *urban*, *meadows* and *grasses*. The restrictions set returned 996 objects and 19 formal attributes. Key to the analysis is the fact that this sub-context contains few enough formal concepts to make visualisation as a concept lattice practical.

Visualising the sub-context in ConExp (Figure 10) produced some interesting information, particularly as a feature of ConExp has been used to label each concept in the lattice with the object count and percentage of the overall number of objects. For example, in the sample, 98% of all the mushrooms live in woods, out of which 61% of them are safe to eat. The most dominant mushrooms are red-capped, of which 576 are edible and 300 are poisonous. This might indicate that mushroom cap colors are not the safest indicator for determining their edibility. On the other hand, pink-capped mushrooms can be found in woods and sometimes in grasses or meadows and all of them are poisonous. Green-capped and purple-capped are extremely rare, but if one is lucky enough to find some they are safe to eat.

As another illustration of using attribute exclusion, Figure 11 shows an example of a sub-context being created from two continuous attributes (*age* and *hours-per-*

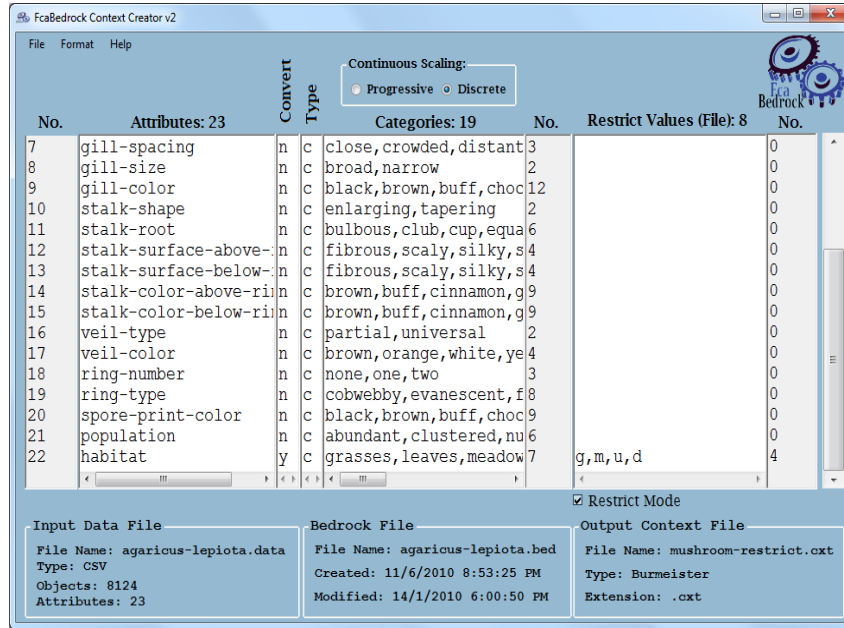


Fig. 9 Creating a *class-habitat-cap color* sub-context from the *Mushroom* data set [6].

week) of the *Adult* US census data set [6], using progressive scaling. The corresponding concept lattice in ConExp is shown in Figure 12. The first scale represents the age of the objects, using the ranges <20 , <40 , <60 and *all*. The second scale represents the hours worked per week, using the ranges <20 , <40 , <60 , <80 and *all*.

Various facts concerning the hours worked per week according to age of the population can be determined by reading the concept lattice. For example, a simple analysis is that 8% of the population work more than 60 hours per week. A more detailed analysis is that about a quarter of young adults (less than 20 years old) work less than 20 hours per week and about three-quarters work less than 40 hours per week. This can be compared to the population as a whole, where only 5% work less than 20 hours and only a quarter work less than 40 hours.

6 Dealing with Concept Simplification and Complexity

The examples in Figures 10 and 12 have shown how FcaBedrock's features can be used to produce smaller and easier to visualise lattices. However, this has been achieved by significantly reducing the size of the formal context by restricting the data conversion to objects matching certain criteria set by the user. An alternative ap-

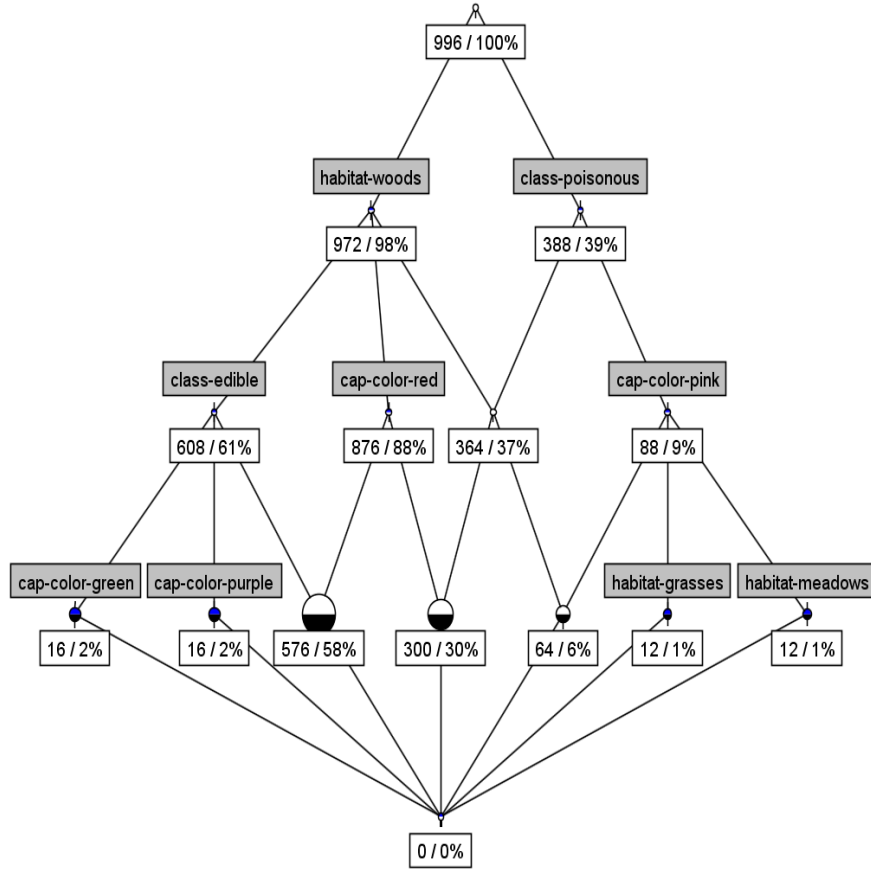


Fig. 10 Visualising the mushroom *class-habitat-cap color* sub-context, created by FcaBedrock, in ConExp.

proach, that involves all of the data, is to focus the analysis on large concepts. Small concepts might not include a significant, to the analysis, amount of objects/attributes and can add unnecessary complexity to the lattice. Filtering out these concepts from a context can thus allow the production of lattices which contain all the data, but can still produce useful and usable information. A tool that can filter out concepts from a context is *In-Close* [3]. *In-Close* accepts as input formal contexts in the Burmeister (.cxt) format and computes its concepts [1]. *In-Close* allows the user to exclude from the computation concepts with fewer than user-specified numbers of attributes and objects (the so-called *minimum support for intent* and *minimum support for extent*). After computing the concepts, *In-Close* outputs the same Burmeister file, but with only those concepts that have the minimum support set by the user (Figure 13). This is an approach similar to the *Iceberg Concept Lattices* of Stumme et al [27],

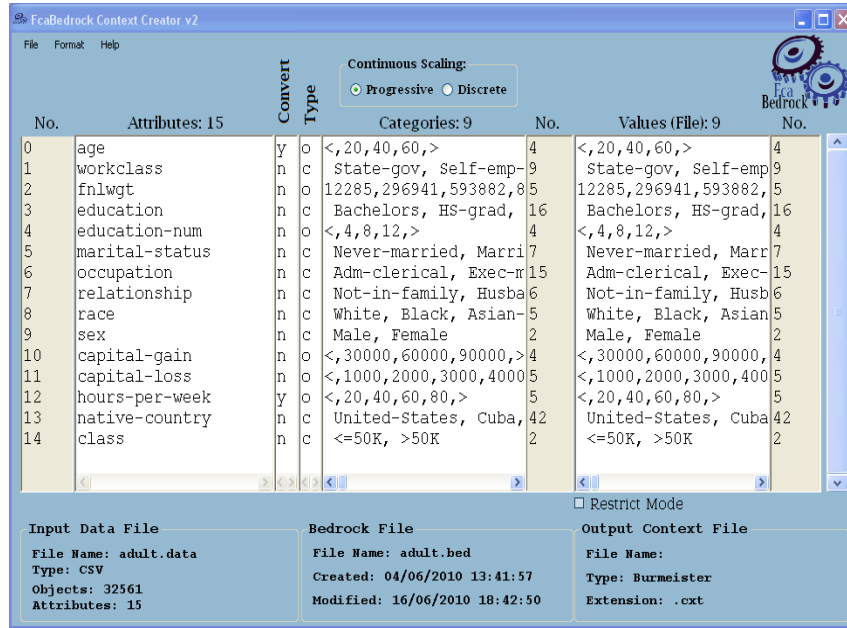


Fig. 11 Creating an *age-hours per week* sub-context from the *Adult* data set [6].

although they do not consider minimum support for intent and do not produce a new context; rather their approach is to truncate the concept lattice by removing nodes with fewer objects than a specified number. They also leave the analysis of iceberg lattices as an open question for further research.

Using In-Close, the minimum support can be set high enough so that a relatively small number of concepts are computed. In this way, simplified contexts can be produced which result in readable concept lattices (Figure 14). Combining the exclusion and restriction capabilities of FcaBedrock with the minimum support features of In-Close adds further possibilities in the visualisation of computational intelligence (Figure 15).

In order to compare features of edible and poisonous mushrooms in the *Mushroom* data set, FcaBedrock was used to create two sub-contexts; one containing all the edible mushrooms and one containing all the poisonous ones. This resulted in one sub-context containing 4208 edible mushrooms and one sub-context containing 3916 poisonous mushrooms. Each sub-context was then processed by In-Close to produce a manageable number of concepts, by setting appropriately large values for minimum support. Figure 16 is a command-line screen shot of In-Close being used to compute all 92,543 concepts in the edible mushroom sub-context (i.e. with no minimum support specified). When the minimum number of attributes was set to 10 and the minimum number of objects was set to 1500, In-Close computed only 9 formal concepts. For the poisonous mushroom sub-context, when the minimum

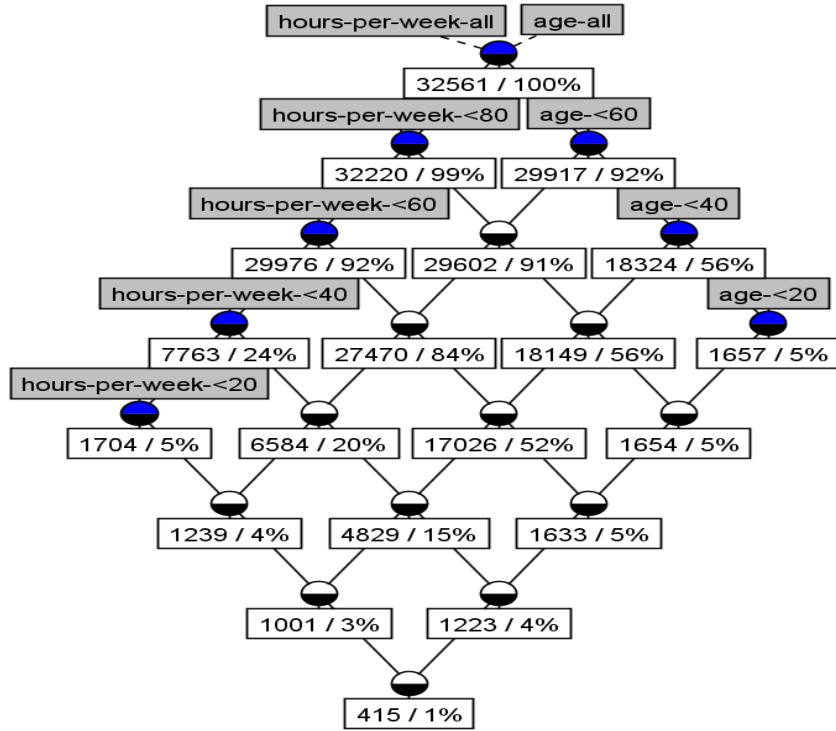


Fig. 12 Visualising the US census *age-hours per week* sub-context, created by FcaBedrock, in ConExp.

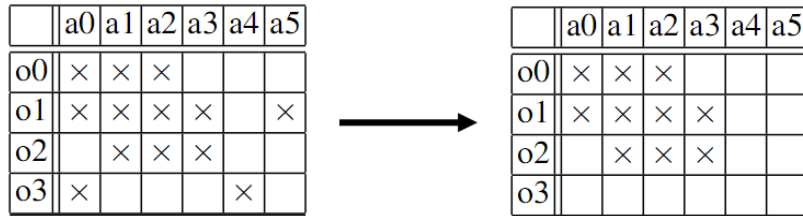


Fig. 13 Simplifying a context using *In-Close*.

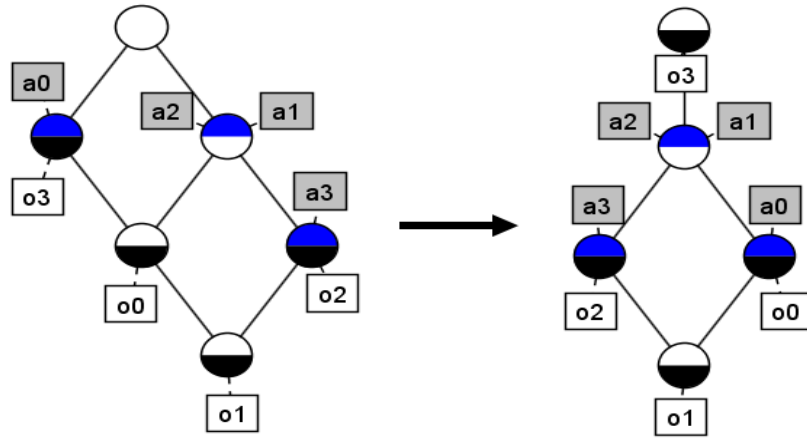


Fig. 14 Visualising the original (to the left) and simplified (to the right) contexts using *ConExp*.

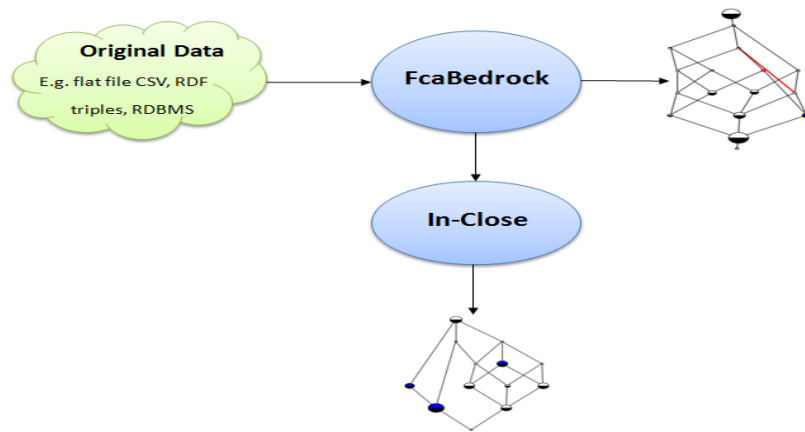


Fig. 15 Visualising formal contexts using *FcaBedrock* and *In-Close*.

number of attributes was set to 9 and the minimum number of objects was set to 1000, this resulted in 7 formal concepts. Although somewhat arbitrary, these sizes were arrived at to provide lattices with a similar small number of formal concepts. Figures 17 and 18 show the resulting concept lattices in *ConExp*.

In both cases, the resulting concepts involved most of the mushrooms in the corresponding 'pre-simplified' sub-context (2368 out of 4208 edible mushrooms and 3344 out of 3916 poisonous mushrooms). The attributes featured by both edible and poisonous mushrooms lattices (such as *veil-color-white*, *ring-number-one* and *gill-attachment-free*) were hidden, as the purpose was to highlight the difference

```

***** In-Close 2.0 Concept Miner *****

Enter cxt file name including extension: edible.cxt
Enter minimum size of intent (no. attributes): 0
Enter minimum size of extent (no. objects): 0
Reading data...Done.
n: 126
m: 4208
x's: 94608
Mining concepts...Done.
Number of concepts: 92543
Output concepts to file? (y/n): n
Outputting context file...Done.
Hit <enter> to finish

```

Fig. 16 *In-Close* computing all Concepts in the edible mushroom context.

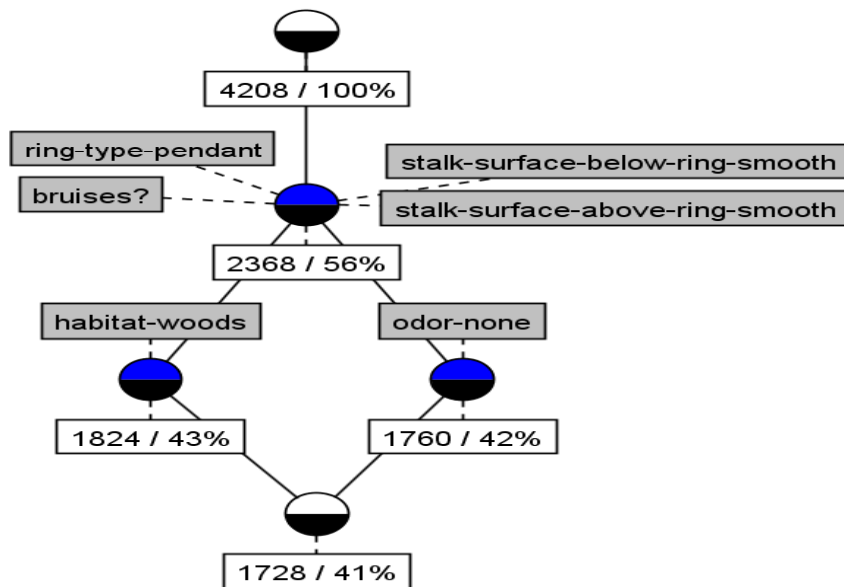


Fig. 17 Edible Mushroom Concept Lattice.

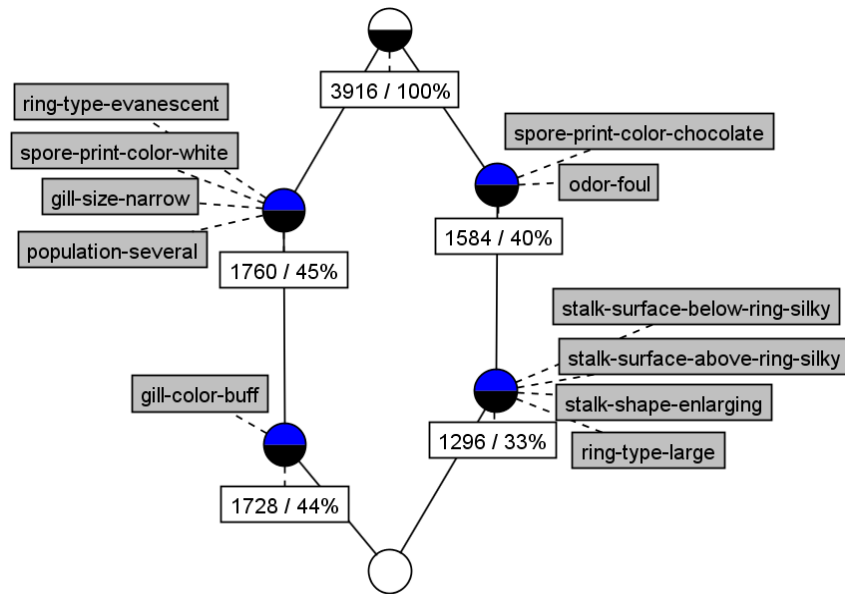


Fig. 18 Poisonous Mushroom Concept Lattice.

between the sub-contexts, not the similarities. This also resulted in the formal concepts of the edible mushrooms reducing to 5. Hiding the common attributes that were commonly supported in both lattices gave a clearer overview between the differences of the two sub-contexts. For example, a smooth stalk would seem to indicate that a mushroom was safe to eat, whereas those with silky stalks should be avoided. Those with a white spore print, a narrow gill and an evanescent ring should be avoided; better to try those with a pendant ring. Less surprising might be the fact that foul smelling mushrooms should be left alone; the mushrooms with no smell look like a safer choice. The fact that edible mushrooms have bruises looks like an interesting observation, perhaps indicating that edible mushrooms are more likely to show damage from foraging animals.

7 An Overall Process

An overall process for visualising computational intelligence through converting data into formal concepts is shown in Figure 19. It depicts the practical operation of the three open-source tools *FcaBedrock* (context creation), *In-Close* (context simplification) and *ConExp* (lattice visualisation). Start-to-finish analysis of a data set can be carried out in real time on a standard PC. Much of the process is automated, although if no Bedrock file exists for the data set being analysed, metadata not de-

tectable by FcaBedrock from the data file must be entered manually (there is currently no ‘magic’ process by which this information can be automatically extracted from a free-text data description document). However, more metadata is available through auto-detection if 3-column, *object-attribute-attribute value*, data files are being analysed.

Managing concept complexity and the level at which there are few enough concepts to make a visualisation readable, is currently a non-scientific process; In-Close can quickly determine the number of concepts in a cxt file created by FcaBedrock but, if there are too many, an iterative trial-and-error process is required to reduce them to a practical number.

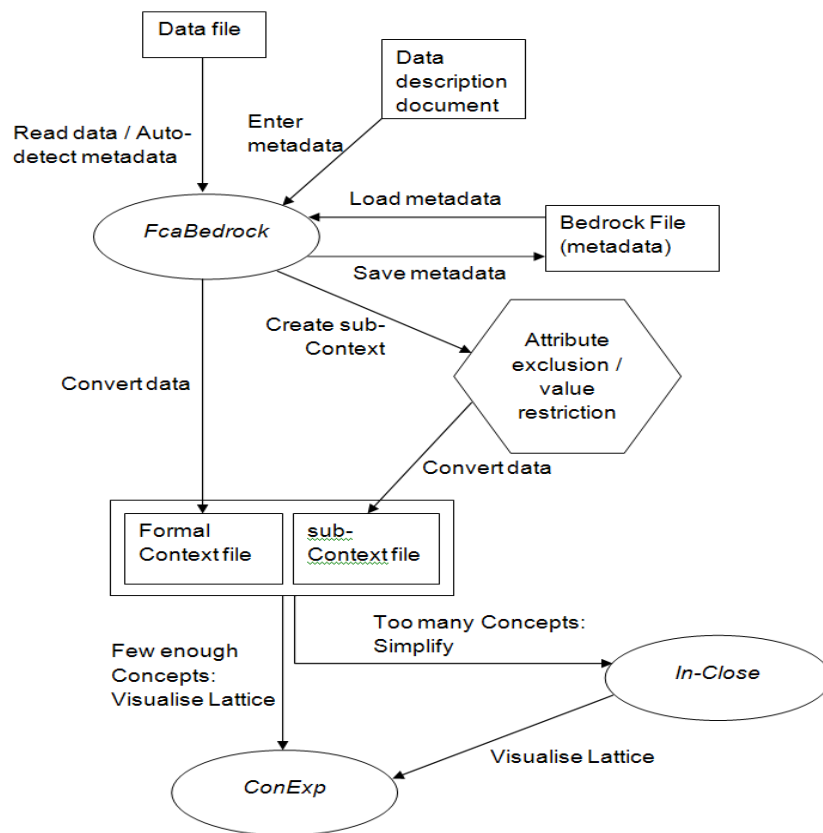


Fig. 19 Overall process incorporating the three open-source FCA tools

7.1 Performance Considerations

In-Close and similar high-performance programs have been shown to compute tens of thousands of formal concepts in seconds [1, 18]. Work on optimising these programs has brought these times down to fractions of a second, enabling real-time analysis of larger and larger data sets, as evidenced in this paper. On a standard desktop PC, the computation of the 220,000 concepts in the complete Mushroom context took 0.5 seconds, for example, and the computation of the 80,000 concepts in the complete Adult context took 0.17 seconds. Computing concepts when a minimum support is specified is even faster. Nevertheless, applying this analysis to even larger data sets will require further improvements in performance, the primary directions of this work being in multiprocessing and parallel programming.

Moving towards creating formal contexts from data in triples form (be it 3-column CSV or RDF triples) will require enhancements to FcaBedrock. Currently, contexts from large traditional flat-file CSV data files can be created in less than a second. However, with the increased parsing required with data in triples form, this time increases to several seconds. The Adult data set, for example, requires over 500,000 triples to represent the 16 many-valued attributes of the 32,561 objects in the data, and FcaBedrock takes about 7 seconds to read the file. However, a current thrust of data warehousing using triple-stores is towards efficiency of querying that stems from the integer-indexing of triples [14]. This can be exploited by data conversion tools such as FcaBedrock.

Work in these areas, to increase the performance of FCA tools and thus increase the scope for visualising hitherto hidden information from larger and larger data sources, is going to be an important part of a new European Commission funded project called *CUBIST*.

8 CUBIST

The approach described in this paper will form a core part of CUBIST (“Combining and Uniting Business Intelligence with Semantic Technologies”). CUBIST is a research project awarded under the European Union’s 7th Framework Programme, 5th ICT call, topic 4.3: Intelligent Information Management; STREP Project No.: FP7 257403. CUBIST aims to develop an approach for Business Intelligence that augments Semantic Technologies with BI capabilities and provides conceptually relevant and user-friendly FCA-based visual analytics. CUBIST will find applications within the Semantic Web through its use of RDF. CUBIST aims to deliver high performance in-warehouse interactive visual analytics for information warehouses and triple stores.

8.1 Semantic Web, RDF and OWL

For the Semantic Web, FcaBedrock is being developed to accept RDF files as input [9, 20, 24]. RDF uses the same logic for both input types currently supported by FcaBedrock; a normal form, where attributes are nested within their corresponding object, which is the same logic used for the flat-file CSV format and the *subject-predicate-object* logic, which is the same logic used for the 3-column CSV format. File 3 is an RDF/XML miniature version of the *Adult* data set [6], using only one object and five attributes: *age*, *education*, *employment*, *sex*, *us_citizen* and *class*. The file structure is straightforward and easy to follow; the example is referring to a 39 year old man who holds a Bachelors degree, works as a clerk, is a US citizen and earns less than 50K per year. A model of this file is depicted at Figure 20.

Functionality is also to be added for deriving data encoded in RDF vocabularies such as *Friend of a Friend (FOAF)* [11] and the Web Ontology Language *OWL* [15]. By using FcaBedrock as a semantic data preparation tool, FCA can find further applications in the Semantic Web and make knowledge representation, information management and visualizing conceptual structures among semantic data possible (Figure 21).

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:ad="http://mini-adult.net/adult#">
  <rdf:Description
    rdf:about="http://mini-adult.net/adult/person1">
    <ad:age>39</ad:age>
    <ad:education>Bachelors</ad:education>
    <ad:employment>Clerical</ad:employment>
    <ad:sex>Male</ad:sex>
    <ad:us_citizen>Yes</ad:us_citizen>
    <ad:class>&lt;=50K</ad:class>
  </rdf:Description>
</rdf:RDF>
```

File 3 mini-adult.xml, *RDF/XML* file.

Number	Subject	Predicate	Object
1	http://mini-adult.net/adult/person1	http://mini-adult.net/adult#age	"39"
2	http://mini-adult.net/adult/person1	http://mini-adult.net/adult#education	"Bachelors"
3	http://mini-adult.net/adult/person1	http://mini-adult.net/adult#employment	"Clerical"
4	http://mini-adult.net/adult/person1	http://mini-adult.net/adult#sex	"Male"
5	http://mini-adult.net/adult/person1	http://mini-adult.net/adult#us_citizen	"Yes"
6	http://mini-adult.net/adult/person1	http://mini-adult.net/adult#class	"<=50K"

Fig. 20 Model of the mini-adult.xml RDF file (see above).

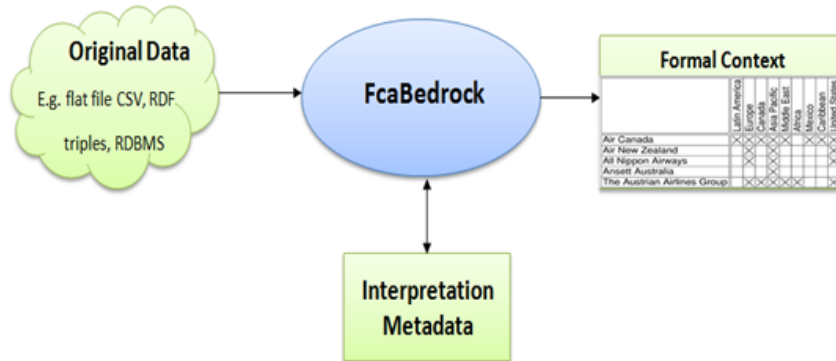


Fig. 21 Proposed Top-Level System Architecture of FcaBedrock.

9 Conclusion

Formal Concept Analysis (FCA) is an emerging data technology that complements collective intelligence such as that identified in the Semantic Web. Using the FCA open source software tools *FcaBedrock* and *In-Close* that were developed by the authors, and the open source lattice visualisation tool *ConExp*, disparate and distributed data can be visualised to discover its hitherto hidden meanings.

This paper has also demonstrated how collective computational intelligence is facilitated by interoperation of data analysis tools; *FcaBedrock* produces files of a standard FCA format for *ConExp* and *In-Close*; *In-Close* also produces such files (simplified contexts) for *ConExp*.

This paper has shown how FCA's visualisation can be applied to large-scale data sets, triples, and the Semantic Web's RDF and OWL. Key to this visualisation of data is that it is an inherent part of an intuitive and responsive interface, as initially demonstrated in this paper. The CUBIST project will develop this enhancement, and provide further use cases demonstrating the integration of FCA with the Semantic Web. CUBIST is bringing together European data warehousing companies, universities with expertise in FCA and commercial use-case partners to develop powerful, insightful and intuitive RDF-based FCA Visual Analytics for BI. The disparate data will come from a range of structured and unstructured sources, providing rich and complex challenges for CUBIST to provide collective intelligence through the use of FCA as an emerging data technology. In the interim this paper evidences that, by visualising this collective computational intelligence through converting data into formal concepts, the underlying knowledge that data depicts begins to be unlocked from the innumerable and increasing mass of it that is being recorded but not fully understood.

References

1. Andrews, S. (2009). *In-Close, A Fast Algorithm for Computing Formal Concepts*. Available: <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-483/paper1.pdf>. Last accessed 06 May 2010.
2. Andrews, S. (2009). *Data conversion and interoperability for FCA*. In: *Conceptual Structures Tools Interoperability Workshop*, ICCS 2009: Moscow.
3. Andrews, S. (2010). *In-Close*. Available at: <http://sourceforge.net/projects/inclose>
4. Andrews, S. and Orphanides, C. (2010). *FcaBedrock, a Formal Context Creator*. In: Croitoru, M., Ferre, S. and Lukose, D. (eds.). *Conceptual Structures: From Information to Intelligence*, 18th International Conference on Conceptual Structures (ICCS) 2010. LNAI 6208. Berlin: Springer. pp. 181-184
5. Andrews, S. and Orphanides, C. (2010). *FcaBedrock, a Formal Context Creator*. Available at: <http://sourceforge.net/projects/fcabedrock>
6. Asuncion, A. and Newman, D.J. (2007). *UCI Machine Learning Repository* [<http://www.ics.uci.edu/~mllearn/MLRepository.html>]. Irvine, CA: University of California, School of Information and Computer Science.
7. Becker, P. and Correia, J.H. (2005). *The ToscanaJ Suite for Implementing Conceptual Information Systems*. In *Formal Concept Analysis*, LNCS, Vol. 3626, pp. 324-348, Springer Berlin / Heidelberg.
8. Becker, P. and Correia, J.H. (2005). *ToscanaJ*. Available at: <http://sourceforge.net/projects/toscanaj>
9. Berners-Lee, T. (1998). *Why RDF model is different from the XML model*. Available: <http://www.w3.org/DesignIssues/RDF-XML>. Last accessed 14 July 2010.
10. *Frequent Itemset Mining Implementations Repository*. Available at: <http://fimi.cs.helsinki.fi>
11. *The Friend of a Friend (FOAF) project*. Available at: <http://www.foaf-project.org/>
12. Ganter, B., Wille, R. (1998) *Formal Concept Analysis: Mathematical Foundations*, Springer-Verlag, Berlin. Translated by C. Franzke.
13. Goethals, B. and Zaki, M. (2004). *Advances in Frequent Itemset Mining Implementations: Report on FIMI'03*. In: *SIGKDD Explorations Newsletter*. 6 (1), pp. 109-117.
14. Harris, S. and Gibbins, N. (2003) *3store: Efficient bulk RDF storage*. In: Proc. of PSSS03, pages 115.
15. Horrocks, I., Patel-Schneider, P. F. and Van Harmelen, F. (2003). *From SHIQ and RDF to OWL: the making of a Web Ontology Language*, *Web Semantics: Science, Services and Agents on the World Wide Web*, Volume 1, Issue 1, December, pp. 7-26, DOI: [dx.doi.org/10.1016/j.websem.2003.07.001](https://doi.org/10.1016/j.websem.2003.07.001)
16. Imberman, S. and Domanski, B. (1999). *Finding Association Rules from Quantitative Data using Data Booleanization*. Available: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.14.4447&rep=rep1&type=pdf>. Last accessed 11 May 2010.
17. Jin, R., Breitbart, Y. and Muoh, C. (2009). *Data discretization unification*. In: *Knowledge and Information Systems*. 19(1), pp. 1-29.
18. Krajca, P., Outrata, J., Vychodil, V. (2008) *Parallel Recursive Algorithm for FCA*. In: Belohlavek, R., Kuznetsov, S.O. (eds.), *Proceeding of the Sixth International Conference on Concept Lattices and their Applications*, pp. 71-82, Palacky University, Olomouc.
19. Kaytoue-Uberall, M., Duplessis, S. and Napoli, A. (2008). *Using Formal concept Analysis for the Extraction of Groups of Co-expressed Genes*. In: Le Thi, H.A., Bouvry, P., Pham Dinh, T. (eds.) *MCO 2008*. CCIS vol. 14, pp. 439-449. Springer-Verlag, Berlin/Heidelberg.
20. Passin, T. B. (2004). *Explorer's Guide to the Semantic Web*. Manning, Greenwich: CT.
21. Priss, U. (2008). *Formal Concept Analysis in Information Science*. In: Cronin, B. (ed.). *Annual Review of Information Science and Technology*, ASIST, vol. 40.

22. Priss, U. (2008) *FcaStone - FCA File Format and Interoperability Software*. In: Croitoru, M., Jaschke, R., Rudolph, S. (eds.), *Conceptual Structures and the Web, Proceedings of the Third Conceptual Structures and Tool Interoperability Workshop*, pp. 33-43.
23. Priss, U. (2008) *FCA Software Interoperability*, In: Belohlavek, R., Kuznetsov, S. O. (eds.), *Proceeding of the Sixth International Conference on Concept Lattices and Their Applications*, pp. 133-144.
24. Semantic Web. (2010). *The Semantic Web*. Available: http://semanticweb.org/wiki/Main_Page. Last accessed 28 Apr 2010.
25. Slezak, D., Wroblewski, J., Eastwood, V. and Synak, P. (2008). *Brighthouse: an analytic data warehouse for ad-hoc queries*. In: *Proceedings of the VLDB Endowment*. vol. 1(2), pp. 1337-1345. ACM Digital Library.
26. *SPARQL Query Language for RDF*. Available at: <http://www.w3.org/TR/rdf-sparql-query/>
27. Stumme, G., Taouil, R., Bastide, Y. and Lakhal, L. (2001). *Conceptual Clustering with Iceberg Concept Lattices*. In: *Proceedings of GI-Fachgruppentreffen Maschinelles Lernen'01, Universitat Dortmund*.
28. World Wide Web Consortium. (2010). *Design Issues*. Available: <http://www.w3.org/DesignIssues/>. Last accessed 08 May 2010.
29. White, P.W. and French, C.D. (1998). *Database system with methodology for storing a database table by vertically partitioning all columns of the table*. US Patent 5,794,229, August 11, 1998.
30. Wille, R. (2005). *Formal Concept Analysis as Mathematical Theory of concepts*. In: Ganter, B., Stumme, G. and Wille, R. (eds.) *Formal Concept Analysis: Foundations and Applications*. Berlin: Springer. pp. 1-6.
31. Wolff, K.E. (1993). *A First Course in Formal Concept Analysis*. Available: http://www.fbmh.de/home/wolff/Publikationen/A_First_Course_in_Formal_Concept_Analysis.pdf. Last accessed 03 May 2010.
32. Yevtushenko, S. (2006). *ConExp*. Available at: <http://sourceforge.net/projects/conexp>
33. Zaki, M.J., Hsiao, C-J. (2005) *Efficient Algorithms for Mining Closed Itemsets and Their Lattice Structure*. In: *IEEE Transactions on Knowledge and Data Mining*, Vol. 17, No. 4, IEEE Computer Society.

Exploring the Applicability of Formal Concept Analysis on Market Intelligence Data

Constantinos Orphanides

Conceptual Structures Research Group
Communication and Computing Research Centre
Faculty of Arts, Computing, Engineering and Sciences
Sheffield Hallam University, Sheffield, UK
`c.orphanides@shu.ac.uk`

Abstract. This paper examines and identifies issues associated with the applicability of FCA on sample data provided by a CUBIST use-case partner. The paper explains the various steps related to the transformation of these data to formal contexts, such as preprocessing, cleansing and simplification, as well as preprocessing and limitation issues, by using two FCA tools currently being developed in CUBIST, FcaBedrock and InClose. The paper demonstrates what is achievable to date, using the above-mentioned tools and what issues need to be considered to achieve more meaningful and intuitive FCA analyses. The paper concludes by suggesting and explaining techniques and features that should be implemented in later iterations of these tools, to deal with the identified barriers. This work has been carried out as a part of the European CUBIST FP7 Project: <http://www.cubist-project.eu>

1 Introduction

It has been shown that a variety of datasets can be converted into formal contexts [8,2] by a process of discretising and Booleanising the data [10]. However, depending on the nature of the dataset, manual or automated means of preprocessing have to be deployed first, in order for FCA to be successfully carried out. Although the open-source and freely available FCA tools currently being developed in CUBIST, FcaBedrock [3,6] and InClose [1,9], are configured to cater for most preprocessing and data cleansing issues [3,4], further issues might arise: types of attribute that have not been catered for or considered so far, such as free-text data and data inconsistencies.

This paper attempts to identify such issues, by conducting FCA on a dataset provided by Innovantage, a CUBIST use-case partner, providing market and competitive intelligence in the United Kingdom. The paper concludes on further work and explains what techniques will be deployed, in later iterations of the tools, to cater for the issues identified while analysing the specific dataset.

2 Dataset Description

The specific dataset consists of job vacancies advertised on the United Kingdom's leading job boards, as well as employers' own websites, tracked in real-time using Innovantage's proprietary software. The dataset is in XML format and has been extracted from a MySQL RDBMS. The dataset comprises of 900 jobs accompanied by their details:

- Title: The job's title.
- Description: A brief description outlining the requirements of the job.
- Date Found: The exact time of when the job was tracked.
- URL: The website where the job was found at.
- Raw Location: The location of the employer.
- Raw Salary: The advertised salary, sometimes also including information about bonuses and benefits.

An example of a job entry is shown below (File 1).

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<jobs>
  <job>
    <title>Data Centre Developer</title>
    <description>Data centers Developer opportunity based in Amsterdam
on a 6 months rolling contract. This is a very senior position and
requires the candidate to have at least 7 years experience and
have extensive knowledge and expertise in the construction
of a data centers facilities including electrical systems,
cooling plants etc and familiar with EU regulations and
best practices.</description>
    <date_found>2011-01-12 17:01:58.0</date_found>
    <url>http://www.itjobspost.com/JobSeeker/</url>
    <raw_location>Amsterdam, Other Countries, UK</raw_location>
    <raw_salary>400-600 Per Day</raw_salary>
  </job>
</jobs>
```

File 1 innovantage_sample.xml, XML file.

3 Data Conversion Process

3.1 Preprocessing

Some issues surfaced during preprocessing, mainly due to the XML file failing to render properly because of illegal, non-UTF-8 characters contained in the data, possibly related to the automated process in which jobs are tracked and recorded. This was worked-around by parsing the XML file using a custom-made algorithm and removing all illegal characters and symbols, without loss of information and without affecting the quality of the data. The XML file was

title	description	date_found	url	raw_location	raw_salary
Software Engineer	Software EngineerWe have a	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Birmingham, Coventry, Tan	35,000
Linux Systems Administrator - Ubuntu, Apache, MYSQL	Linux Systems Administrator	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	London	45k - 50k pa + Bonus & benefits
Assistant Buyer	Assistant Buyer reporting to I	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Basingstoke	18k - 23k pa + Pension
Credit Controller	The individuals will be part o	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Liverpool	16000 - 18000
Lead Firmware Engineer - Embedded, C/ C++, Linux, J2EE	Our client is recognised as th	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Poole	45k pa
Bid Manager - Contract Cleaning - Soft FM	Bid ManagerContract Cleanin	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	London	4k - 45k pa + benefits
Implementation Consultant for leading Buy Side Vendor	Key words -Implementation (2011-01-12 17:00:46.0	http://www.jobsite.co.uk	London, Uk, Europe	70,000 - 90,000
Business Analyst - Solvency 2	Due to an expanding Program	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	London	Neg
Area / Branch Manager	Area / Branch ManagerDomic	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Tyne & Wear, Newcastle, S	28k - 30k pa
SAP HCM Technical Architect	We are looking for a SAP HR	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	England	650 - 750 p/day + Expenses
Principal Developer C#.NET (Application Architect)	Principal Developer C#.NET (2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Basingstoke, Reading, Alde	45k - 70k pa
Business Development Manager (Cloud / AMS)	Business Development Mana	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	London	Salary plus benefits
Human Resource Administrator	We are currently recruiting f	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Manchester	Negotiable
Gas Engineer	Benefits- Pension Scheme- C	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	North Allerton And North Y	Negotiable
Senior Estimator/ Bid Manager	Bid Manager/Senior Estimato	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	North West England, Lanca	Negotiable + package
Site Manager	Benefits- Bonus discretionar	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Cambridgeshire	30,000 to 40,000
Experienced Admin Assistant	YOU WILL NOT BE CONSIDERE	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Shrewsbury	Negotiable
Electrical - Clerk of Works	Our client is looking for a Elei	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Leicestershire	18 to 22
Senior Quantity Surveyor	This major civil engineering c	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Cambridgeshire	28,000 to 35,000
General Manager - Water Bottling Plant	Benefits- tax free salary- acco	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Afghanistan	Negotiable
Area Sales Manager - Building Products - Kent	JOB TITLE: Area Sales Manage	2011-01-12 17:00:46.0	http://www.jobsite.co.uk	Kent, Essex, London	30k basic, 35k ote

Fig. 1. XML-to-CSV transformation of the dataset.

then loaded in MS Excel and converted to CSV (Figure 1), as FcaBedrock does not currently support XML as input.

Another issue that affected the analysis was the fact that free-text attributes, as their name implies, are inconsistent, mostly due to the fact that the recorded data originate from various sources. Taking the ‘Raw Location’ attribute as an example, a job’s location can be recorded as “Manchester” for one job, “Manchester, United Kingdom” for another job and “Greater Manchester” for another job. The same problem applied for the ‘Raw Salary’ attribute, as some employers use ranges (e.g. “15000-20000”), some use finite values and also include currencies (e.g. “18000 GBP”) and others also include additional information (e.g. “25000 per annum, negotiable”). For this type of free-text attribute to be successfully and meaningfully converted, some kind of Semantic Extract Transform Load (SETL) or Natural Language Processing (NLP) process would be required first, in order to identify values. As such, the data had to be manually modified; 100 jobs were randomly selected from the lot and had the above-mentioned attributes re-configured for consistency. In addition, some attributes were excluded from the analysis; in particular, ‘Description’ was excluded as it is not an attribute, but rather the title (or descriptive annotation) of the object, although it could be useful as part of the case-study in terms of adding meaning to an analysis. The

‘Date Found’ attribute was also excluded, as all of the jobs in the dataset were tracked on the same date, thus adding no specific value to the analysis. The ‘URL’ attribute was excluded, as URLs are unique for each job posted (thus considered free-text data). In terms of reconfiguring attributes, the ‘Raw Location’ attribute was configured to hold only city names and the ‘Raw Salary’ attribute was configured to be purely numeric. An extra attribute was created to hold additional information originally contained in the ‘Raw Salary’ attribute, such as whether the salary is negotiable or not. This resulted in four attributes remaining: ‘Title’, ‘Raw Location’, ‘Raw Salary’ and ‘Negotiable Salary’ (the new attribute that resulted during preprocessing). A screenshot showing how the dataset looks after preprocessing is shown at Figure 2 below.

title	raw_location	raw_salary	sal_negotiable
Assistant Buyer	Aberdeenshire	18000	n
Credit Controller	Afghanistan	16000	n
Lead Firmware Engineer -	Antrim	45000	n
Bid Manager - Contract Cle	Barrow-in-Furness	45000	n
Business Analyst - Solvenc	Basingstoke	Neg	y
Business Development Ma	Bedfordshire	Salary plus benefits	y
Human Resource Administ	Birmingham	Negotiable	y
Site Manager	Birmingham	30000	y
Experienced Admin Assist	Brighton	Negotiable	y
Electrical - Clerk of Works	Brighton	22000	y
Senior Quantity Surveyor	Bristol	28000	y
General Manager - Water l	Bristol	Negotiable	y
Administrator	Bromborough	14874	n
Customer Service Advisor	Cambridge	14000	n
Solidworks Design Engineer	Cambridge	NEG.	n
Assistant Accountant	Cambridgeshire	22000	n
Product/ Configuration En	Cambridgeshire	26000	n
Web Tester - Berkshire - 1	Cheltenham	39600	n
Project Manager - URGENT	Colchester	45000	n

Fig. 2. Final version of the dataset, after preprocessing.

3.2 Transforming the Dataset into a Formal Context

The dataset was loaded in FcaBedrock and the ‘Title’ attribute was excluded from the analysis, using the attribute exclusion feature. The metadata auto-detection feature of FcaBedrock was used to avoid entering metadata manually (Figure 3).

Converting the dataset with FcaBedrock resulted in a formal context with 67 formal attributes. Feeding the formal context in InClose resulted in 110 formal concepts; although not quite a large amount, the concepts had to be reduced to an amount where the concept lattice would be readable and manageable. Over a trial-and-error process, using InClose and the well-known idea

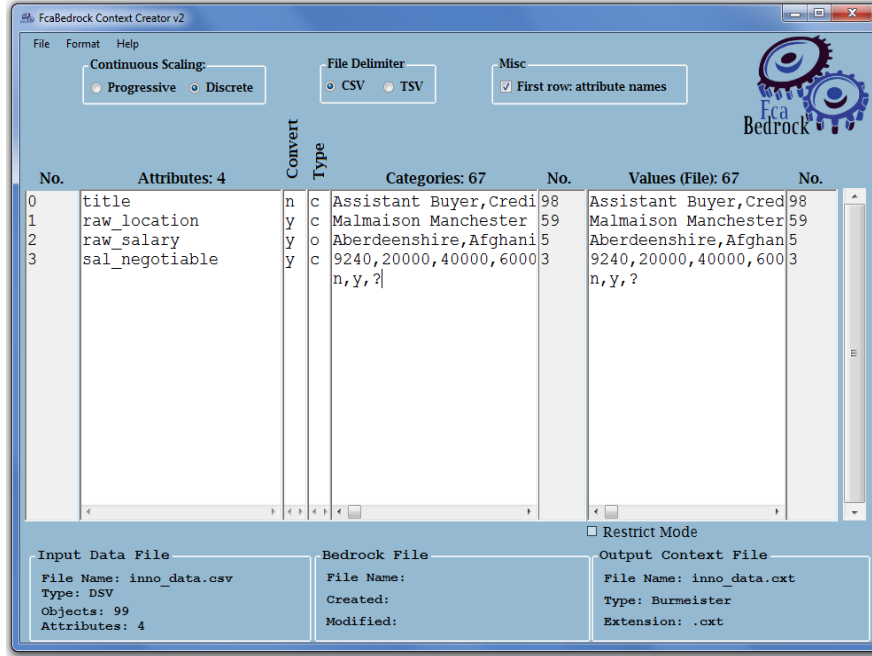


Fig. 3. Autodetecting the metadata in FcaBedrock.

of minimum-support (a semi-automated form of lattice ‘iceberging’ [12]), the minimum-support for the intent was set to 2 and the minimum-support for the extent was set to 5. This resulted in 9 concepts. When visualised in ConExp [13,5], however, 20 concepts are displayed. This is because where the large concepts ‘overlap’, other concepts are found during a *second pass* of concept mining, with no minimum support, when producing the concept lattice [4]. In this way, possibly significant concepts, that would not have satisfied the initial minimum-support are retained and a complete hierarchy is maintained in the resulting concept lattice (Figure 4).

4 Analysis

Even with a small amount of objects and attributes, interesting information can be extracted from the lattice. For example, all non-negotiable salaries are the ones that fall in the £40000-60000 range, while the negotiable salaries fall in the £20000-40000 range. For salaries where it is undefined (or unknown) if they are negotiable, there seems to be no distinct indication as to why this is the case. As for salaries in the £9240-20000 range, they all fall under the ‘sal_negotiable-n’ and ‘sal_negotiable-?’ attributes, with a 50-50 ratio. The overall conclusion

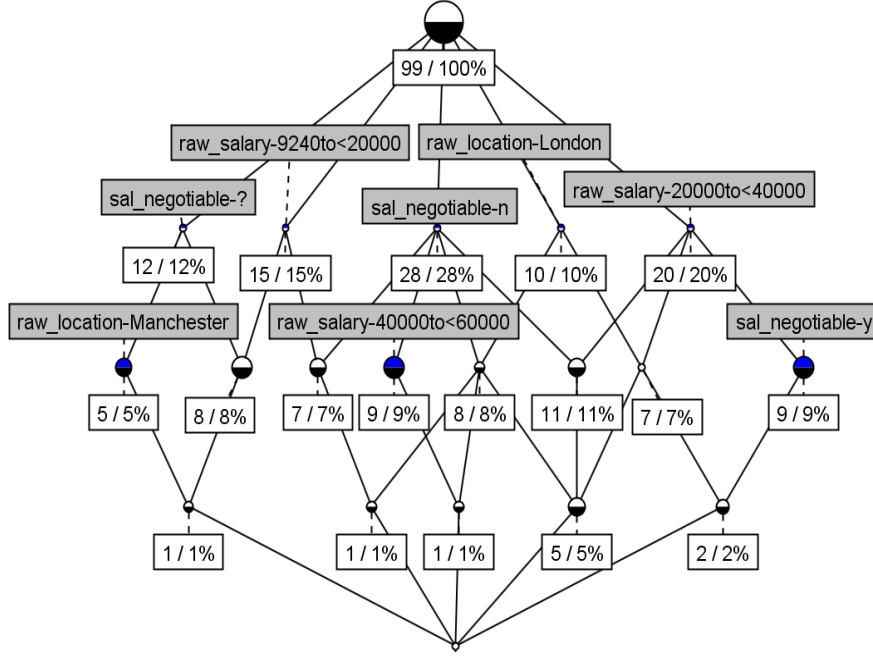


Fig. 4. Visualising the resulting formal context in ConExp.

indicates that for high-end salaries negotiation is not an option, while negotiation is possible for mid-end salaries. Interesting is the fact that low-end salary jobs tend not to specify whether their salaries are negotiable or not, though when they do they are not negotiable. Why is that the case? Questions of these nature require further investigation.

While the insights provided by the resulting lattice might not be groundbreaking, a close collaboration of the FCA analyst with the domain expert would help in refining the requirements, to produce meaningful business questions that would be more suitable for analysis of such data. For example, the domain expert might want to investigate why employers tend to not specify or negotiate jobs with low-end salaries. Could it have something to do with their geographic location or the domain of the job? Such kind of analysis is perfectly feasible in FCA, by restricting the context to specific attributes (and attribute values) of interest. Figure 5 shows how this can be done in FcaBedrock, where the location was restricted to London, Manchester and Birmingham, the raw salary was restricted to low-end only and the negotiable salary attribute was set to unknown. As such, the business question has been redefined to “display jobs in London, Manchester or Birmingham with low-end salaries, where salary negotiation is unknown or unspecified”.

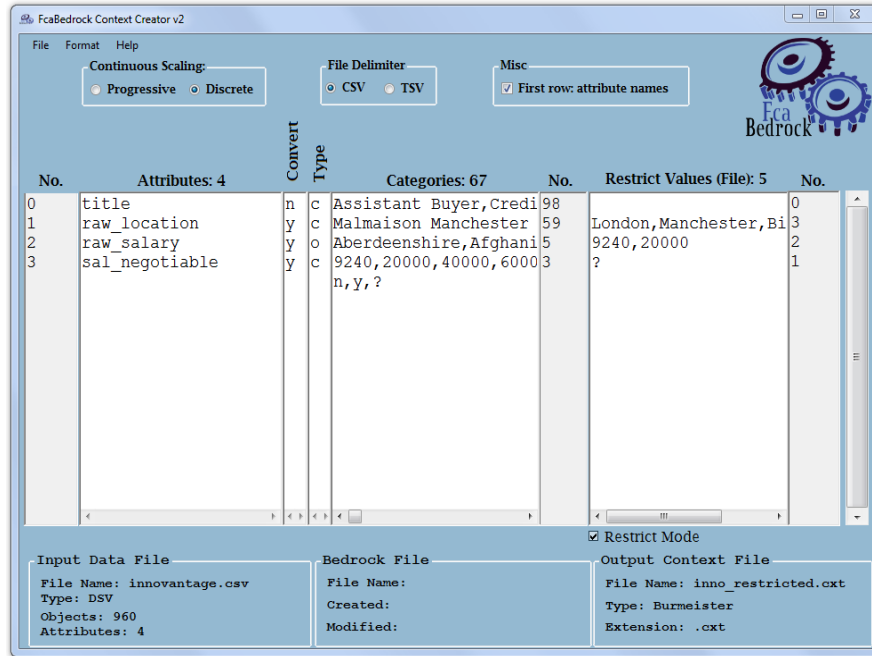


Fig. 5. Using FcaBedrock’s restriction capabilities to focus the analysis on specific attributes and attribute values.

5 Further Work

It is evident that as new data sources and data types are introduced, more preprocessing issues arise. With regards to the tools used in this analysis (FcaBedrock and InClose), further development is currently in process and various issues, mentioned below, are already being considered.

In terms of data sources, XML should be added as a default data source, to avoid XML-to-CSV transformations. Pulling data directly from an RDBMS source would be quite useful as well, by selecting specific database tables, or even specific columns from each table, to use in the analysis. Manipulation of RDF data are of high importance as well, given the fact that CUBIST revolves around semantic technologies.

Free-text data have proven to be not suitable for FCA, unless some kind of Semantic ETL or NLP process, in order to identify values, is deployed first. Use of thesauri, such as the approach described in [11], to tokenize free-text data into categories could prove useful as well, although whether these kind of processes will be manual, semi-automated or automated remain research questions which require further study.

Another feature that would be particularly useful would be to embed additional functionality in the autodetection features of FcaBedrock, particularly for selecting appropriate scales and intervals for continuous attributes. Understanding the true nature of a continuous attribute at the moment, using FcaBedrock, is only feasible when datasets include documentation, such as the ones in the UCI Machine Learning Repository [7], or by manually investigating the data. As such, suggesting ranges and scales, using the same ‘guided automation’ approach that FcaBedrock uses [3] would make analyzing such attributes more meaningful and insightful.

6 Conclusion

The paper has explored the application of FCA within a market intelligence scenario, using real-life data from a CUBIST use-case partner, deploying freely-available and open-source FCA tools, currently being developed in CUBIST, for the analysis. Several preprocessing issues have been identified and suggestions, techniques and features have been proposed for further work.

Although the work presented in this paper is still at an early stage, it demonstrates how the market data and FCA communities can benefit from each other. The market data community has provided new challenges that FCA has to consider, mostly in terms of usability and user-friendliness. Within the context of CUBIST, we envisage that the market data analysts will be able to conduct FCA analysis on their data, without collaborating with FCA experts.

Acknowledgement This work is part of the CUBIST project (“Combining and Uniting Business Intelligence with Semantic Technologies”), funded by the European Commission’s 7th Framework Programme of ICT, under topic 4.3: Intelligent Information Management. More information on the project can be found at <http://www.cubist-project.eu>

References

1. Andrews, S.: *In-Close, A Fast Algorithm for Computing Formal Concepts*. In: Rudolph, S., Dau, F., Kuznetsov, S.O. (eds.) ICCS'09, <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-483/> (2009)
2. Andrews, S.: *Data Conversion and Interoperability for FCA*. In: CS-TIW 2009, pp. 42-49, http://www.kde.cs.uni-kassel.de/ws/cs-tiw2009/proceedings_final_15July.pdf (2009)
3. Andrews, S. and Orphanides, C.: *FcaBedrock, a Formal Context Creator*. In: Croitoru, M., Ferre, S. and Lukose, D. (eds.) ICCS 2010, LNAI 6208. Springer-Verlag, Berlin/Heidelberg (2010)
4. Andrews, S. and Orphanides, C.: *Analysis of Large Data Sets using Formal Concept Lattices*. In: Kryszkiewicz, M. and Obiedkov, S. (eds.). Proceedings of the 7th International Conference on Concept Lattices and Their Applications (CLA) 2010, ISBN 978-84614-4027-6. Seville: University of Seville. pp. 104-115 (2010)
5. ConExp (Concept Explorer). Available at <http://sourceforge.net/projects/conexp>
6. FcaBedrock Formal Context Creator. Available at <http://sourceforge.net/projects/fcabedrock>
7. Frank, A. and Asuncion, A.: *UCI Machine Learning Repository* [<http://archive.ics.uci.edu/ml>]. Irvine, CA: University of California, School of Information and Computer Science (2010)
8. Ganter, B. and Wille, R.: *Conceptual Scaling*. In: Roberts, F. (ed.) Applications of Combinatorics and Graph Theory to the Biological and Social Sciences. IMA, vol. 17, pp. 139-168, Springer, Berlin-Heidelberg-New York (1989)
9. InClose Formal Concept Miner. Available at <http://sourceforge.net/projects/inclose>
10. Kaytoue-Uberall, M., Duplessis, S. and Napoli, A.: *Using Formal Concept Analysis for the Extraction of Groups of Co-expressed Genes*. In: Le Thi, H.A., Bouvry, P., Pham Dinh, T. (eds.) MCO 2008. CCIS vol. 14, pp. 439-449. Springer-Verlag, Berlin/Heidelberg (2008)
11. Poelmans, J., Elzinga, P., Viaene, S. and Dedene, G.: *Formal Concept Analysis in Knowledge Discovery*. In: Croitoru, M., Ferre, S. and Lukose, D. (eds.) ICCS 2010, LNAI 6208. Springer-Verlag, Berlin/Heidelberg (2010)
12. Stumme, G., Taouil, R., Bastide, Y. and Lakhal, L.: *Conceptual Clustering with Iceberg Concept Lattices*. In: Proceedings of GI-Fachgruppentreffen Maschinelles Lernen'01, Universitat Dortmund, vol. 763. (2001)
13. Yevtushenko, S.A.: *System of data analysis "Concept Explorer"*. (In Russian). Proceedings of the 7th national conference on Artificial Intelligence KII-2000, p. 127-134, Russia, 2000.

Knowledge discovery through creating formal contexts

Simon Andrews* and Constantinos Orphanides

Conceptual Structures Research Group,
Communication and Computing Research Centre,
Sheffield Hallam University,
Cantor Building, City Campus, 153 Arundel Street,
Sheffield, S1 2NU, UK
E-mail: s.andrews@shu.ac.uk
E-mail: c.orphanides@shu.ac.uk

*Corresponding author

Abstract: Knowledge discovery is important for systems that have computational intelligence in helping them learn and adapt to changing environments. By representing, in a formal way, the context in which an intelligent system operates, it is possible to discover knowledge through an emerging data technology called formal concept analysis (FCA). This paper describes a tool called *FcaBedrock* that converts data into formal contexts for FCA. This paper describes how, through a process of guided automation, data preparation techniques such as attribute exclusion and value restriction allow data to be interpreted to meet the requirements of the analysis. Examples are given of how formal contexts can be created using *FcaBedrock* and then analysed for knowledge discovery, using real datasets. Creating formal contexts using *FcaBedrock* is shown to be straightforward and versatile. Large datasets are easily converted into a standard FCA format.

Keywords: formal concept analysis; FCA; formal context; formal concepts; concept lattice; guided automation; visualisation; knowledge discovery; business intelligence; BI.

Reference to this paper should be made as follows: Andrews, S. and Orphanides, C. (2012) 'Knowledge discovery through creating formal contexts', *Int. J. Space-Based and Situated Computing*, Vol. 2, No. 2, pp.123–138.

Biographical notes: Simon Andrews is a Senior Lecturer in Software Engineering at Sheffield Hallam University, co-leader of the Conceptual Structures Research Group and an international expert on formal concept analysis (FCA). He is the Editor-in-Chief of a new journal called the *International Journal of Conceptual Structures and Smart Applications (IJCSSA)*. He is currently a Principle Investigator for a prestigious European Commission FP7 project, called CUBIST: <http://www.cubist-project.eu/>, where he is working on leading-edge semantic large-scale data analysis techniques that include knowledge discovery through FCA-based visual analytics.

Constantinos Orphanides is a Senior Research Assistant at Sheffield Hallam University and a member of the Conceptual Structures Research Group. He is an expert on scaling data for FCA and the author of software to automate this task. He is currently working as part of a prestigious European Commission FP7 project, called CUBIST, where he is developing scaling techniques and software to support FCA in a semantic technology setting, working with RDF and data triple stores.

1 Introduction

Formal concept analysis (FCA) was introduced in the 1990s by Rudolf Wille and Bernhard Ganter (Ganter and Wille, 1998), building on applied lattice and order theory developed by Birkhoff and others in the 1930s. It was initially developed as a subsection of applied mathematics based on the mathematisation of concepts and concepts hierarchy, where a concept is constituted by its *extension*, comprising of all objects which belong to the concept, and its *intension*, comprising of all attributes (properties, meanings) which apply to all objects of the extension

(Wille, 2005). The set of objects and attributes, together with their relation to each other, form a *formal context*, which can be represented by a cross-table (Table 1).

1.1 Formal contexts

The cross-table (Table 1) shows a formal context representing destinations for five airlines. The elements on the left side are formal objects; the elements at the top are formal attributes. If an object has a specific property (formal attribute), it is indicated by placing a cross in the corresponding cell of the table. An empty cell indicates that

the corresponding object does not have the corresponding attribute. In the airlines context, Air Canada flies to Latin America (since the corresponding cell contains a cross) but does not fly to Africa (since the corresponding cell is empty). However, an empty cell might also mean that it is unknown whether the corresponding object has the corresponding attribute (Wolff, 1993).

Table 1 Airlines context

Airlines	Latin America	Europe	Canada	Asia Pacific	Middle East	Africa	Mexico	Caribbean	USA
Air Canada	×	×	×	×	×		×	×	×
Air New Zealand		×		×					×
Nippon Airways		×		×					×
Ansett Australia				×					
Austrian Airlines		×	×	×	×	×			×

In mathematical terms, a formal context is defined as a triple $\mathbb{K} := (G, M, I)$, with G being a set of objects, M a set of attributes and I a relation defined between G and M . The relation I is understood to be a subset of the cross product between the sets it relates, so $I \subseteq G \times M$. If an object g has an attribute m , then $g \in G$ relates to m by I , so we write $(g, m) \in I$, or gIm . For a subset of objects $A \subseteq G$, a derivation operator $'$ is defined to obtain the set of attributes, common to the objects in A , as follows:

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

Similarly, for a subset of attributes $B \subseteq M$, the derivation operator $'$ is defined to obtain the set of objects, common to the attributes in B , as follows:

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

1.2 Formal concepts

Now, a pair (A, B) is a *formal concept* in a given formal context (G, M, I) only if $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. The set A is the extent of the concept and the set B is the intent of the concept. A formal concept is, therefore, a closed set of object/attribute relations, in that its extension contains all objects that have the attributes in its intension, and the intension contains all attributes shared by the objects in its extension. In the airlines example, it can be seen from the cross-table that Air Canada and Austrian Airlines fly to both USA and Europe. However, this does not constitute a formal concept because both airlines also fly to Asia Pacific, Canada and the Middle East. Adding these destinations completes (closes) the formal concept:

({Air Canada, Austrian Airlines}, {Europe, USA, Asia Pacific, Canada, Middle East}).

1.3 Computing formal concepts

For the computation of formal concepts, a Boolean matrix is frequently used to represent the formal context. In the Boolean matrix, the rows and columns are computationally interchangeable; the same will apply if 'A', 'extent', 'object' and 'row' are substituted for 'B', 'intent', 'attribute' and 'column' (and vice-versa).

There are several algorithms for computing the formal concepts in a formal context – reviews of such algorithms, that take a variety of approaches, can be found in Arevalo et al. (2007), Kuznetsov and Obiedkov (2002), and Zaki and Hsiao (2005).

Figure 1 The in-close algorithm

```

InClose( $r, y$ )
1  begin
2     $r_{new} \leftarrow r_{new} + 1$ ;
3    for  $j \leftarrow y$  upto  $n - 1$  do
4       $A[r_{new}] \leftarrow \emptyset$ ;
5      foreach  $i$  in  $A[r]$  do
6        if  $I[i][j]$  then
7           $A[r_{new}] \leftarrow A[r_{new}] \cup \{i\}$ ;
8      if  $|A[r_{new}]| > 0$  then
9        if  $|A[r_{new}]| = |A[r]|$  and  $|B[r]| = 0$  then
10          $B[r] \leftarrow B[r] \cup \{j\}$ ;
11        else
12          if IsCanonical( $r, j - 1$ ) then
13             $B[r_{new}] \leftarrow B[r] \cup \{j\}$ ;
14            InClose( $r_{new}, j + 1$ );
15  end

```

An example of such an algorithm is given in Figure 1, implemented in the formal concept miner In-Close (<http://sourceforge.net/projects/inclose/>), developed by one of the authors. In-Close is based conceptually on Kuznetsov's (1999) *close-by-one* algorithm. To better understand the algorithm, I is a Boolean matrix representing a formal context, with m rows, representing a set of objects, and n columns, representing a set of attributes. A formal concept is represented by an extent $A[r]$ (an ordered list of objects) and an intent $B[r]$ (an ordered list of attributes), where r is the index of the concept. In the algorithm, j is the current attribute, r is the index of the concept being closed, r_{new} is a global index of the candidate new concept and y is a starting column. InClose(r, y) means 'incrementally close concept r , beginning at attribute y '.

IsCanonical checks if a concept is new by examining its canonicity (lexicographical order) to circumvent the need to explicitly search for repeated results (Ganter, 1984). For a more thorough and technical explanation of how the algorithm works, refer to Andrews (2009b, 2011).

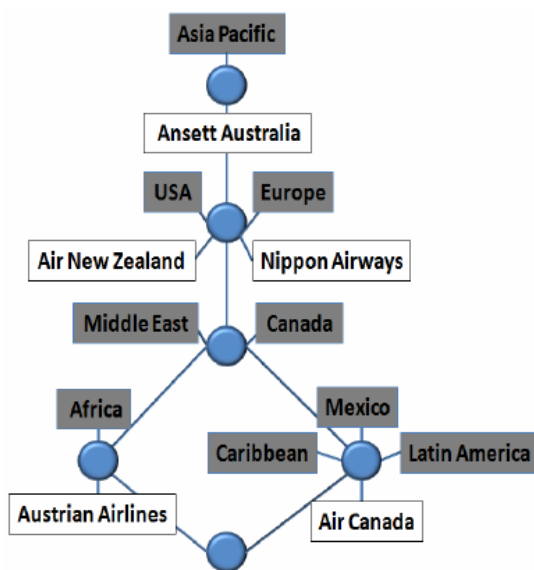
1.4 Galois connections

Another central notion of FCA is a duality called a ‘Galois connection’, which is often observed between items that relate to each other in a given domain, such as objects and attributes. A Galois connection implies that “if one makes the sets of one type larger, they correspond to smaller sets of the other type, and vice versa” (Priss, 2008). Using the formal concept above as an example, if Africa is added to the list of destinations, the set of airlines reduces to {Austrian Airlines}.

1.5 Concept lattices

The Galois connections between the formal concepts of a formal context can be visualised in a *concept lattice* (Figure 2), which is an intuitive way of discovering hitherto undiscovered information in data and portraying the natural hierarchy of concepts that exist in a formal context.

Figure 2 A lattice corresponding to the airlines context (see online version for colours)



A concept lattice consists of the set of concepts of a formal context and the subconcept-superconcept relation between the concepts. The nodes in Figure 2 represent formal concepts. It is conventional that formal objects are noted slightly below and formal attributes slightly above the nodes, which they label.

A concept lattice can provide valuable information when one knows how to read it. As an example, the node which is labelled with the formal attribute ‘Asia Pacific’ shall be referred to as *Concept A*. To retrieve the extension of Concept A (the objects which feature the attribute ‘Asia Pacific’), one begins at the node where the attribute is

labelled and traces all paths which lead down from the node. Any objects one meets along the way are the objects which have that particular attribute. Looking at the lattice in Figure 2, if one takes the attribute ‘Asia Pacific’ and traces all paths which lead down from the node, one will collect all the objects. Thus, Concept A can be interpreted as ‘All airlines fly to Asia Pacific’. Similarly, the node which is labelled with the formal object ‘Air New Zealand’ shall be referred to as *Concept B*. To retrieve the intension of Concept B (the attributes of ‘Air New Zealand’), one begins at the node where the object is labelled and traces all paths which lead up from the node. Any attributes one meets along the way, are the attributes of that particular object. Looking at the lattice once again, if one takes the object ‘Air New Zealand’ and traces all paths which lead up from the node, one will collect the attributes ‘USA’, ‘Europe’, and ‘Asia Pacific’. This can be interpreted as ‘The Air New Zealand airline flies to USA, Europe and Asia Pacific’. The concept that we formed previously by inspecting the cross-table,

{(Air Canada, Austrian Airlines)}, {Europe, USA
Asia Pacific, Canada, Middle East}},

is the node in the centre of the lattice; the one labelled with ‘Middle East’ and ‘Canada’. It becomes quite clear, for example, that although Air New Zealand and Nippon Airways also fly to Europe, USA and Asia Pacific, only Air Canada and Austrian Airlines fly to Canada and the Middle East as well.

Although the airline context is a small example of FCA, visualising the formal context clearly shows that concept lattices provide richer information than from looking at the cross-table alone. This type of hierarchical intelligence that is gleaned from FCA is not so readily available from other forms of data analysis.

1.6 Converting data for FCA

It has been shown that FCA can be usefully applied to large sets of data (Poelmans et al., 2011; Kaytoute-Uberall et al., 2008; Sawase et al., 2009; Ignatov and Kuznetsov, 2009) and that it has applications in data mining (Boulcaut and Besson, 2008; Rioult et al., 2003). Algorithms exist that are capable of processing large formal contexts in good time (Andrews, 2011; Krajca et al., 2008). However, data is rarely in a form immediately suitable for FCA tools and applications. For FCA to be carried out, these data must be converted into formal contexts. The existing, typically many-valued, attributes must be converted into formal attributes. This can be an involved and time consuming task, usually requiring a programming element. The task is, essentially, a process of Booleanising data; taking each many-valued attribute and converting it into as many Boolean attributes as it has values. The incorporation of many-valued attributes into FCA is well known (Wolff, 1993; Ganter and Wille, 1998) and continuous data can be used for FCA by being hierarchically scaled (Priss, 2008) or discretised as disjoint ranges of values (Kaytoute-Uberall

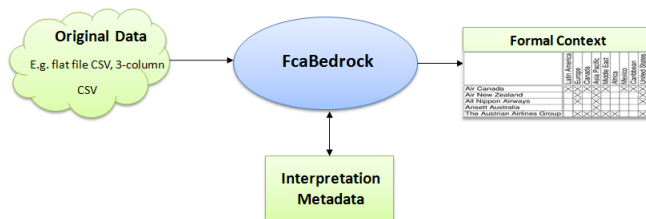
et al., 2008). However, it is not always obvious how or even if a particular attribute should be converted. If an attribute has hundreds of values, converting it into hundreds of formal attributes may be pointless if this leads to a hugely complex lattice – it is probably not suitable for FCA. For continuous values the question arises as to how to form the ranges to discretise them; how many and how large? Consequently, there is less work describing how these techniques can be applied in an automated, reproducible way.

A process of data conversion leads to the possibility of diverse interpretation of data. Different choices of which of the data to convert and how to convert them can, without careful documentation, lead to inconsistent and incomparable analyses and lead to problems in measuring the performance of FCA tools and algorithms (Kuznetsov and Obiedkov, 2002). It was thus proposed at CSTIW'09 that a tool should be developed to apply automation to the process of conversion and to address these issues (Andrews, 2009a).

2 FcaBedrock overview

FcaBedrock (<https://sourceforge.net/projects/fcabedrock/>) is an open-source tool for automating the conversion of existing, flat-file csv datasets into formal contexts. Creating formal contexts is a common process which, until now, was done manually. Several FCA tools, such as the ToscanaJ kit for browsing conceptual schemas and the lattice builder ConExp (<https://sourceforge.net/projects/conexp/>) (Yevtushenko, 2000), exist which are capable of creating formal contexts, but are time consuming and not ‘automated’, as they were not built for this purpose.

Figure 3 FcaBedrock process (see online version for colours)



FcaBedrock implements a novel approach, using a process of guided automation (Andrews and Orphanides, 2010a), meaning that while most processes of the tool are automated, it is the user who has the final say on how to interpret the data (guided automation is explained in Section 5 of this article). The user supplies the tool with appropriate metadata for conversion, such as the names of the attributes and their values, and with decisions as to what to convert and how to convert it. After reading in the original data file, these metadata are used by FcaBedrock to create a formal context file in a standard form for FCA (Figure 3). The metadata is stored in a separate text document called a *Bedrock* (.bed) file. This can be used for subsequent conversions and act as a record of the interpretation made of the dataset. Bedrock files can be

loaded into FcaBedrock, allowing the reproduction of context files and allowing changes in the interpretation to be made. User-defined constraints applied to the data allow different analyses to be carried out. Each analysis can be documented with a Bedrock file. Multiple data files with the same attributes can be converted using the same Bedrock file.

2.1 Input formats

FcaBedrock currently supports two input data file formats: many-column and three-column delimiter separated values (dsv). The comma and tab characters are currently supported as value delimiters. FcaBedrock outputs the formal context in the widely used *Burmeister* (.cxt) and *FIMI* (.dat) file formats.

Many-column dsv is a traditional flat-file format for many-valued data. Each row represents an instance (object) and each column a many-valued attribute.

The three-column format is becoming popular as a standardised data format and is the approach used, for example, in triple-stores (Rohloff et al., 2007) and the resource description format (RDF) (Abadi et al., 2009). FcaBedrock assumes that, when reading a three column data file, the first value is a formal object, the second is a many-valued attribute and the third is an attribute value. The first values are used by FcaBedrock as object names when outputting a formal context in the *Burmeister* format (see below). If auto-detection is used (this feature shall be explicated later on in this article), the second values are detected as attribute names. The order in which triples appear in a data file is not significant.

2.2 Output formats

The *Burmeister* formal context format (cxt) is a popular format in FCA. A cxt file begins with the number of objects and number of attributes and then lists the objects followed by the attributes. It then stores the body of the formal context as a grid using crosses for *true* values and dots for *false* values.

The *frequent itemset mining implementations* (FIMI) data format (dat) is used in data mining, particularly in testing the efficiency of algorithms (Goethals and Zaki, 2004). Unlike Burmeister, FIMI does not include object or attribute names, but just the formal attributes featured in each formal object.

3 Attribute values

Figure 4 shows FcaBedrock. There are three sections on top, for specifying how continuous and ordinal attributes should be scaled, what delimiter is used in the input file and if the first row in the input file is to be used as attribute names. There are windows for the names and types of the original data attributes, whether they are to be converted or not, their categories and the corresponding category values found in

the data file. All these must initially be entered by the user (unless FcaBedrock's auto-detection feature is being used – see Section 5.2), although copy and paste is available and useful if the names of the categories and the category values are the same. This is often but not always the case, so FcaBedrock uses both; the category values are required for the computation because these are the values that appear in the input file, but the category names are the ones that will appear in the output context file. This means that human-readable names can be given to code values, for example, to make the resultant concept lattice understandable. The example shown is of the well-known *mushroom* dataset from UCI (Frank and Asuncion, 2010), where arbitrary letters are used in the data file to represent attribute categories. The *cap-surface* attribute, for example, has four possible categories, *fibrous*, *grooves*, *scaly* and *smooth*, represented by the values *f*, *g*, *y* and *s*, respectively, in the data file. In another UCI example, the *adult* dataset has category values in the data file that each contains a leading space character. The spaces are better omitted from the category names to make them more readable, but are required for converting the data.

4 Attribute types

4.1 Categorical attributes

The predominant attribute type is categorical. This is the typical many-valued attribute and is converted by creating one formal attribute for each of the attribute categories. In the conversion process, each row of comma separated values read in from the data file is split into a list of individual values. Each value is compared with the category values listed for the corresponding attribute in the values (file) window. Attribute zero corresponds to the first column in the data file and so on. A match is recorded as a true value in the formal context.

In the *mushroom* example shown in Figure 4, all but one of the attributes are categorical. The *gill-spacing* attribute, for example, has three categories: *close*, *crowded*, and *distant*. FcaBedrock will create three corresponding formal attributes and name them by concatenating the attribute and category names, thus *gill-spacing-close*, *gill-spacing-crowded*, and *gill-spacing-distant*.

Figure 4 FcaBedrock showing metadata for the UCI *mushroom* dataset (see online version for colours)

FcaBedrock Context Creator v2

File Format Help

Continuous Scaling: ☐ Progressive ☒ Discrete

File Delimiter: ☒ CSV ☐ TSV

Misc: ☐ First row: attribute names

Attributes: 23 **Categories: 124** **Values (File): 124**

No.	Attributes: 23	Convert Type	Categories: 124	No.	Values (File): 124	No.
0	class	n	c edible, poisonous	2	e, p	2
1	cap-shape	y	c bell, conical, convex, f	6	b, c, x, f, k, s	6
2	cap-surface	y	c fibrous, grooves, scaly	4	f, g, y, s	4
3	cap-color	y	c brown, buff, cinnamon, g	10	n, b, c, g, r, p, u, e, w, y	10
4	bruises?	y	b bruises, no	2	t, f	2
5	odor	y	c almond, anise, creosote	9	a, l, c, y, f, m, n, p, s	9
6	gill-attachment	y	c attached, descending, f	4	a, d, f, n	4
7	gill-spacing	y	c close, crowded, distant	3	c, w, d	3
8	gill-size	y	c broad, narrow	2	b, n	2
9	gill-color	y	c black, brown, buff, choc	12	k, n, b, h, g, r, o, p, u, e,	12
10	stalk-shape	y	c enlarging, tapering	2	e, t	2
11	stalk-root	y	c bulbous, club, cup, equa	6	b, c, u, e, z, r	6
12	stalk-surface-above-	y	c fibrous, scaly, silky, s	4	f, y, k, s	4
13	stalk-surface-below-	y	c fibrous, scaly, silky, s	4	f, y, k, s	4
14	stalk-color-above-ri	y	c brown, buff, cinnamon, g	9	n, b, c, g, o, p, e, w, y	9
15	stalk-color-below-ri	y	c brown, buff, cinnamon, g	9	n, b, c, g, o, p, e, w, y	9

☐ Restrict Mode

Input Data File
File Name: agaricus-lepiota.data
Type: DSV
Objects: 8123
Attributes: 23

Bedrock File
File Name: agaricus-lepiota.bed
Created: 27/7/2011 5:02:49 PM
Modified: 14/1/2010 7:00:50 PM

Output Context File
File Name: agaricus-lepiota.ext
Type: Burmeister
Extension: .ext

Figure 5 UCI *Adult* metadata in FcaBedrock (see online version for colours)

Continuous Scaling:			File Delimiter		Misc	
<input type="radio"/> Progressive <input checked="" type="radio"/> Discrete			<input checked="" type="radio"/> CSV <input type="radio"/> TSV		<input type="checkbox"/> First row: attribute names	
No.	Attributes: 15	Convert Type	Categories: 120	No.	Values (File): 120	No.
0	age	y o	<, 20, 40, 60, >	4	<, 20, 40, 60, >	4
1	workclass	y c	State-gov, Self-emp	9	State-gov, Self-emp	9
2	fnlwgt	n o	12285, 296941, 593882, 8	5	12285, 296941, 593882, 8	5
3	education	y c	Bachelors, HS-grad,	16	Bachelors, HS-grad,	16
4	education-num	n o	<, 4, 8, 12, >	4	<, 4, 8, 12, >	4
5	marital-status	y c	Never-married, Marri	7	Never-married, Marr	7
6	occupation	y c	Adm-clerical, Exec-m	15	Adm-clerical, Exec-15	15
7	relationship	y c	Not-in-family, Husb	6	Not-in-family, Husb	6
8	race	y c	White, Black, Asian-	5	White, Black, Asian5	5
9	sex	y c	Male, Female	2	Male, Female	2
10	capital-gain	y o	<, 30000, 60000, 90000, >	4	<, 30000, 60000, 90000, 4	4
11	capital-loss	y o	<, 1000, 2000, 3000, 4000	5	<, 1000, 2000, 3000, 4005	5
12	hours-per-week	y o	<, 30, 60, >	3	<, 30, 60, >	3
13	native-country	y c	United-States, Cuba,	42	United-States, Cuba42	42
14	class	y c	<=50K, >50K	2	<=50K, >50K	2

4.2 Boolean attributes

A Boolean attribute can be interpreted as a single formal attribute. Typically, a Boolean attribute in a dataset is one that has two categories that represent *true* and *false*. In the *mushroom* example, the Boolean type is being used for the *bruises?* attribute. The attribute has two categories, *bruises* and *no*, represented in the data file by the values *t* and *f*. For a Boolean attribute, FcaBedrock uses the first category value as the *true* value, *t* in this example. During the conversion process, it compares the corresponding value read in from the data file with the *true* value. If they match, this is recorded as a *true* value in the formal context. FcaBedrock names the formal attribute using the original attribute name, without concatenating the name of the *true* category. So, in this example, the original *bruises?* Attribute becomes a single formal attribute also called *bruises?*.

4.3 Continuous attributes

Continuous attributes are dealt with in FcaBedrock by discretising the data using user-defined ranges (e.g., 0–10, 10–20...), or by hierarchical scaling (e.g., > 0, > 10, > 20...) (Priss, 2008). Figure 5 shows the metadata of the *Adult* dataset in FcaBedrock. The *age* attribute is to be converted as a continuous attribute. The boundaries of the ranges are entered as comma separated values. In this case, the boundaries are <, 20, 40, 60 and >. FcaBedrock will apply these boundaries as four categories: ‘less than 20, 20 to less than 40, 40 to less than 60 and greater than or equal to 60’. The formal attribute names are created by concatenating values appropriately: *age*< 20 and *age*-20 to < 40, for example. During the conversion process, FcaBedrock compares a continuous value read in from the data file with the appropriate boundary values, assigning a *true* value to the corresponding formal attribute in the formal context.

4.4 Ordinal attributes

Ordinal attributes are categorical attributes where the ordering of the attribute values is important. Taking as an example an attribute *Education* containing higher education degrees, it is important that a foundation course comes before a Bachelors degree, a Doctorate comes after a Masters degree and so on. Ordinality also allows the creation of sensible ranges to summarise values for non-continuous attributes. Using the same example, creating ranges for the *Education* attribute can be used to group education levels to undergraduate and postgraduate: Foundation-Bachelors, Masters-Doctorate. Capturing the order of non-numerical data is a feature that allows analyses that are not easy using traditional means of data analysis. FCA of such ordinal data can provide novel intelligence from a concept lattice, as will be seen later in this article.

Declaring ranges for ordinal attributes is possible in FcaBedrock using the same process used for declaring ranges for continuous attributes. For example, the boundaries <, *Bachelors*, *Masters* and > will be used by FcaBedrock to create three categories: ‘less than Bachelors, Bachelors to less than Masters and greater than or equal to Masters’. Discrete and hierarchical scaling is supported for ordinal attributes as well.

When an ordinal attribute has been declared, FcaBedrock assigns a numerical value to each of the attribute values. During the conversion process, FcaBedrock reads an attribute value, finds its numerical counterpart and compares it with the appropriate boundary values, assigning a *true* value to the corresponding formal attribute in the formal context.

4.5 Date attributes

The most common date formats are supported. As an example, creating ranges for a dates attribute can be used to

group months into seasons: 01/12/2010 to 28/02/2011, 01/03/2011 to 31/05/2011, 01/06/2011 to 31/08/2011 and 01/09/2011 to 30/11/2011. Declaring ranges for date attributes is possible in FcaBedrock using the same process used for declaring ranges for continuous attributes. For example, the boundaries 01/12/2010, 01/03/2011, 01/06/2011, 01/09/2011 and > will be used by FcaBedrock to create four categories: ‘01/12/2010 to less than 01/03/2011, 01/03/2011 to less than 01/06/2011, 01/06/2011 to less than 01/09/2011 and greater than or equal to 01/09/2011’. Adding hours, minutes and seconds to a date is also possible – 23/09/2011 01:55:40 AM is valid as a date value.

During the conversion process, FcaBedrock reads a date value and compares it with the appropriate boundary values, assigning a *True* value to the corresponding formal attribute in the formal context.

5 Guided automation

5.1 Attribute exclusion and restriction

It may be necessary or desirable to exclude attributes from a formal context for a number of reasons: the attribute may be unsuitable for conversion (if it is a free-text attribute for example) or it may be that a particular attribute is not of interest in the analysis being undertaken. The user of FcaBedrock can decide which attributes in the dataset to include in the formal context using the convert window. Entering a *y* (for *yes*) will mean that the corresponding attribute will be included in the conversion. Entering an *n* (for *no*) will mean that the corresponding attribute will be excluded from the conversion. Individual attribute categories can be excluded from the formal context by simply not including them in the list.

Sub-contexts can be also created by restricting the conversion to user-specified attribute values. By specifying one or more category values of one or more attributes, the formal context will only contain objects with those values. For example, taking the *mushroom* dataset, a sub-context containing only poisonous mushrooms can be created simply by specifying the category value *p* for the class attribute.

5.2 Auto-detection of metadata

If a data file is read without first loading or creating a Bedrock file, FcaBedrock can detect attribute values and create metadata automatically. It initially assumes that all attributes are categorical. It adds each new value it finds in a data-column to a corresponding list of category values. If FcaBedrock determines that there are a high proportion of different numerical values for an attribute, it provides the option of automatically creating ranges and declares the attribute as continuous. This also applies for dates – if FcaBedrock determines that there is a high proportion of different dates for an attribute, it provides the option of

automatically creating ranges and declares the attribute as date.

6 A mini-adult example

The following is an example adapted from the UCI *adult* dataset, to better illustrate the transformation of data into a formal context.

The example uses a data file (see File 1) called *mini-adult.data* with just eight instances and five attributes: *age*, *education*, *employment*, *sex* and *US-citizen*. There are two classes indicating a salary above or below \$50 k.

File 1 mini-adult.data

39,	Bachelors, Clerical, Male, Yes, <=50K
50,	Bachelors, Managerial, Female, Yes, <=50K
38,	HS-grad, Unskilled, Male, Yes, <=50K
53,	11th, Unskilled, Male, Yes, <=50K
28,	Bachelors, Professional, Female, Yes, >50K
37,	Masters, Managerial, Female, No, <=50K
49,	?, Clerical, Female, No, <=50K
52,	HS-grad, Managerial, Male, Yes, >50K

File 2 mini-adult.cxt, *Burmeister* context file.

B
8
15
0
1
2
3
4
5
6
7
age-<30
age-30to<40
age-40to<50
age->=50
education-Bachelors
education-Masters
education-11th
education-HS-grad
employment-Clerical
employment-Managerial
employment-Professional
employment-Unskilled
sex-Male

sex-Female
US-citizen
..X..X..X.X
...XX...X...XX
.X....X...XX.X
...X..X...XX.X
X...X.....X.XX
.X...X...X..X.
..X....X...X.
...X...X.X..X.X

The *mini-adult* metadata in FcaBedrock are shown in Figure 6. The output *Burmeister* context file, *mini-adult.cxt*, is shown in File 2.

Figure 6 *mini-adult* metadata in FcaBedrock (see online version for colours)

No.	Attributes: 6	Convert Type	Categories:	No.	Values (File):	No.
0	age	y	<,30,40,50,>	4	<,30,40,50,>	4
1	education	y	Bachelors,Masters,1lt	4	Bachelors,Masters,1lt	4
2	employment	y	Clerical,Managerial,F	4	Clerical,Managerial,F	4
3	sex	y	Male,Female	2	Male,Female	2
4	US-citizen	y	Yes,No	2	Yes,No	2
5	class	n	>50K,<=50K	2	>50K,<=50K	2

7 Analysing formal contexts for knowledge discovery

To illustrate how knowledge discovery is possible through analysing formal contexts, practical examples are given, using real datasets from the UCI machine learning repository.

7.1 Mushroom dataset

The mushroom dataset contains information of 8,124 edible and poisonous mushrooms of the *agaricus* and *lepiota* families. These are many-valued categorical data which have attributes that describe properties such as veil type, veil colour, population and habitat.

Figure 7 (a) *Agaricus Arvensis* and (b) *Agaricus Xanthodermus* (see online version for colours)

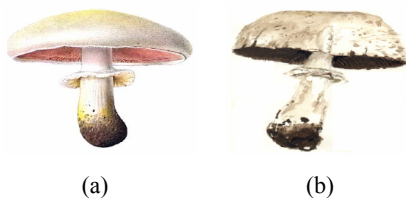


Figure 7 shows two mushrooms of the *agaricus* family; *Agaricus Arvensis* and *Agaricus Xanthodermus*. On first

sight, both mushrooms seem to be edible. However, *Agaricus Xanthodermus* has poisonous properties. This is a good example of how mushroom identification can be a confusing process for non-experts who are unable to distinguish subtle but important differences between mushrooms which resemble to each other. Figure 8 shows how the attribute restriction and attribute exclusion capabilities of FcaBedrock can be applied to create a sub-context of this dataset, in order to determine if a mushroom is safe to eat or not based on some of its characteristics. In this analysis, 20 attributes were excluded from this conversion, except from the class (edible and poisonous), *cap-colour* and *habitat* attributes. In addition, the *cap-colour* attribute was restricted to the values *red*, *pink*, *green* and *purple* and the *habitat* attribute was restricted to the values *woods*, *urban*, *meadows* and *grasses*. The restrictions set resulted in 996 objects (out of the initial 8,124) and 19 formal attributes (out of the initial 125). Essential to the analysis is the fact that this sub-context contains few enough formal concepts (nodes) to make the visualisation practical.

Visualising the sub-context in ConExp (Figure 9) produced some interesting information. By labelling each concept in the lattice with the object count and percentage of the overall number of objects, in ConExp, it appears that, in the sample, 98% of all the mushrooms live in woods, out of which 61% are safe to eat. It also appears that the most dominant mushrooms have a red cap, of which 576 are edible and 300 are poisonous, perhaps indicating that mushroom cap colours are not the safest indicator for determining their edibility. In contrast, pink-capped mushrooms can be found in woods and less frequently in grasses or meadows and all of them are poisonous. Mushrooms with a green or purple cap are extremely rare, but if one is lucky enough to find some they are safe to eat. Let us now say that we are interested in the relationship between mushroom habitat and population type. Once again, FcaBedrock’s attribute exclusion feature was used to create a sub-context containing only the habitat and population type attributes. There were 13 formal attributes in all, corresponding to the seven categories of habitat and the six categories of population type.

The concept lattice of this sub-context is shown in Figure 10. In ConExp, the size of the node in the lattice was made proportional to the number of own objects (mushrooms), so it can be seen that, for example, clusters of mushrooms are found in similar numbers in woods, leaves and waste ground, but not in other habitats. Solitary mushrooms are usually found in woods, although they can be occasionally found in paths, urban areas and grassland.

Figure 8 Creating a mushroom *class-habitat-cap colour* sub-context in FcaBedrock (see online version for colours)

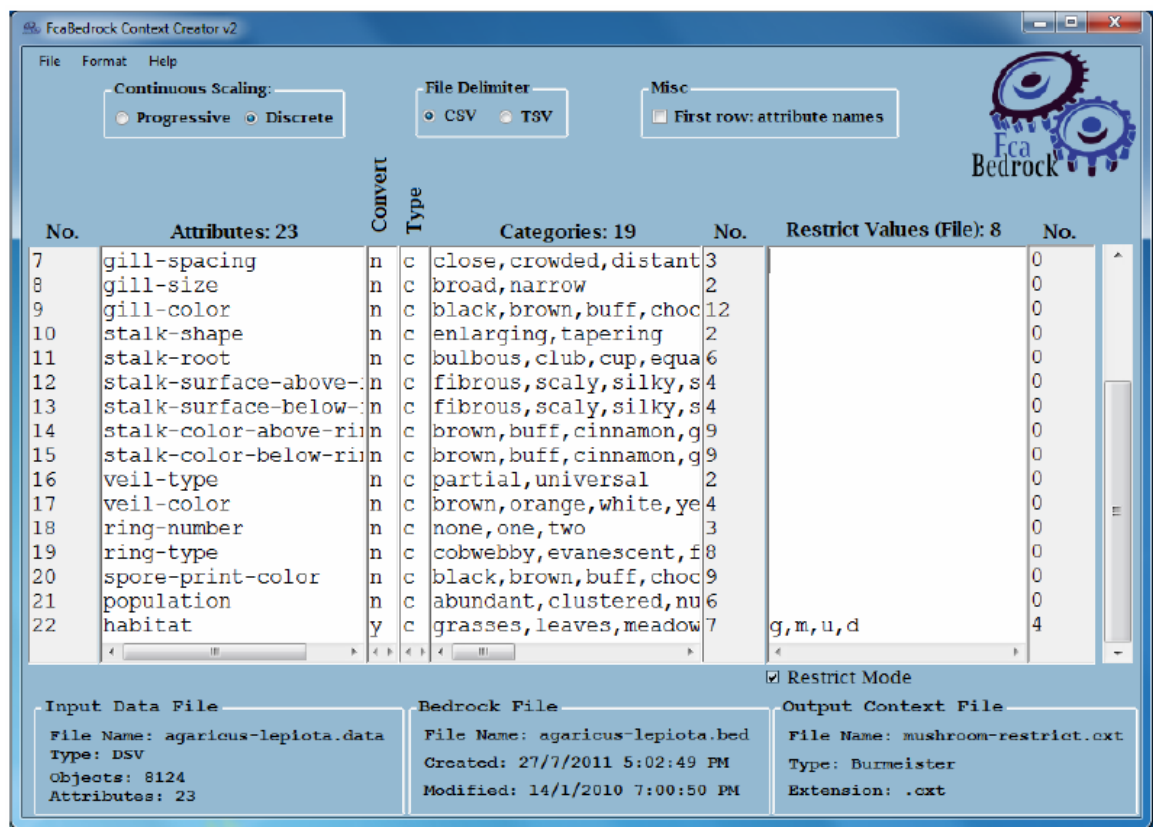
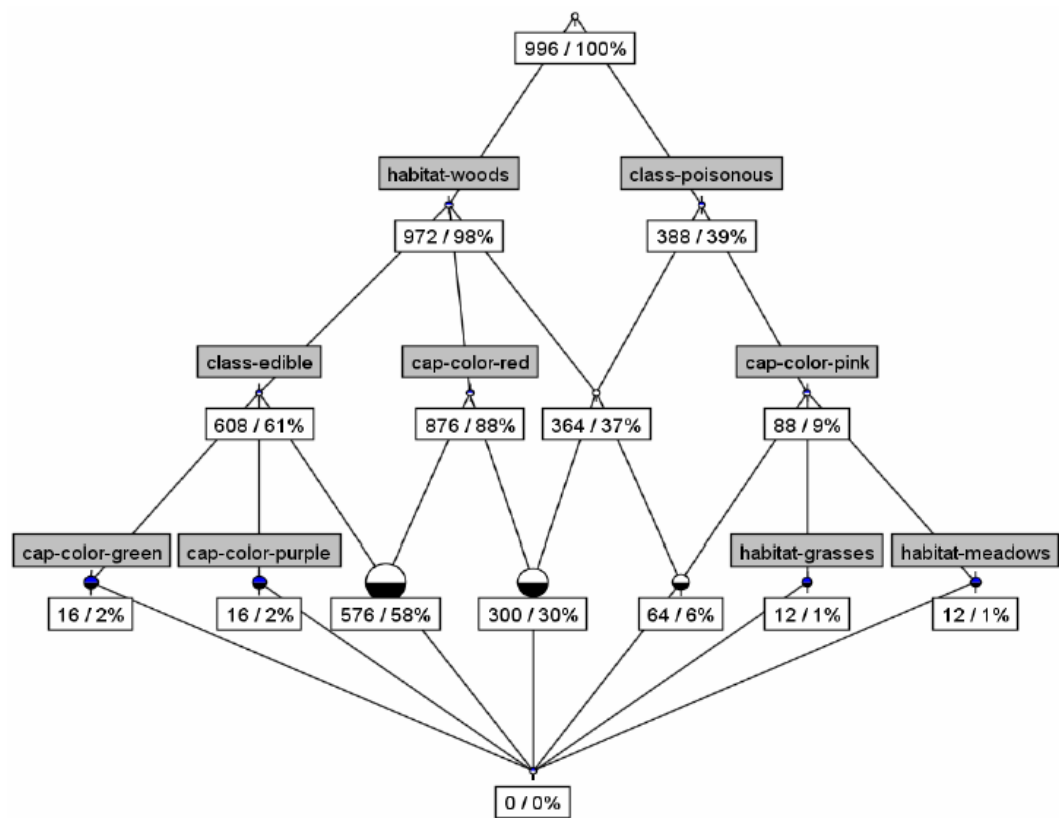


Figure 9 A mushroom *class-habitat-cap colour* sub-context, created by FcaBedrock, visualised in ConExp (see online version for colours)



were re-ordered from lower to higher education levels, as shown in Figure 14. The boundaries $<$, *Some-college*, *Bachelors* and $>$ were set for the education attribute, resulting in three ranges: $<Some-college$, *Some-college* to

$<Bachelors$ and $\geq Bachelors$. For the age attribute, the boundaries $<$, 20, 30 and 45 were set, resulting in three ranges: < 20 , 20 to < 30 and 30 to < 45 .

Figure 11 Adult degree-sex-pay sub-context, created by FcaBedrock (see online version for colours)

The screenshot shows the FcaBedrock Context Creator v2 window. It has a menu bar (File, Format, Help) and several configuration sections:

- Continuous Scaling:** Progressive (selected), Discrete.
- File Delimiter:** CSV (selected), TSV.
- Misc:** First row: attribute names (checked).

The main table lists attributes and their restrictions:

No.	Attributes: 15	Convert	Type	Categories: 20	No.	Restrict Values (File): 3	No.
0	age	n	o	<, 20, 40, 60, >	4		0
1	workclass	n	c	State-gov, Self-emp-	9		0
2	fnlwgt	n	o	12285, 296941, 593882, 85	5		0
3	education	y	c	Bachelors, HS-grad, 16	16	Bachelors, Masters,	3
4	education-num	n	o	<, 4, 8, 12, >	4		0
5	marital-status	n	c	Never-married, Marri	7		0
6	occupation	n	c	Adm-clerical, Exec-m	15		0
7	relationship	n	c	Not-in-family, Husba	6		0
8	race	n	c	White, Black, Asian-	5		0
9	sex	y	c	Male, Female	2		0
10	capital-gain	n	o	<, 30000, 60000, 90000, >	4		0
11	capital-loss	n	o	<, 1000, 2000, 3000, 4000	5		0
12	hours-per-week	n	o	<, 30, 60, >	3		0
13	native-country	n	c	United-States, Cuba,	42		0
14	class	y	c	<=50K, >50K	2		0

At the bottom, there are three sections:

- Input Data File:** File Name: adult.data, Type: DSV, Objects: 32561, Attributes: 15.
- Bedrock File:** File Name: adult.bed, Created: 27/7/2011 5:05:03 PM, Modified: 16/6/2010 9:42:50 PM.
- Output Context File:** File Name: adult-restrict.txt, Type: Burmeister, Extension: .ext.

The **Restrict Mode** checkbox is checked.

Figure 12 Adult degree-sex-pay sub-context, created by FcaBedrock, visualised in ConExp (see online version for colours)

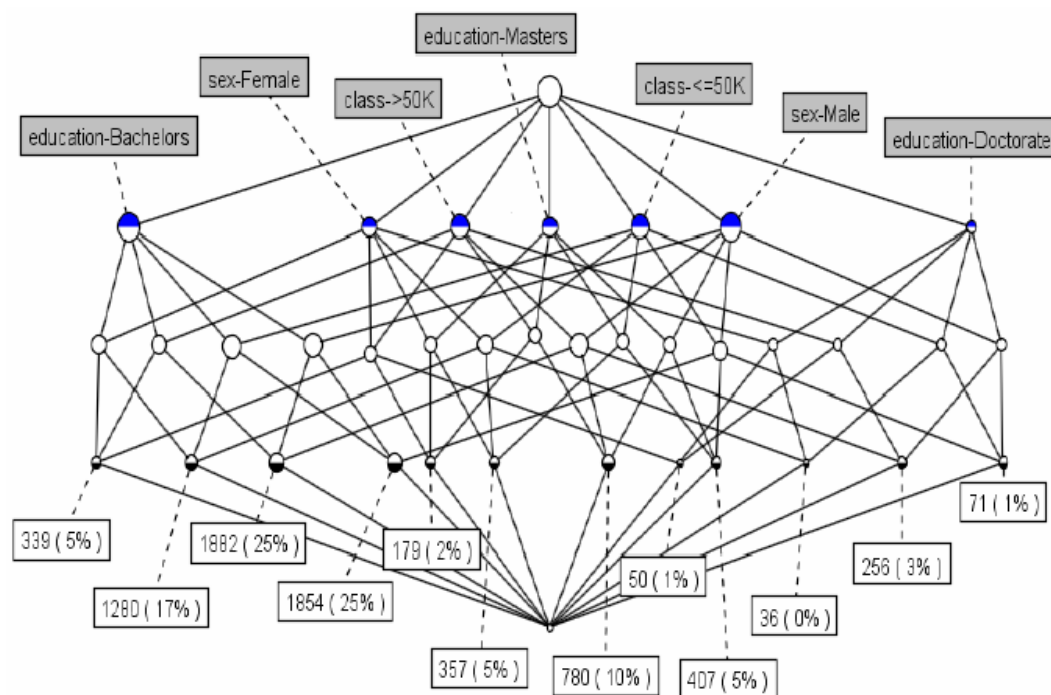


Figure 13 Adult age-hours per week sub-context, created by FcaBedrock, visualised in ConExp (see online version for colours)

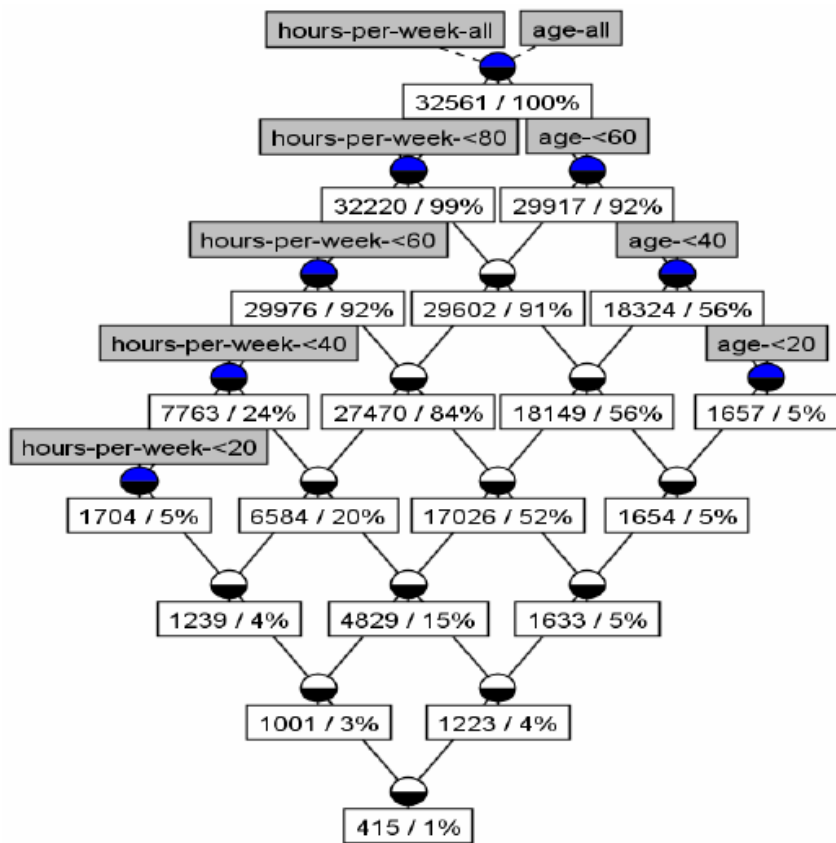


Figure 14 Adult age-pay-degree sub-context, created by FcaBedrock. Note the ordering of the attribute values of education: *Preschool, 1st-4th, 5th...* (see online version for colours)

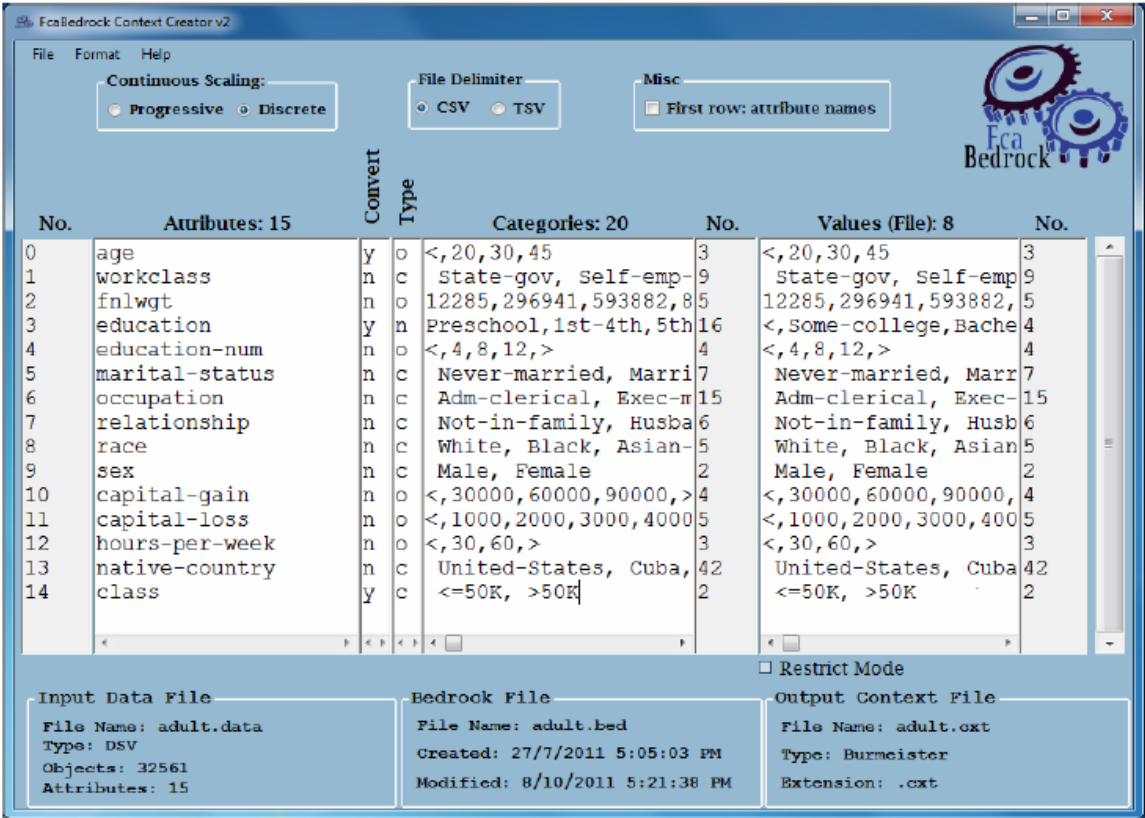
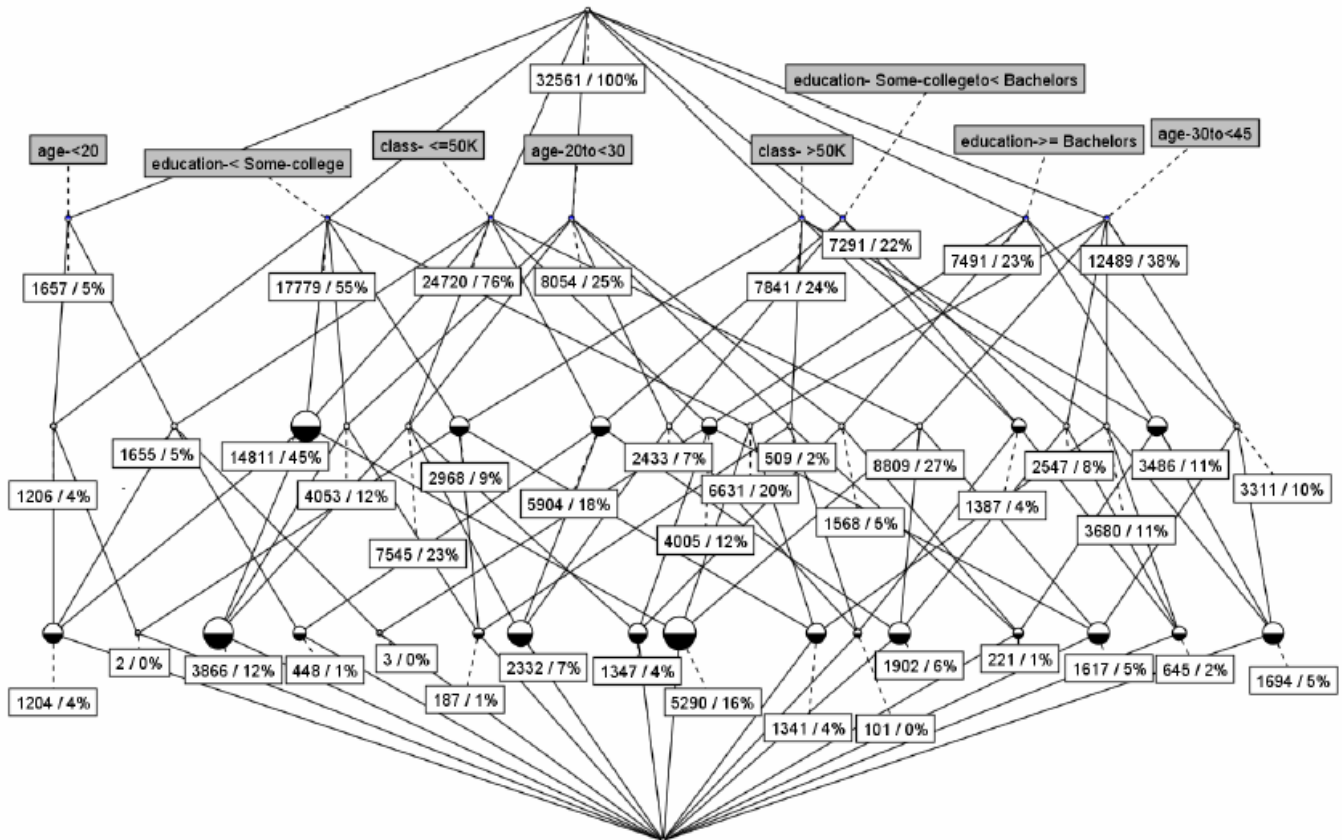


Figure 15 Adult *age-pay-degree* sub-context, created by FcaBedrock, visualised in ConExp (see online version for colours)**Figure 16** Parking tickets *issue date-offence* sub-context, created by FcaBedrock (see online version for colours)

FcaBedrock Context Creator v2

FileFormatHelp

Continuous Scaling:

Progressive

Discrete

File Delimiter:

CSV

TSV

Misc

First row: attribute names

FcaBedrock

No.

Attributes: 9

Convert Type

Categories: 27

No.

Values (File): 27

No.

0

Issue Date / Time

y

d

1/4/2009,1/6/2009,1/8/4

4

1/4/2009,1/6/2009,1/4

4

1

Notice No

n

c

FD30015526,FD23016336

100

FD30015526,FD2301633

100

2

Contravention Location

n

c

Ainsley Road : Ainsle

100

Ainsley Road : Ainsl

100

3

Offence

y

c

Road,Arundel Street :24

24

Road,Arundel Street

24

4

Officer ID

n

c

No permit,During load

5

Parked in a permit s

5

5

Current Status

n

c

906,198.2,396.4,594.6

16

906,198.2,396.4,594.16

16

6

Next Status

n

c

Notice cancelled,Char2

2

Notice cancelled,Cha

2

7

Payment Date/Time

n

d

,Selected for debt re

3

,Selected for debt r

3

8

Payment Method

n

c

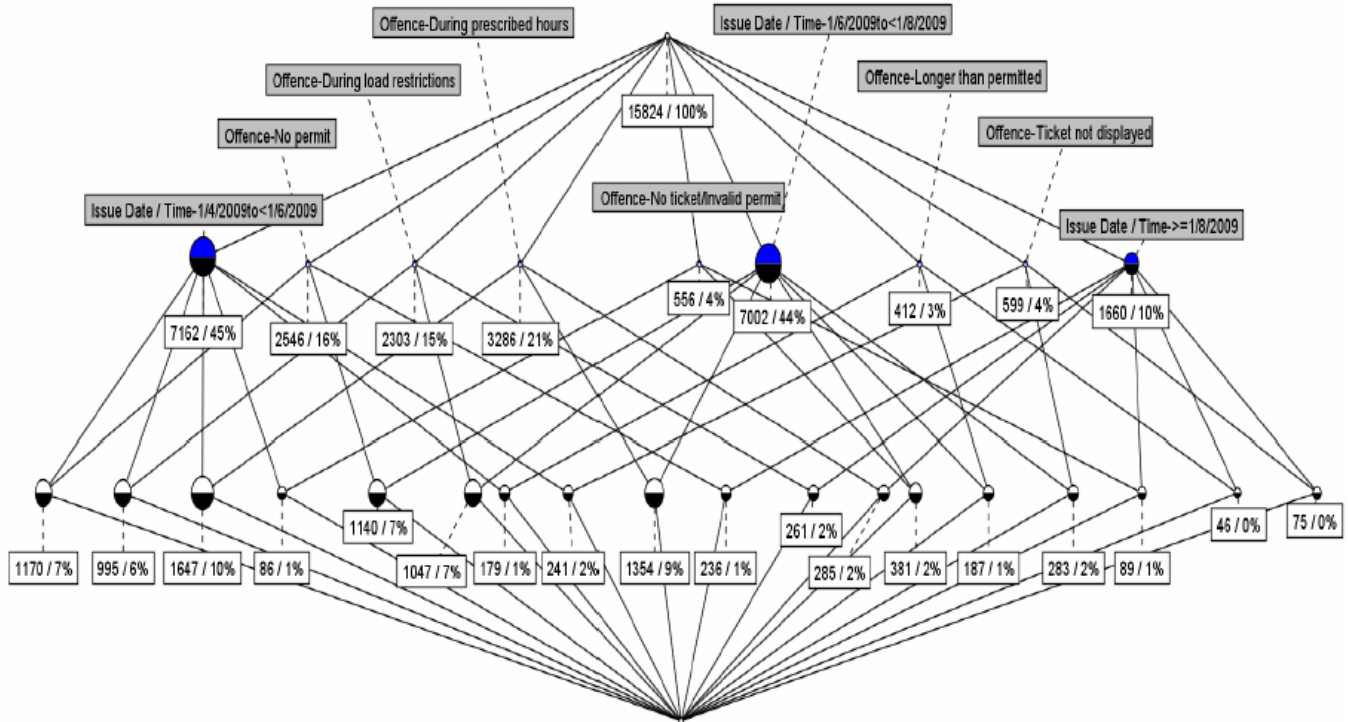
1/4/2009 12:00:00 AM,10

10

1/4/2009 12:00:00 AM

10

Note: How the *offence description* attribute was renamed to *offence* and how its values in the *categories* window were shortened, when compared to the values as they appear in the original data, in the *values (File)* window.

Figure 17 Parking tickets *issue date-offence* sub-context, created by FcaBedrock, visualised in ConExp (see online version for colours)

Although somewhat cluttered with lines, the resulting concept lattice in Figure 15 still shows some interesting results. It appears that 55% of the entire population has not attended college whatsoever, while the rest 45% is almost evenly distributed between attending some college and earning one or more degrees. Looking at the obvious, larger concepts (nodes) in the lattice, it can be seen that 45% of the population did not attend college and earn less than 50 k, out of which 35.7% are in the 30–45 age group, 26.1% are in the 20–29 age group and 8.1% are less than 20 years old. This could possibly indicate the economic and social difficulties of attending college for older generations and how education is considered an essential advantage in terms of career evolution for younger generations.

On the other hand, only 24% of the entire population earns more than 50k, out of which 29% belongs to the 30–45 group, with 48% of them having a college degree. For the 20–30 group, 19.4% have a college degree, out of which only 14% earn more than 50 k, although a similar percentage of this age group earns more than 50 k without attending college whatsoever.

Some of the smaller concepts yield useful intelligence too. For example, it appears that 187 people of the 20–30 age group and two people younger than 20 earn more than 50k and have not attended college. Similarly, there are 101 people of the 20–30 group which are college dropouts and earn more than 50k. It appears that, for some, lack of educational attainment has not been a barrier to gainful employment. Indeed, a significant few have forgone further education to pursue early success.

7.3 Sheffield parking tickets dataset

This dataset is provided by the City Council of Sheffield (UK) and contains raw data of parking tickets issued in the name of the council within the last five-year period, broken down by date, time, location, narrative of infraction, current status of case, next status of case, transaction details (i.e., date, time and payment methods) and the IDs of the officers issuing the tickets.

Let us say that we are interested in seeing what kind of offences have taken place for the reporting period of April 2009 to August 2009. For this analysis to take place, the *Issue Date/Time* and *offence description* attributes were converted. The *issue date/time* attribute was grouped into three ranges: April–June, June–August and greater than August. Furthermore, in order to avoid long labels in the lattice, the *offence description* attribute was shortened to *offence*, and each offence description was given a shorter title than the ones appearing in the dataset; for example, ‘Parked in a permit space without displaying a valid permit’ was renamed to ‘No permit’. These modifications were made directly in FcaBedrock without altering the original dataset whatsoever, by just modifying the names of the offence descriptions, as they would appear in the formal context, in the *Categories* window (Figure 16).

The resulting concept lattice is shown in Figure 17. Only the first six offences are displayed – the rest have been deselected, in ConExp, to make the lattice more readable. The lattice shows that the most dominant of the six offences is parking in a restricted street during prescribed hours, with a percentage of 21%, while the least dominant offence is parking for longer than permitted, with a percentage of 3%. The lattice also shows that a similar amount of parking

tickets have been issued for the periods 1/4/2009–1/6/2009 and 1/6/2009–1/8/2009 (7,162 and 7,002, respectively). Only 1660 tickets have been issued from 01/08/2009 onwards, but it should be noted that these data only contain entries up to 10/8/2009. The same offence ‘popularity’ applies when observing each time period separately, apart from the ten-day reporting period of August, where the most popular offences are evenly distributed between parking during prescribed hours, parking during load restrictions and parking without a permit.

8 Conclusions

Knowledge discovery is possible through creating formal contexts and then visualising them. FcaBedrock has been used successfully to convert large datasets into formal contexts. Where the concepts in a formal context become far too many to visualise, FcaBedrock’s attribute restriction and exclusion features have been used to create sub-contexts of the same data, so as to focus the analysis on specific business questions and to make the resulting concept lattices readable and manageable. Furthermore, datasets containing different kinds of attribute have been used to demonstrate how FcaBedrock can handle datasets of different nature, although free-text data have proven to still be a problem and no means for converting them has been implemented yet. For converting free-text data, further work is required, such as implementing a natural language processing (NLP) technique, in order to identify values.

The tool is straightforward to use: the metadata required are usually easy to obtain (datasets usually come with descriptive documentation) and enter, but even without descriptive documentation, the auto-detection and repeat features can be used to obtain the initial metadata, after which the user can modify them as required. In terms of converting attributes, the five attribute types categorical, Boolean, continuous, ordinal and dates are straightforward and easy to convert, although more means for dealing with attributes are being considered for the next version of FcaBedrock, such as giving the user the ability to create ranges for continuous attributes based on certain characteristics of the numerical data under investigation. For example, if during the auto-detection phase the tool determines that the attribute being read is logarithmic, it could let the user know that this is the case and suggest ranges suitable for demonstrating the true distribution of that attribute.

FcaBedrock is a versatile tool. It currently supports two standard input formats and more formats will be added to the next version of the tool, such as XML and the ability to read and query RDBMS data. The two output formats Burmeister and FIMI allow interoperability between various FCA tools, in order to avoid unnecessary and time-consuming conversions between FCA file formats.

Through a process of guided automation, FcaBedrock gives the user control over the conversion process to interpret data and create sub-contexts, to suit the needs of an analysis.

The further development of FcaBedrock will form a core part of the CUBIST project (see Acknowledgements, below). CUBIST aims to develop an approach for semantic and user-friendly business intelligence (BI) by augmenting semantic technologies with BI capabilities and providing conceptually relevant, user-friendly, visual analytics. To this end, FcaBedrock is currently being developed to accept RDF formats as input and pulling data directly from triple stores. In addition, the NLP element of CUBIST will make intelligence available from unstructured data sources. The project is also providing industrial use cases to demonstrate that the creation of formal contexts from industrial data can yield important business intelligence. The data provided by the use cases are large in volume (millions of objects, thousands of attribute values). Optimisations are taking place in FcaBedrock to significantly reduce the time required for parsing the data and converting them to formal contexts: faster methods for handling attributes with many attribute values, indexing the data using integers rather than strings, better algorithms for reading triples, and making the tool scalable to the processing cores of the environment it is running on, so as to make FCA of large-scale data less time-consuming and less resource-intensive.

Acknowledgements

This work is part of the CUBIST (<http://www.cubist-project.eu>)⁴ project (‘Combining and Uniting Business Intelligence with Semantic Technologies’), a research project funded by the European Commission’s 7th Framework Programme of ICT, under topic 4.3: Intelligent Information Management.

This article is an extended version of the paper with the same name presented at the second International Conference on Intelligent Networking and Collaborative Systems (INCoS) 2010 (Andrews and Orphanides, 2010b).

References

- Abadi, D.J., Marcus, A., Madden, S.R. and Hollenbach, K. (2009) ‘SW-store: a vertically partitioned DBMS for semantic web data management’, in *VLDB*, Vol. 18, pp.385–406, Springer-Verlag.
- Andrews, S. (2009a) ‘Data conversion and interoperability for FCA’, in *CS-TIW 2009*, pp.42–49, available at http://www.kde.cs.uni-kassel.de/ws/cs-tiw2009/proceedings_final_15July.pdf.
- Andrews, S. (2009b) ‘In-Close, a fast algorithm for computing formal concepts’, in Rudolph, Dau and Kuznetsov (Eds.): *Supplementary Proceedings of ICCS’09, CEUR WS 483*.
- Andrews, S. (2011) ‘In-Close2, a high performance formal concept miner’, in *Proceedings of the 19th International Conference on Conceptual Structures (ICCS) 2011*. *LNAI*, Vol. 6828, pp.50–62, Springer-Verlag, Berlin.
- Andrews, S. and Orphanides, C. (2010a) ‘FcaBedrock, a formal context creator’, in Croitoru, M., Ferre, S. and Lukose, D. (Eds.): *Conceptual Structures: From Information to Intelligence, Proceedings of the 18th International Conference on Conceptual Structures (ICCS) 2010*. *LNAI*, Vol. 6208, pp.181–184, Springer, Berlin.

- Andrews, S. and Orphanides, C. (2010b) 'Knowledge discovery through creating formal contexts', in Hill, R. (Ed.): *First International Workshop on Computational Intelligence in Networks and Systems (CINS 2010)*, in Xhafa, F., Demetriadis, S., Caballe, S. and Abraham, A. (Eds.): *Second International Conference on Intelligent Networking and Collaborative Systems (INCOS) 2010*, IEEE Computer Society, pp.455–460, ISBN: 978-0-7695-4278-2/10, DOI 10.1109/INCOS.2010.53.
- Arevalo, G., Berry, A., Huchard, M., Perrot, G. and Sigayret, A. (2007) 'Performances of Galois sub-hierarchy-building algorithms', in Kuznetsov, S.O. and Schmidt, S. (Eds.): *ICFCA 2007, LNAI*, Vol. 4390, pp.166–180, Springer-Verlag, Berlin.
- Boulicaut, J-F. and Besson, J. (2008) 'Actionability and formal concepts: a data mining perspective', in Medina, R. and Obiedkov, S. (Eds.): *ICFCA 2008. LNCS (LNAI)*, Vol. 4933, pp.14–31, Springer-Verlag, Berlin/Heidelberg.
- Frank, A. and Asuncion, A. (2010) 'UCI machine learning repository', University of California, School of Information and Computer Science, Irvine, CA, available at <http://archive.ics.uci.edu/ml> (accessed on November 2011).
- Ganter, B. (1984) 'Two basic algorithms in concept analysis', Technical Report FB4-Preprint No. 831, TH Marstadt.
- Ganter, B. and Wille, R. (1998) *Formal Concept Analysis: Mathematical Foundations*, Springer-Verlag, Berlin.
- Goethals, B. and Zaki, M. (2004) 'Advances in frequent itemset mining implementations: report on FIMI'03', in *SIGKDD Explorations Newsletter*, ACM, New York. Vol. 6, No. 1, pp.109–117.
- Ignatov, D.I. and Kuznetsov, S.O. (2009) 'Frequent itemset mining for clustering near duplicate web documents', in Rudolph, S., Dau, F. and Kuznetsov, S.O. (Eds.): *ICCS 2009. LNCS (LNAI)*, Vol. 5662, pp.185–200, Springer-Verlag, Berlin/Heidelberg.
- Kaytoue-Uberall, M., Duplessis, S. and Napoli, A. (2008) 'Using formal concept analysis for the extraction of groups of co-expressed genes', in Le Thi, H.A., Bouvry, P. and Pham Dinh, T. (Eds.): *MCO 2008. CCIS*, Vol. 14, pp.439–449, Springer-Verlag, Berlin/Heidelberg.
- Krajca, P., Outrata, J. and Vychodil, V. (2008) 'Parallel recursive algorithm for FCA', in Belohlavek, R. and Kuznetsov, S.O. (Eds.): *CLA 2008*, Palacky University, Olomouc, pp.71–82.
- Kuznetsov, S.O. (1999) 'Learning of simple conceptual graphs from positive and negative examples', in *Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery, Lecture Notes in Computer Science*, Vol. 1704, pp.384–391, Springer-Verlag, London.
- Kuznetsov, S.O. and Obiedkov, S.A. (2002) 'Comparing performance of algorithms for generating concept lattices', in *Journal of Experimental and Theoretical Artificial Intelligence*, Vol. 14, Nos. 2–3, pp.189–216.
- Poelmans, J., Elzinga, P., Dedene, G., Viaene, S. and Kuznetsov, S.O. (2011) 'A concept discovery approach for fighting human trafficking and forced prostitution', in Andrews, S., Polovina, S., Hill, R. and Akhgar, B. (Eds.): *ICCS 2011. LNCS (LNAI)*, Vol. 6828, pp.201–214.
- Priss, U. (2008) 'Formal concept analysis in information science', in Cronin, B. (Ed.): *Annual Review of Information Science and Technology*, ASIST, Vol. 40.
- Riout, F., Robardet, C., Blachon, S., Crémilleux, B., Gandrillon, O. and Boulicaut, J-F. (2003) 'Mining concepts from large SAGE gene expression matrices', in Boulicaut, J-F. and Dzeroski, S. (Eds.): *WKDID 2003*, Catat-Dubrovnik, Catat-Dubrovnik, Croatia, pp.107–118.
- Rohloff, K. et al. (2007) 'An evaluation of triple-store technologies for large data stores', in *On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops, LNCS*, Vol. 4806, pp.1105–1114.
- Sawase, K., Nobuhara, H. and Bede, B. (2009) 'Visualizing huge image databases by formal concept analysis', in Barjiela, A. and Pedrycz, W. (Eds.): *Human-Centric Information Processing Through Granular Modelling. Studies in Computational Intelligence*, Vol. 182, Springer, Berlin/Heidelberg.
- Wille, R. (2005) 'Formal concept analysis as mathematical theory of concepts', in Ganter, B., Stumme, G. and Wille, R. (Eds.): *Formal Concept Analysis: Foundations and Applications*, Springer, Berlin, pp.1–6.
- Wolff, K.E. (1993) 'A first course in formal concept analysis', available at http://www.fbmh.h-da.de/home/wolff/Publikationen/A_First_Course_in_Forma_Concept_Analysis.pdf (accessed on 5 September 2011).
- Yevtushenko, S.A. (2000) 'System of data analysis 'concept explorer'', (In Russian), in *Proceedings of the 7th National Conference on Artificial Intelligence KII-2000*, Russia, pp.127–134.
- Zaki, M.J. and Hsiao, C-J. (2005) 'Efficient algorithms for mining closed itemsets and their lattice structure', in *IEEE Transactions on Knowledge and Data Mining*, Vol. 17, No. 4, IEEE Computer Society.

A conceptual approach to gene expression analysis enhanced by visual analytics

MELO, Cassio, ORPHANIDES, Constantinos, MCLEOD, Kenneth, AUFAIRE, Marie-Aude, ANDREWS, Simon <<http://orcid.org/0000-0003-2094-7456>> and BURGER, Albert

Available from Sheffield Hallam University Research Archive (SHURA) at:

<http://shura.shu.ac.uk/7158/>

This document is the author deposited version. You are advised to consult the publisher's version if you wish to cite from it.

Published version

MELO, Cassio, ORPHANIDES, Constantinos, MCLEOD, Kenneth, AUFAIRE, Marie-Aude, ANDREWS, Simon and BURGER, Albert (2013). A conceptual approach to gene expression analysis enhanced by visual analytics. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing. New York, NY, USA, ACM New York, 1314-1319.

Copyright and re-use policy

See <http://shura.shu.ac.uk/information.html>

A Conceptual Approach to Gene Expression Analysis Enhanced by Visual Analytics

Cassio Melo
École Centrale Paris
Châtenay-Malabry, France
cassio.melo@ecp.fr

Marie-Aude Aufaure
École Centrale Paris
Châtenay-Malabry, France
ma.aufaure@ecp.fr

Constantinos Orphanides
Sheffield Hallam University
Sheffield, United Kingdom
c.orphanides@shu.ac.uk

Simon Andrews
Sheffield Hallam University
Sheffield, United Kingdom
s.andrews@shu.ac.uk

Kenneth McLeod
Heriot-Watt University
Edinburgh, Scotland
kenneth.mcleod@hw.ac.uk

Albert Burger
Heriot-Watt University
Edinburgh, Scotland
a.g.burger@hw.ac.uk

ABSTRACT

The analysis of gene expression data is a complex task for biologists wishing to understand the role of genes in the formation of diseases such as cancer. Biologists need greater support when trying to discover, and comprehend, new relationships within their data. In this paper, we describe an approach to the analysis of gene expression data where overlapping groupings are generated by Formal Concept Analysis and interactively analyzed in a tool called CUBIST. The CUBIST workflow involves querying a semantic database and converting the result into a formal context, which can be simplified to make it manageable, before it is visualized as a concept lattice and associated charts.

Categories and Subject Descriptors

H.4 [Information Systems Applications]: Miscellaneous;
J.3 [Life and Medical Sciences]: Medical information systems

General Terms

Algorithms, Experimentation.

Keywords

Formal Concept Analysis, Bioinformatics, Visual Analytics, Association Rules

1. INTRODUCTION

Biology is increasingly an information-centric discipline. This amplifies the biologist's need for greater support when trying to discover, and comprehend, new relationships within the data [4]. In particular, the analysis of gene expression is of vital importance to the understanding of the phenomena

leading to diseases such as cancer, alzheimer's disease and multiple sclerosis.

A gene is a unit of instructions that directs the body how to do one essential task, i.e. create a protein. Gene expression information describes whether or not a gene is expressed (active) in a location. There are many types of gene expression experiment. This work focuses on a technology called *in situ* hybridisation (ISH) gene expression. Completed ISH experiments are published online. For the mouse, one of the main resources in this field is EMAGE¹.

A number of computational methods have been proposed to help biologists discover unexpected patterns and formulate interesting hypotheses. Popular techniques employ unsupervised classification methods such as clustering, to group and visualise co-expressed genes (see [15] for a survey). The main limitation of most clustering algorithms is that they do not allow clusters to overlap, a counter-intuitive idea in this domain as genes are not restricted to a specific function and usually take part in several biological processes when interacting with other genes.

Other methods use Formal Concept Analysis (FCA) to mine frequent patterns of gene expression data. FCA emerged in the early 80's as a mathematical framework to reveal co-occurrence patterns between sets of objects and attributes, usually depicted in a hierarchy of partially-ordered concepts in a structure known as concept lattice.

One of the main motivations for the use of FCA in life sciences comes from the idea that FCA provides an intuitive understanding of generalisation and specialisation relationships among objects and their attributes. In [10] for example, FCA was used to extract groups of genes with similar expressions profiles from data of the fungus *Laccaria bicolor*. In [5] authors developed a tool to query a set of extracted formal concepts in a human gene expression data set according to various criteria (e.g. presence of a keyword in a gene description) and then to cluster concepts according to similarity, in terms of the attributes (samples) and the objects (genes above a threshold of expression) constituting the concepts. They called these clusters *quasi-synexpression-groups* (QSGs).

In common with previous works is the traditional Hasse diagram, used to represent concept lattices. In particular, they suffer from considerable edge crossings when number of

¹<http://www.emouseatals.org/emage/>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC'13 March 18-22, 2013, Coimbra, Portugal.

Copyright 2013 ACM 978-1-4503-1656-9/13/03 ...\$10.00.

concepts exceed a few dozen, as is the case of gene expression data, leading to reduced graph readability and aesthetics [12]. Visual analytics can be seen as an integral approach combining visualization, human factors and data analysis [11] and can greatly enhance the visualisation and exploration of concept lattices [1]. The use of Visual Analytics in the biology domain is a promising alternative to the large cluster analysis as demonstrated by Akand *et al.* in [1]. In [1] an algorithm is proposed that generates a browse-able concept lattice, allowing incremental exploitation of large concept lattices on a biological data set.

In this work we propose a combined approach using FCA, Association Rules and Visual Analytics to address the challenges of gene expression analysis, through filtering and clustering of large amounts of data, interactive exploration of the data, display of relevant statistics and in particular, identification of inconsistencies. We developed a tool, called CUBIST, that implements the approach, addressing those challenges.

This work is part of a three-year project which aims to unite Semantic Technologies and Business Intelligence in order to facilitate analysis of large volumes of structured and unstructured data.

The remainder of the paper is organized as follows. Section 2 briefly introduces the EMAGE data set. In Section 3 the CUBIST user interface and its functionalities are introduced, after a short introduction to Formal Concept Analysis. Finally, the conclusion and future work are presented in Section 4.

2. EMAGE DATA SET

A *gene* is a unit of instructions that provides directions for one essential task. *Gene expression* information describes whether or not a gene is expressed (active) in a location, e.g., the gene helps create the heart.

In situ hybridization (ISH) gene expression information is given in relation to a tissue in an organism. Here the organism is the *developmental mouse*. This is the mouse from the moment it is conceived until it is born - this period is split into 26 distinct stages called Theiler Stages (TS). Each stage has its own anatomy (ontology) called EMAP².

The result of an ISH experiment is analyzed under a microscope by a human expert. That expert provides a series of triples (gene - level of expression - tissue) to describe the result., e.g., *bmp4* is weakly expressed in the heart TS17. "Level" is a textual description of how active (important) the gene is in the development of the tissue - *strongly* expressed genes are very active, *weakly* expressed genes less so, and genes that are *not expressed* are inactive.

The triples are published online in a resource such as EMAGE³. In this paper tissues may be referred to by EMAP IDs rather than their full name. Therefore the "heart TS17" becomes "EMAP:2411".

A subset of the EMAGE data set, covering the textual annotations, has been provided to the CUBIST project and federated into a semantic repository.

ISH information enables biologists to discover relationships between genes, for example, when genes are active in the same location. Such information provides insights into the ways in which relationships between genes affect the de-

velopment of a tissue. Comparisons between healthy and abnormal tissues lead to a better understanding of diseases (e.g. cancer) and birth defects (e.g. a cleft lip).

Regrettably, the tools for analyzing this information are still in an early stage of development. Accordingly, sophisticated analysis of the data is beyond the average biologist. There is a need to enable every biologist to perform complex investigations of their data. Yet, it is important to remember that biologists are not computer scientists, i.e. the tools should be powerful, yet easy to understand and use.

3. BACKGROUND ON FCA

Before proceeding, we would like to recall the FCA terminology. In mathematical terms, a formal context is defined as a triple $\mathbb{K} = (G, M, I)$, with G being a set of objects, M a set of attributes and I a relation defined between G and M . The relation I is understood to be a subset of the cross product between the sets it relates, so $I \subseteq G \times M$. If an object g has an attribute m , then $g \in G$ relates to m by I , so we write $(g, m) \in I$, or gIm . For a subset of objects $A \subseteq G$, a derivation operator $'$ is defined to obtain the set of attributes, common to the objects in A , as follows:

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

In a similar manner, for a subset of attributes $B \subseteq M$, the derivation operator $'$ is defined to obtain the set of objects, common to the attributes in B , as follows:

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

A pair (A, B) is a *formal concept* in a given formal context (G, M, I) only if $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. The set A is the extent of the concept and the set B is the intent of the concept. A formal concept is, therefore, a closed set of object/attribute relations, in that its extension contains all objects that have the attributes in its intension, and the intension contains all attributes shared by the objects in its extension. The table below shows an example of a formal context of genes and EMAP identifiers in which they were detected. Genes *Upk2*, *Itpr3* and identifiers EMAP:7847, EMAP:7364, EMAP:7363 constitute a formal concept because objects *Upk2* and *Itpr3* share all attributes EMAP:7847, EMAP:7364, EMAP:7363 and attributes EMAP:7847, EMAP:7364, EMAP:7363 are featured by both objects *Upk2* and *Itpr3* (thus making the relation closed).

Expression	EMAP:7847	EMAP:7364	EMAP:7363	EMAP:7371	EMAP:8385	EMAP:7204
Upk2	×	×	×			
Itpr3	×	×	×	×		×
Cops7b			×	×		
Tgfbi	×				×	

However, density and 'noise' of a formal context are factors that can dramatically increase the number of formal concepts, which can result in unmanageable concept lattices. In this case, relevance measures such as stability and support can minimise input data needed for the computation of concept lattices [9]. In the subsequent sections we will describe the techniques we used to deal with noise in data and reduce the number of concepts generated.

²<http://www.emouseatlas.org/emap/ema/home.html>

³<http://www.emouseatlas.org/emage/>

4. A CONCEPTUAL APPROACH TO GENE EXPRESSION ANALYSIS ENHANCED BY VISUAL ANALYTICS

In [3] authors presented an approach to use FCA to analyse large clusters of gene co-expressions. The approach makes use of the formal context creator FcaBedrock⁴ and the formal concept miner In-Close⁵ (the aforementioned tools are being redeveloped in CUBIST) to convert and simplify formal contexts. The workflow is explained as follows. The user supplies metadata for conversion in FcaBedrock, such as the names of the genes or tissues and their values, and with decisions as to what to convert and how to convert it. These metadata are used to create a formal context. User-defined constraints, such as object exclusion, attribute exclusion and attribute restriction, applied to the data, allow different analyses to be carried out and the creation of sub-contexts which only focus on particular portions of the data. Next, In-Close is applied to the context file generated in the previous step. Using a trial and error approach, the user has to find an appropriate minimum size to mine a small number of large concepts, i.e., find the largest co-expressions of genes within the data. Finally, a third tool, Concept Explorer⁶ was used to visually display the corresponding concept lattice.

Whilst the above workflow required three standalone tools and FCA expertise, the same workflow has now been incorporated in CUBIST. Most of the complexity has been hidden, empowering the biologists to run the entire workflow themselves. Additionally, whilst FCA could only be visualised via a static lattice, CUBIST provides a series of analytical features and is able to deal with the implicit relationships and inconsistencies in the EMAGE data.

To illustrate, gene expression information should propagate through the mouse because the anatomy is organized as a series of part-of relationships, e.g. the paw is part-of the limb. If a gene is expressed in the paw, it is also found in the limb. Likewise, if a gene is not expressed in the limb it cannot be found in the paw. So when one experiment suggests a gene is expressed in the paw, and a second experiment shows that the same gene is not expressed in the limb, there is an inconsistency to be handled.

In the following sections we describe the methods to scale and convert the formal context from a semantic database; Visually explore clusters of expressed genes; Identify and treat cases of noise and inconsistencies in the data and; Highlight patterns of co-occurrence with association rules.

4.1 Querying and Converting the Gene Expression Ontology Data to Formal Contexts

In contrast with traditional FCA, which takes as input a binary table of objects and attributes, our approach is based on the querying of ontology data which is then converted to a formal context in a process transparent to the user. The conceptual analysis of ontologies provides unique information of gene expression data, by grouping entities belonging to particular properties in a hierarchical fashion and highlighting patterns of co-occurrence for those groups. Because biologists have little or no knowledge of SPARQL, the language we use to query ontologies, a set of pre-defined queries

⁴<http://sourceforge.net/projects/fcabedrock>

⁵<http://sourceforge.net/projects/inclose>

⁶<http://sourceforge.net/projects/conexp>

are available.

The procedure consists of translating each object o , attribute a and their incidence relation I in the result table such as $o \in G$ and $a \in M$, to create the formal context $\mathbb{K} = (G, M, I)$. As the number of variables in a query can be arbitrary, CUBIST has an option that allows users to select whether a given column in the result table is an object column, attribute or none. Contrarily to traditional FCA tools, in CUBIST the formal context is transparent to the user. The data is displayed in a table where its rows contain attributes, followed by attribute values and objects, in the columns (Figure 1). We found that this structure is more accessible to non-FCA experts. Filtering, sub-selection and conversion operations are possible through functionalities that came from the aforementioned tool FcaBedrock.

Attribute	Values	Objects
<input checked="" type="checkbox"/> embryo	level_detected , level_not_detected , level_weak ,	Arc, Foxa2, Smad2, Otx2, Smad5, Smad1
<input checked="" type="checkbox"/> endoderm	level_detected , level_not_detected ,	Foxa2, Zic2, Zic3
<input checked="" type="checkbox"/> definitive endoderm	level_detected ,	Hhex, Cer1
<input checked="" type="checkbox"/> parietal endoderm	level_strong ,	Fst
<input checked="" type="checkbox"/> extraembryonic ectoderm	level_detected ,	Bmp4, Zic2
<input checked="" type="checkbox"/> ectoderm	level_detected ,	Wnt3, Smad5, Otx2, Eomes, Zic2, TdGF1, Zic3, T
<input checked="" type="checkbox"/> mesoderm	level_detected ,	Wnt3, Smad5, Lefty2, Eomes, T, Mesp1, TdGF1
<input checked="" type="checkbox"/> primitive streak	level_detected , level_strong , level_not_detected ,	Lefty2, Eomes, Gsc, Foxa2, T, Fgf8, Mesp1, Fst, Smad1, Otx2
<input checked="" type="checkbox"/> primitive endoderm	level_detected , level_not_detected ,	Hhex, Wnt3, Fgf8, Smad2, Otx2, Gsc, Foxa2, Cer1, Sox17
<input checked="" type="checkbox"/> visceral endoderm	level_strong , level_detected ,	Smad1, Foxa2, Sox17
<input checked="" type="checkbox"/> extraembryonic component	level_weak , level_strong , level_detected ,	Smad5, Bmp4, Arc, Smad2

Figure 1: Genes, tissues and level of expression in Theiler Stage 9.

After the creation of the formal context, it is passed to the concept miner, which returns the number of formal concepts to the user. If the number of formal concepts is too high, the user can exclude from the computation concepts with fewer than a user-specified number of attributes and objects (so-called *minimum support*), to simplify the context further. Apart from the user being able to manually define minimum-support criteria, in CUBIST, the minimum-support feature is being reconfigured to be automatically calculated and applied to the formal context without user intervention.

4.2 Visual Analysis of Expression Clusters

Traditional software in FCA makes little use of visualisation techniques, producing poorly readable lattice graphs when the number of concepts exceeds a few dozen [5, 7]. To reduce the complexity of lattices, simplified diagrams can be produced by condensing or clustering concepts according to their intent frequency [8]. Visualisations can also be restricted to portions of the data [7], and concept size reduction is possible by incorporating conditions into the data mining process [17]. Finally, conceptual measures can be applied to identify the most relevant concepts and filter outliers [13].

CUBIST uses visual analytics techniques to allow users to interactively analyse the conceptual data, by filtering and

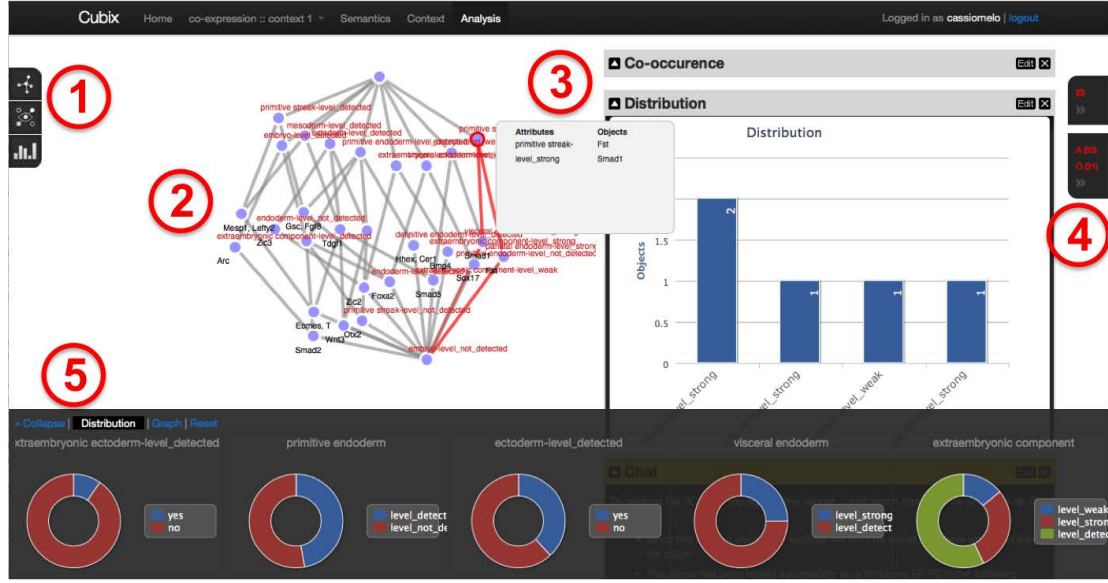


Figure 2: CUBIST user interface displaying the concept lattice for genes, tissues and level of expression in Theiler Stage 9 . Its main components: 1) Toolbar; 2) Visualisation canvas; 3) Dashboard; 4) Selection & entities bar and; 5) Filter bar.

selecting, transforming and clustering concepts. Figure 2 - 2 displays the selected visualisation for the concept lattice. Other visualisation options are: *Hasse* diagram (layered graph), matrix (objects \times attributes), sunburst (nodes as concentric arcs), and tree (no edges crossings). A filter bar (Figure 2 - 5) has two functions: first it allows the filtering of concepts through the visual selection of attributes; second, it displays the current conceptual distribution for each attribute. It is possible to perform textual search on the genes or tissues names for instance, with an auto-completion feature to help users easily locate the entity they are looking for. The results are dynamically highlighted as the user searches in the different concept nodes.

In addition to the main concept lattice visualisation, several charts display different aspects of the underlying conceptual structure such as co-occurrence of attributes, concepts distribution, stability vs. support, etc. (Figure 2 - 3). Some charts are updated when the user points the mouse over a concept, highlighting details of the concept. Similarly, a selection of a point/series in the chart will highlight the concerned concepts in the lattice. This technique is called *Linking and Brushing*.

Clustering of concepts (as opposed to clustering of objects or attributes) can be useful to facilitate the browsing of concepts and to identify zones of interest in the gene expression data. In our experiment (Figure 3), we used a K-means clustering algorithm to identify clusters of gene expression information. Some similarity measures are based on the concept lattice topology (e.g. counting the number of links between two concepts); Intent/extent similarity (e.g. Jaccard for the extent); or confidence between two pairs of concepts. In this case, we used similarity measures proposed by Boutari *et al.* [6].

Concept similarity (Jaccard). It is a coefficient for calculating the ratio of shared attributes between concepts. We define concept similarity as:

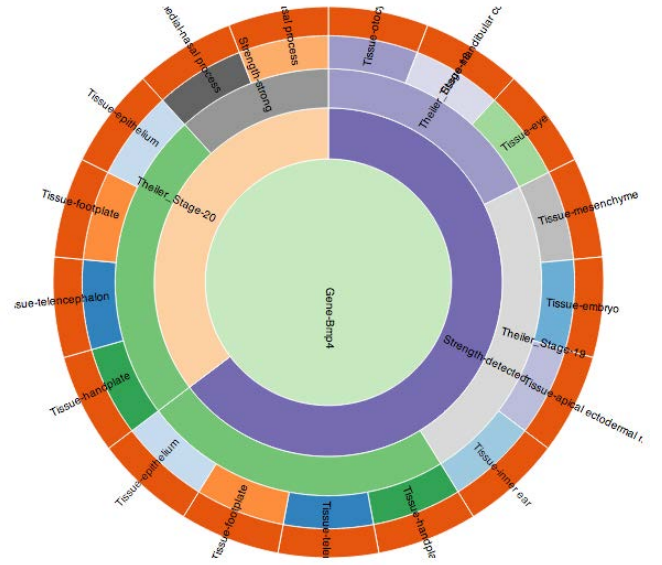


Figure 3: A radial-filling “Sunburst” visualisation of the gene expression data with colours depicting clusters.

$$CSim(A, B) = \frac{|m_a \cap m_b|}{|m_a| + |m_b|} + \frac{|g_a \cap g_b|}{|g_a| + |g_b|} \quad (1)$$

Proximity. Conceptual proximity is the topological distance between concepts A and B in the concept lattice.

$$prox(A, B) = 1 - \frac{shortestDistance(A, B)}{diameter(Lattice)} \quad (2)$$

Strength. It is the average concept similarity value (CSim) along the shortest path between a pair of concepts.

Figure 3 visualises the concept lattice from the context shown in Figure 1. Each concept is an arc and the hierarchy is represented from the innermost to the outermost layer. Colours show how concepts are distributed in the clusters, which facilitates the identification of zones of interest (e.g. concepts related to genes expressed in tissues of the same organ).

4.3 Dealing with Noise and Inconsistencies

Biological data are naturally inconsistent and incomplete. This is due to the sheer complexity of the subject matter; something as simple as a 1° degree change in temperature can cause otherwise identical experiments to provide different outcomes. Often experiments are repeated in order to confirm the original conclusion, yet this creates the potential for conflicting results, e.g.

bmp4 - strong - epiblast TS8
bmp4 - not detected - epiblast TS8

These two annotations (from two different experiments) deal with the same gene and the same structure (at the same point in time). Ideally, they would have the same level of expression too. Yet, this is not the case: the first annotation suggests the gene is expressed, whilst the second says it is not. To address inconsistencies of this type, CUBIST emphasizes the conflict by utilizing distinct colours according to the inconsistency type: binary (expressed versus not expressed) and analogue (e.g. strong expression is distinct from weak expression despite both levels suggesting a gene is expressed).

Another cause of inconsistency is the propagation of gene expression information. The mouse anatomy is a partonomy, with each tissue being part of another, e.g., the brain is part of the head. If a gene is expressed in the brain, it must be expressed in the head, i.e., positive expression is propagated up the anatomy. In contrast, if a gene is not expressed in the head, then it cannot be expressed in any sub-component of the head. Accordingly, the gene is not expressed in the brain, i.e., negative expression is propagated down the anatomy. In CUBIST, this information is used to resolve propagation at run time whenever a user queries the EMAGE data. Once again, detected inconsistencies are brought to the user's attention.

CUBIST also looks for flip-flops. This occurs when a gene is expressed in stages t and $t + 2$ but not in $t + 1$. For example, a gene is shown to be expressed in the dorsal mesentery in stages 16 and 18, but is not expressed in TS17. This flip-flop may be a sign that the annotation for TS17 is either missing or incorrect. To tackle this, we created a background process that is triggered whenever the ontology data is updated, checking for flip-flops and fixing (adding) missing annotations. The added instances are marked with an "autofix" label and stored back in the repository.

Another technique for approximating concepts in CUBIST, apart from the aforementioned minimum-support, is fault tolerance [14]. By applying fault tolerance to a formal context, missing data can be inferred, i.e., missing crosses in the formal context can be assumed to exist. If, for example, all missing crosses in a formal context are filled, the formal context can be approximated to a single formal concept. A

practical application of minimum-support and fault tolerance is evidenced in [2], where a large co-expression of genes in the skull bones of a mouse embryo was discovered. This was possible by mining the EMAGE data for large formal concepts. Disjoint groups of large concepts were then identified and fault tolerance applied to each disjoint group to produce a single very large co-expression from each group.

4.4 Using Association Rules to Highlight Gene Expression Occurrence Patterns

Association Rules (AR) are under the form premise \implies conclusion: $m_1 \text{ AND } m_2 \text{ AND } \dots m_n \implies n_1 \text{ AND } n_2 \text{ AND } \dots n_n$ for $m, n \in M$ and can be used to extract biological knowledge [16]. However, ARs carry very little information about how they can be visualized. They are typically displayed as a list of logical sentences, unpractical to analyse when the number of rules is large.

CUBIST currently provides two visualisations for association rules, combined with statistics and charts to enable progressive exploration of the ruleset: A matrix view, where each rule is displayed in a row and the concerned pairs of attribute-value in columns, with purple cells representing the premise and yellow ones the conclusion. The second visualisation is a radial graph showing how pairs attribute-value implies to each other in a radial graph layout. A scatterplot matrix in the dashboard shows the distribution of confidence, support and lift for the association rules, allowing users to graphically select portions of the distribution they are interested in.

To illustrate, after filtering the rules generated for the context in Figure 1 by confidence down to a manageable size, results revealed a few interesting facts. A significant amount (75%) of the genes detected in the embryo were also detected in the primitive endoderm (Figure 4) in TS9. This is not surprising, since the later tissue is part of the former in the anatomy hierarchy. On the other hand, another rule showed that 71% of the genes detected in the mesoderm were detected also in the ectoderm (against 62% the other way around) in TS9. This follows an intuitive reasoning since both tissues are part of the same organ.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we described an approach to the analysis of gene expression data where overlapping groupings are generated by Formal Concept Analysis and interactively analysed in an analytical tool called CUBIST. CUBIST workflow allows users to carry out an analysis starting from querying a semantic database, converting it into a formal context, simplifying the context to make it manageable, and visualising the result as a concept lattice and associated relevant statistics.

Existing tools for genes expression analysis, such as Cytoscape⁷ and Orange⁸ have specific advantages, e.g. Cytoscape can run efficient analysis in network data and Orange is a machine learning tool with some predictive features. In contrast, CUBIST operates at a conceptual level and it is less data-mining centric and more analytics oriented (i.e. dashboards, drill-down, selection and filtering, etc). Besides, those tools are designed to run locally whereas CUBIST is being designed to run on a cluster of comput-

⁷<http://www.cytoscape.org>

⁸<http://orange.biolab.si>

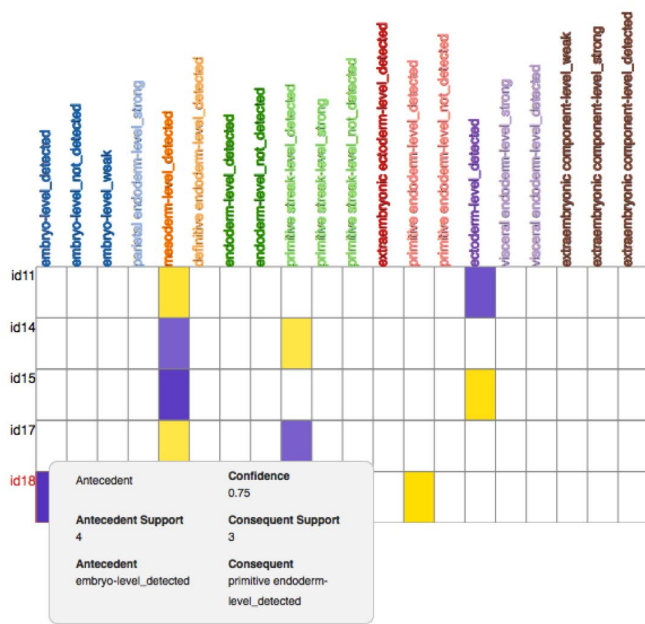


Figure 4: Genes, tissues and level of expression in Theiler Stage 9.

ers eventually in the cloud. This will allow scalable analysis of data of orders of magnitude higher than the mentioned tools.

Although most of the functionalities in CUBIST can be used with other data than EMAGE (with the corresponding scaling of data), as future work we will extend our experiments to other genes expression data sets like cancer and brain development. We also intend to provide a public web service API to allow interoperability with other platforms.

Acknowledgments

Funding was provided by the EU project CUBIST (FP7-ICT-2010-257403).

References

- [1] E. Akand, M. Bain, and M. Temple. A Visual Analytics Approach to Augmenting Formal Concepts with Relational Background Knowledge in a Biological Domain. Dec. 2010.
- [2] S. Andrews. In-close2, a high performance formal concept miner. In *Proceedings of the 19th international conference on Conceptual structures for discovering knowledge*, ICCS'11, pages 50–62, Berlin, Heidelberg, 2011. Springer-Verlag.
- [3] S. Andrews and K. McLeod. Gene co-expression in mouse embryo tissues. In *Proceedings of the 1st Combining and Uniting Business Intelligence with Semantic Technologies workshop*, volume 753, 2011.
- [4] E. Z. Antezana San Roman, M. Kuiper, and V. Mironov. Biological knowledge management: the emerging role of the semantic web technologies. *Briefings in Bioinformatics*, 10(4):392–407, 2009.
- [5] S. Blachon, R. Pensa, J. Besson, C. Robardet, and J.-F. Boulicaut. Clustering formal concepts to discover biologically relevant knowledge from gene expression data. *In Silico Biology*, 7(0033), July 2007.
- [6] A. M. Boutari, C. Carpineto, and R. Nicolussi. Evaluating term concept association measures for short text expansion: two case studies of classification and clustering. *7th International Conference on Concept Lattices and their Applications (CLA 2010)*, 2010.
- [7] Ducrou, P. J., Eklund, and T. Wilson. An intelligent user interface for browsing and searching mpeg-7 images using concept lattices. In *CLA 2006*, volume 4923, pages 1–21. Springer-Verlag Berlin Heidelberg, 2008.
- [8] S. Gerd, R. Taouil, Y. Bastide, N. Pasquier, and L. Lakhal. Computing iceberg concept lattices with titanic. *Data & Knowledge Engineering*, 42:189–222, August 2002.
- [9] R. Godin, R. Missaoui, and A. April. Experimental comparison of navigation in a galois lattice with conventional information retrieval methods. *International Journal of Man-machine Studies*, 38:747–767, 1998.
- [10] M. Kaytoug-Uberall, S. Duplessis, and A. Napoli. Using formal concept analysis for the extraction of groups of co-expressed genes. In *MCO'08*, pages 439–449, 2008.
- [11] D. A. Keim, F. Mansmann, J. Schneidewind, H. Ziegler, and J. Thomas. Visual analytics: Scope and challenges. December 2008. Visual Data Mining: Theory, Techniques and Tools for Visual Analytics, Springer, Lecture Notes In Computer Science (Incs).
- [12] C. Melo, B. Le-Grand, M. Aufaure, and A. Bezerianos. Extracting and visualising tree-like structures from concept lattices. In *15th International Conference on Information Visualisation*, pages 261–266, july 2011.
- [13] M.-A. A. Michel Soto, Benedicte Le Grand. Spatial visualisation of conceptual data. *International Conference Information Visualisation*, pages 57–61, 2009.
- [14] R. Pensa and J.-F. Boulicaut. Towards fault-tolerant formal concept analysis. In *Proc. 9th Congress of the Italian Association for Artificial Intelligence AI*IA'05*, LNAI, pages 212–223. Springer, Sept. 2005.
- [15] A. Prelić, S. Bleuler, P. Zimmermann, A. Wille, P. Bühlmann, W. Gruissem, L. Hennig, L. Thiele, and E. Zitzler. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, 22(9):1122–1129, May 2006.
- [16] P. C. Saez, M. Chagoyen, A. Rodriguez, O. Trelles, J. Carazo, and A. P. Montano. Integrated analysis of gene expression by association rules discovery. *BMC Bioinformatics*, 7(1):54+, 2006.
- [17] M. Zaki and C.-J. Hsiao. Efficient algorithms for mining closed itemsets and their lattice structure. In *IEEE Transactions on Knowledge and Data Mining*, volume 17. IEE Computer Soc., 2005.

Discovering Knowledge in Data using Formal Concept Analysis

Simon Andrews and Constantinos Orphanides

Conceptual Structures Research Group
Communication and Computing Research Centre
Faculty of Arts, Computing, Engineering and Sciences
Sheffield Hallam University, Sheffield, UK
{s.andrews,c.orphanides}@shu.ac.uk

Abstract. Formal Concept Analysis (FCA) has been successfully applied to data in a number of problem domains. However, its use has tended to be on an ad hoc, bespoke basis, relying on FCA experts working closely with domain experts and requiring the production of specialised FCA software for the data analysis. The availability of generalised tools and techniques, that might allow FCA to be applied to data more widely, is limited. Two important issues provide barriers: raw data is not normally in a form suitable for FCA and requires undergoing a process of transformation to make it suitable, and even when converted into a suitable form for FCA, real data sets tend to produce a large number of results that can be difficult to manage and interpret.

This article describes how some open-source tools and techniques have been developed and used to address these issues and make FCA more widely available and applicable. Three examples of real data sets, and real problems related to them, are used to illustrate the application of the tools and techniques and demonstrate how FCA can be used as a semantic technology to discover knowledge. Furthermore, it is shown how these tools and techniques enable FCA to deliver a visual and intuitive means of mining large data sets for association and implication rules that complements the semantic analysis. In fact, it transpires that FCA reveals hidden meaning in data that can then be examined in more detail using an FCA approach to traditional data mining methods.

1 Introduction

Formal Concept Analysis (FCA) is a term introduced by Rudolf Wille in 1984, building on applied lattice and order theory developed by Birkhoff and others in the 1930's. It was initially developed as a subsection of Applied Mathematics, based on the mathematisation of *concepts* and *concepts hierarchy* (Wille 2005). These concepts and concepts hierarchy form an intuitive visualisation called a formal concept lattice that provides insight into the relationships between objects and attributes. FCA is carried out on a set of binary relations between objects and attributes, usually in the form of a cross-table called a formal context.

Airlines	Latin America	Europe	Canada	Asia Pacific	Middle East	Africa	Mexico	Caribbean	USA
Air Canada	×	×	×	×	×		×	×	×
Air New Zealand		×		×					×
Nippon Airways		×		×					×
Ansett Australia				×					
Austrian Airlines		×	×	×	×	×			×

The cross-table above shows a formal context representing destinations for five airlines. The elements on the left side are formal objects; the elements at the top are formal attributes. If an object has a specific property (formal attribute), this is indicated by placing a cross in the corresponding cell of the table. An empty cell indicates that the corresponding object does not have the corresponding attribute. In the Airlines context above, Air New Zealand flies to Europe but does not fly to the Caribbean.

A formal context is defined as a triple $\mathbb{K} := (G, M, I)$, with G being a set of objects, M a set of attributes and I a binary (True/False) relation defined between G and M . The relation I is understood to be a subset of the cross product between the sets it relates, so $I \subseteq G \times M$. If an object g has an attribute m , then $g \in G$ relates to m by I , so we write $(g, m) \in I$, or gIm . For a subset of objects $A \subseteq G$, a derivation operator $'$ is defined to obtain the set of attributes, common to the objects in A , as follows:

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

In the same way, for a subset of attributes $B \subseteq M$, the derivation operator $'$ is defined to obtain the set of objects, common to the attributes in B , as follows:

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

In FCA, a *formal concept* is constituted by its *extension*, which comprises of all objects belonging to the concept and its *intension*, comprising of all attributes which are shared by all objects of its extension. A pair (A, B) is a formal concept in a given formal context (G, M, I) only if $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. The set A is the extent of the concept and the set B is the intent of the concept.

A formal concept is, therefore, a closed set of object/attribute relations, in that its extension contains all objects that have the attributes in its intension, and the intension contains all attributes shared by the objects in its extension. In the Airlines example, it can be seen from the cross-table that Air Canada, Nippon Airways and Austrian Airlines all fly to Europe and Asia Pacific. However, this

does not constitute a formal concept because all three airlines also fly to USA. Adding this destination completes (closes) the formal concept:

$(\{Air\ Canada, Nippon\ Airways, Austrian\ Airlines\}, \{Europe, Asia\ Pacific, USA\})$.

Another central notion in FCA is a duality called a *Galois connection*, often observed between items that relate to each other in a given domain, such as objects and their attributes. A Galois connection implies that “if one makes the sets of one type larger, they correspond to smaller sets of the other type, and vice versa” (Priss 2006). Using the formal concept above as an example, adding Mexico to the list of destinations reduces the set of airlines to $\{Air\ Canada\}$.

The Galois connections between the formal concepts can be visualized in a *concept lattice* (Figure 1), providing an intuitive way of portraying the concepts and concepts hierarchy.

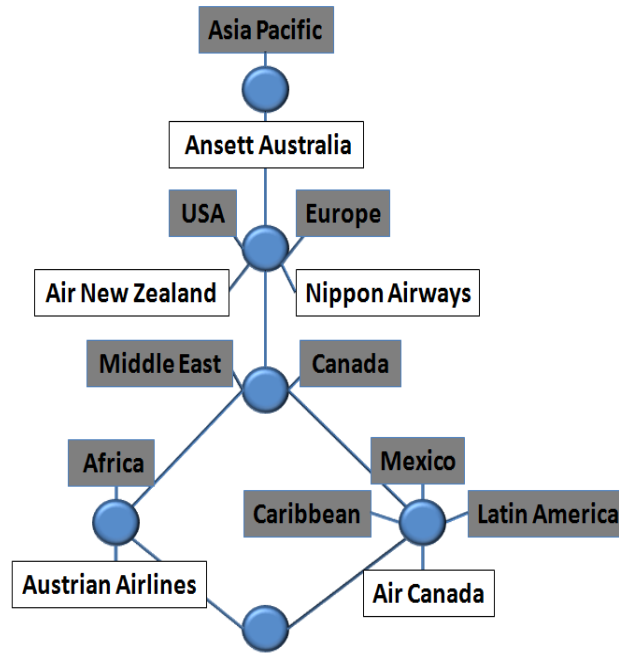


Fig. 1. A concept lattice corresponding to the Airlines context (after (Andrews et al. 2011))

A concept lattice consists of the set of concepts of a formal context and their subconcept-superconcept relations (Priss 2006). Each node in Figure 1 is a formal concept. The formal objects are noted slightly below and the formal attributes slightly above the nodes, which they label.

Extracting information from a lattice is a straightforward process which can provide valuable information. As an example, in Figure 1, we refer to the node labeled with the formal attribute ‘Africa’, as *Concept A*. In order to retrieve the extension of Concept A (the objects having the attribute ‘Africa’), one begins from the node where the attribute is labeled and traces all paths which lead *down* from the node; any objects one meets along the way are objects having that particular attribute. Tracing all paths leading down from the node labeled ‘Africa’, one will collect the object ‘Austrian Airlines’. Thus, Concept A can be interpreted as ‘Africa is flown to by Austrian Airlines only’. Similarly, the node labeled with the formal object ‘Austrian Airlines’ shall be referred to as *Concept B*. To retrieve the intension of Concept B (the attributes possessed by ‘Austrian Airlines’), one begins from the node where the object is labeled and traces all paths which lead *up* from the node; any attributes one meets along the way are attributes possessed by that particular object. Tracing all paths leading up from the node labeled ‘Austrian Airlines’, one will collect the attributes ‘Africa’, ‘Middle East’, ‘Canada’, ‘USA’, ‘Europe’ and ‘Asia Pacific’. Thus, Concept B can be interpreted as ‘Austrian Airlines flies to Africa, the Middle East, Canada, USA, Europe and Asia Pacific’.

2 FCA of data sets

Converting datasets into formal contexts is possible through a process of data discretization and data booleanization called *scaling* (Ganter and Wille 1989, 1998; Priss 2006; Andrews 2009a). Nominal, many-valued, attributes are converted into as many boolean attributes as they have values and continuous/ordinal attributes are hierarchically scaled or discretized as disjoint ranges of values. These processes are key to conducting FCA on data sets, and have been successfully applied, in a bespoke manner, to data in a number of problem domains, including crime detection (Poelmans et al. 2010, 2011), classification (Eklund and Wray 2010) linguistics (Falk et al. 2010), and gene expression (Kaytoue-Uberall, Duplessis and Napoli 2008). However, important issues provide barriers towards wider, more general, applicability and adoption of FCA. Firstly, the process of transformation described above is complex and time consuming, and requires a programmatic approach, and secondly, datasets of only a modest size can produce formal contexts containing hundreds of thousands of formal concepts (Krajca, Outrata and Vychodil 2009; Andrews et al. 2011), making the resulting visualisations unreadable and unmanageable. The most general and mature software in this area for FCA is the ToscanaJ suite (Becker 2005), which provides an interface to database tables that allows the user to visualise such data in formal concept lattices. ToscanaJ deals with complexity primarily by scaling individual table columns and then using nested lattices when a second column is scaled in the same analysis. Although ToscanaJ is the most general and integrated of the FCA data analysis tools available, it does not provide a means of dealing easily with large scale data analysis or the ability to scale multiple columns in the same concept lattice.

This article presents and describes some open-source FCA tools and techniques that have been developed and used to deal with data in rather different ways to ToscanaJ: *FcaBedrock*, a formal context creator developed by the authors that allows the user to query data tables in various ways, scaling data values and combining columns of data in a number of ways, and *In-Close*, a high performance formal concept miner and context simplifier, also developed by the authors, that allows complex results to be managed and approximated and provides the computing power to deal with large scale data analysis.

In combination with another open-source tool, Concept Explorer (Yevtushenko 2000), three examples of real data sets and real problems related to them, will be used to demonstrate the application of the tools and techniques. *FcaBedrock* will convert data into a formal context and *In-Close* will simplify the context to make the resulting concept lattice manageable and readable. Whilst this process of simplification is necessarily at the expense of some detail, the results obtained are still meaningful and provide a focus for further, more detailed analysis. This further analysis can also be provided by FCA, using the same tools to produce lattices that provide a visual approach to traditional data mining techniques such as the identification of implication and association rules.

2.1 FCA Scaling

In terms of data, formal contexts are boolean data. While data predominantly exist in non-boolean forms, FCA introduces non-boolean attributes via many-valued contexts, using *conceptual scaling* to transform them into single-valued contexts. Each non-boolean attribute is given a *scale*, each scale being a context in itself.

In (Wolff 1993), an example is given of a table containing the sex and age of eight persons. In order to convert this many-valued context into a single-valued formal context, a scale was created for each of the many-valued attributes; one for sex and one for age. The first scale contains the two possible values of sex, ‘m’ for male and ‘f’ for female. Age, however, has potentially many values, but is an ordinal attribute, so rather than using individual values, a scale was created using boundary values: ‘<18’, ‘<40’, ‘<=65’, ‘>65’ and ‘>=80’. ‘ADAM’ is 21 years old, so has the formal attributes ‘<40’ and ‘<=65’. ‘GEORGE’ is 90 years old, so has the formal attributes ‘>65’ and ‘>=80’. The two scales were then merged to form the complete formal context representing ‘the gender and age of eight persons’ in Figure 2.

Figure 3 shows the formal context in Figure 2 as a concept lattice, visualised in an open-source FCA tool called *Concept Explorer* (ConExp) (ConExp 2011). It is easily seen from the lattice, for example, that there are two people, Dora and George, aged 80 or above, there are three females: Dora, Betty and Eva, and Dora is the only female aged 80 or above. The Shaded upper and lower semi-circles in a concept denote that there are attributes (upper) and objects (lower) labeled at that concept.

The process of scaling can be automated, although it is clear that certain decisions have to be made, such as where to place boundary values. Whilst such

	sex	age
ADAM	m	21
BETTY	f	50
CHRIS	?	66
DORA	f	88
EVA	f	17
FRED	m	?
GEORGE	m	90
HARRY	m	60

	sex		age				
	m	f	<18	<40	<=65	>65	>=80
ADAM	X			X	X		
BETTY		X			X		
CHRIS						X	
DORA		X				X	X
EVA		X	X	X	X		
FRED	X						
GEORGE	X					X	X
HARRY	X				X		

Fig. 2. The transformation of data into formal contexts (after (Wolff 1993))

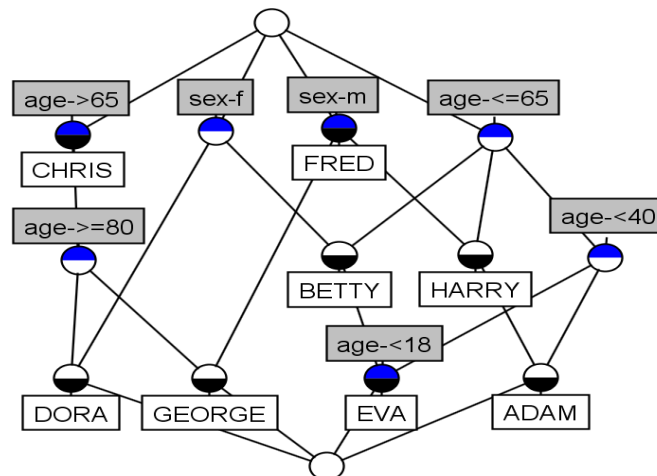


Fig. 3. A concept lattice corresponding to the formal context in Figure 2

automation has been carried out on a number of data sets on a bespoke basis, FcaBedrock is a tool that generalizes the approach.

2.2 FCA Tools

FcaBedrock FcaBedrock is a tool, developed by the authors (Andrews and Orphanides 2010a; FcaBedrock 2011), for creating formal contexts by converting many-valued attributes into formal attributes (Andrews and Orphanides 2010b; Andrews et al. 2011). By parsing a data file, the tool obtains the metadata required for conversion, such as attribute names and attribute types. If necessary, the user can guide the automated scaling by making decisions such as where to place boundary values. FcaBedrock allows the user to determine which many-valued attributes of the original data, and even which attribute values, should be included in a formal context, depending on the analysis that is being carried out. This type of data ‘query’ means that a sub-context of the data set can be created that may well result in a manageable concept lattice without the need for further simplification.

In-Close In-Close is a fast formal concept miner and formal context simplifier developed by the authors (Andrews 2009b; In-Close 2011) (Andrews 2011). The tool accepts a formal context in the well-known Burmeister .cxt format and computes its formal concepts. It allows the user to exclude from the computation concepts with fewer than a specified number of attributes and/or a specified number objects (so-called *minimum-support for intent* and *minimum-support for extent*). After computing the concepts, In-Close outputs another context, but with only those concepts that satisfy the specified minimum support. In this way, minimum support can be used to simplify (approximate) a formal context by excluding ‘smaller’ concepts, resulting in a readable concept lattice (Andrews and Orphanides 2010b; Andrews et al. 2011). This is a semi-automated form of a process known as lattice ‘iceberging’ (Stumme et al. 2001). Although detail will be lost in this process, the intension is to include concepts that have enough support to be more significant in the data, thereby resulting in a meaningful approximation of the concept lattice. Given that even modest-sized data sets will usually correspond to formal contexts with hundreds of thousands of formal concepts, a means of simplification is essential before a readable lattice can be visualised. Whereas a focused and well crafted data query using FcaBedrock can often lead to a readable concept lattice, a different approach is required if all of the data is being analysed. The query then is not so much about a specific attribute or value but more about “what is conceptually significant in this data?”. In other words, approximation through minimum support attempts to reveal the ‘large concepts’ in data.

Concept Explorer Concept Explorer (ConExp) is a concept lattice visualiser, implementing basic ideas needed for the study and research of FCA (Yevtushenko 2000; ConExp 2011). It takes a formal context in the Burmeister format and

computes the formal concepts and corresponding Galois connections to construct and visualise a concept lattice. It provides features such as formal context editing, attribute exploration and other analysis aids for FCA. FcaBedrock and In-Close can be used to create formal contexts from data files that result in readable concept lattices in ConExp.

3 Dataset Analyses

FcaBedrock, In-Close and Concept Explorer were used to analyse three sets of data:

- *Market Intelligence Data*, to explore job vacancies in London, UK, and investigate the existence of a UK ‘North/South divide’ of career opportunities.
- *Mouse Embryo Gene Expression Data*, to find large gene co-expressions in the mouse embryo.
- *Mushroom Data*, to find indicators for edible and poisonous mushrooms.

3.1 Market Intelligence Data

The specific dataset consists of job vacancies advertised on the United Kingdom’s leading web-based job boards, as well as employers’ own websites, tracked in real-time using the proprietary software of a market intelligence company called Innovantage. An initial analysis of this dataset was carried out as part of the CUBIST project (Orphanides 2011).

The dataset comprises of a sample of 400 jobs accompanied by their details:

- Title: The job’s title.
- Description: A brief description outlining the requirements of the job.
- Date Found: The exact time of when the job was tracked.
- URL: The website where the job was found at.
- Raw Location: The location of the employer.
- Raw Salary: The advertised salary, sometimes also including information about bonuses and benefits.

An example of a job entry is shown below (Figure 4).

This semi-structured data required some manual pre-processing to obtain many-valued tabular data suitable for FcaBedrock. Although FcaBedrock can scale a variety of data types for FCA, it is not capable of Natural Language Processing of free text. Taking the original ‘Raw Location’ attribute as an example, a job’s location recorded as “Manchester”, “Manchester, United Kingdom” or “Greater Manchester”, were all treated as the single value “Manchester”. Similarly, for ‘Raw Salary’ single numerical values were extracted from a variety of forms such as, “15000-20000”, “18000 GBP” and “25000 per annum, negotiable”. Job titles were categorised so that subsequent analyses could produce aggregated, generalised results. For example, a mechanical engineer and a civil engineer both fall into the ‘Engineer’ category.

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<jobs>
  <job>
    <title>Data Centre Developer</title>
    <description>Data centers Developer opportunity based in Amsterdam
on a 6 months rolling contract. This is a very senior position and
requires the candidate to have at least 7 years experience and
have extensive knowledge and expertise in the construction
of a data centers facilities including electrical systems,
cooling plants etc and familiar with EU regulations and
best practices.</description>
    <date_found>2011-01-12 17:01:58.0</date_found>
    <url>http://www.itjobspost.com/JobSeeker/</url>
    <raw_location>Amsterdam, Other Countries, UK</raw_location>
    <raw_salary>400-600 Per Day</raw_salary>
  </job>
</jobs>

```

Fig. 4. Example of a job entry.

London Jobs Let us say we are interested in what kind of jobs are advertised in London and how much they pay. FcaBedrock was used to create a formal context using the salary and job category attributes, whilst restricting the location to London. The resulting concept lattice is shown in Figure 5.

In these data, objects (job adverts) do not have individual names so the lattice shows the number of objects in the extent of each concept and the corresponding percentage that represents of the total number of objects in the context. In addition, the size of the concept node is now proportional to the size of the extent of the concept. The lattice can be studied for a number of results, acting rather like a visualisation of a complex data base query. For example,

- There are six engineering jobs, five of which have a salary between 25 and 40K. The other has a salary less than 25K. There are no engineering jobs with a salary greater than 40K.
- A high proportion of advertised London jobs (24 out of 85) are managerial.
- There are four executive jobs advertised, two of which have salaries $\geq 40K$. The other two salaries are not specified. A closer inspection of the original data reveals that the salary in both these cases is “negotiable”.
- Most of the vacancies are of a corporate professional type (e.g. project managers, business analysts, etc.), perhaps displaying the corporate nature of London as the UK’s financial and commercial centre.

The North/South divide It is traditional in the UK to think of a ‘North/South’ divide in economic and demographic terms, with the South being considered as the more prosperous. There is a saying in the UK that it is “grim up North”. To investigate this hypothesis a second analysis was carried out, by creating a

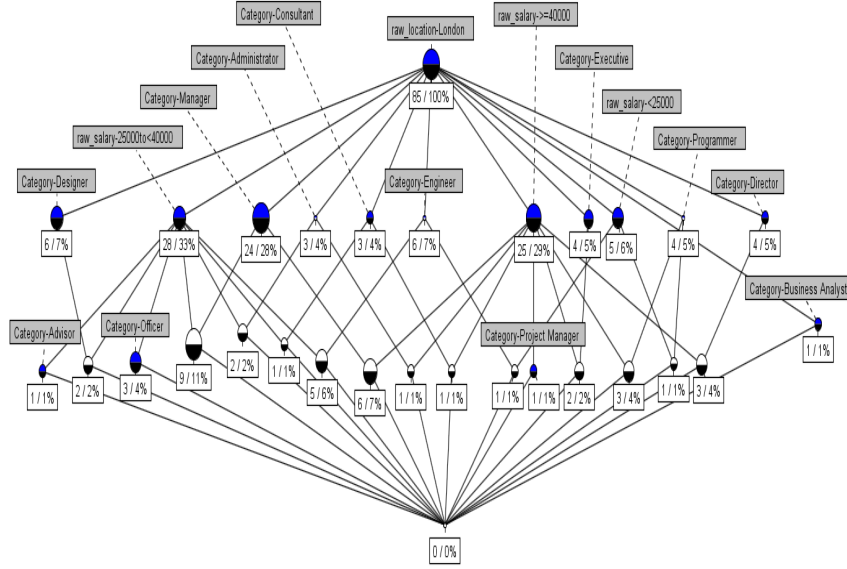


Fig. 5. Visualising the resulting formal context in ConExp

new attribute to classify locations as either North or South in the UK. There is a traditional, informal, dividing line between the North and South in the UK at the Watford Gap service station on the M1 motorway. Given the colloquial nature of the hypothesis in this analysis it was therefore requisite to use this for the North/South classification of the job locations.

FcaBedrock was used to create a formal context using the salary attribute progressively scaled against the newly created North/South attribute. The resulting concept lattice is shown in Figure 6.

In all, there were 220 jobs that had a numerical salary value and an identifiable North or South location. Of these, 124 were in the South and 76 were in the North. Some simple arithmetic is required to obtain some pertinent results for the analysis:

- For jobs in the South, there are 46 out of 124 (37%) with a salary <30K, compared to 51 out of 76 (67%) in the North.
- For jobs in the South, there are $(124-98)=26$ out of 124 (21%) with a salary $\geq 50K$, compared to $(76-70)=6$ out of 76 (8%) in the North.

Although this is a fairly small sample of job advertisements, it is still quite a convincing argument in favor of the hypothesis. According to the FCA, it is indeed “grim up North”.

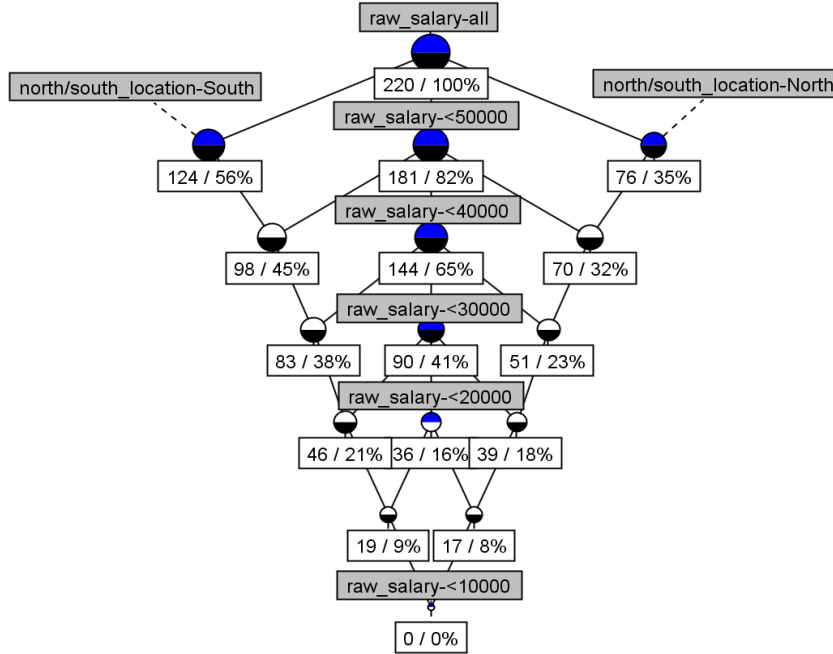


Fig. 6. The ‘North/South Divide’ visualised in ConExp

3.2 Mouse Embryo Gene Expression Data

This dataset originates from the EMAGE Edinburgh Mouse Atlas Project (EMAGE 2011; Richardson et al. 2010) and was produced in collaboration with Herriot-Watt University, Edinburgh, UK. The data is part of a use-case in the European CUBIST project (Combining and Uniting Business Technologies with Semantic Technologies) (CUBIST 2011). As part of this project, an initial analysis of this dataset was carried out for the 1st CUBIST workshop in Derby, UK, 2011 (Andrews and McLeod 2011). The analysis presented here is a development of that work.

The dataset consists of mouse gene expression data for 6,838 genes in 2,335 mouse tissues. Genes are named using coded abbreviations, e.g., ‘Mapk8ip2’, ‘Tgfb1’, ‘Zcchc6’. There are seven levels of strength of gene expression in the tissues, ranging from ‘not detected’ to ‘strong’. A formal context was created from this data set using FcaBedrock by considering a gene as a formal object and a combination of tissue with level of expression as a formal attribute. In the context, In-Close computed 208,378 concepts, each representing a gene co-expression, i.e., a set of tissues associated with a set of shared gene expressions. Gene coexpressions are of interest to biologists because information regarding gene coexpression is useful to predict gene function; the function of every gene

depends on that of another gene or genes, and networks of genes are usually responsible for any particular biological function or process.

<pre> **** In-Close 3.0 Concept Miner **** Enter cxt file name including extension: emage.cxt Enter minimum size of intent (no. attributes): 0 Enter minimum size of extent (no. objects): 0 Reading data... Sorting... Mining concepts... Sort time : 0.178463 seconds Inner time : 1.97868 seconds Inner+Sort time : 2.15714 seconds Total time : 2.70412 seconds Number of concepts: 208378 Output concepts to file? (y/n): n Hit <enter> to finish </pre>	<pre> **** In-Close 3.0 Concept Miner **** Enter cxt file name including extension: emage.cxt Enter minimum size of intent (no. attributes): 12 Enter minimum size of extent (no. objects): 23 Reading data... Sorting... Mining concepts... Sort time : 0.168868 seconds Inner time : 1.74649 seconds Inner+Sort time : 1.91536 seconds Total time : 2.46796 seconds Number of concepts: 22 Output concepts to file? (y/n): n Outputting context file... Hit <enter> to finish </pre>
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Fig. 7. Computing the concepts in the EMAGE data - all concepts (left), and using minimum support (right)

By specifying a minimum support of 23 genes and 12 tissue/levels (through a process of trial and error) the number of concepts was reduced to 22 (see Figure 7). In other words, ‘large’ gene co-expressions were computed that involve at least 23 genes and 12 tissues. On inspecting the simplified context it also became clear that these large co-expressions formed two disjoint groups. One of those groups is shown as a formal context in Figure 8.

The co-expressions discovered in this analysis are showing large groups of genes that are strongly expressed in a number of bone structures. It is beyond the scope of this paper to investigate the biological significance of the groupings of the particular bone structures involved or the biological processes that the particular genes are involved in. However, having discovered these large co-expressions, domain experts in the relevant fields of biology may be interested in investigating how these genes may be working together and what biological processes may be associated with these collective bone structures. It is noted, for example, that the tissues are either bones in the skull or in the limbs and that they are all from the same development stage of the mouse embryo (*Theiler Stage 23*). Further biological investigation may determine the significance of this co-expression, particularly as to what happens in the development of the mouse skull and limb bones at Theiler Stage 23.

Gene Co-Exp	orbito-sphenoid-strong	axial skeleton-strong	femur-strong	humerus-strong	basisphenoid bone-strong	basioccipital bone-strong	turbinate bones-strong	scapula-strong	tibia-strong	fibula-strong	otic capsule-strong	petrous part-strong	pelvic girdle-strong
Mapk8ip2	x	x	x	x	x	x	x	x	x	x	x	x	x
Tgfb1	x	x	x	x	x	x	x	x	x	x	x	x	x
Zcchc6	x	x	x	x	x	x	x	x	x	x	x	x	x
Brpf3	x	x	x	x	x	x	x	x	x	x	x	x	x
Caly	x	x	x	x	x	x	x	x	x	x	x	x	x
Bcl9l	x	x	x	x	x	x	x	x	x	x	x	x	x
Zc3h18	x	x	x	x	x	x	x	x	x	x	x	x	x
Dnajc18	x	x	x	x	x	x	x	x	x	x	x		
H2-T22	x	x	x	x	x	x	x	x	x	x	x	x	x
Colec12	x	x	x	x	x	x	x	x	x	x	x	x	x
Wwp2	x	x	x	x	x	x		x	x	x	x	x	x
Emp3	x	x	x	x	x	x	x	x	x	x	x	x	x
Tcam1	x	x	x	x	x	x	x	x	x	x	x	x	x
Papss2	x	x	x	x	x	x	x	x	x	x		x	x
Ubxn10	x	x	x	x	x	x	x	x	x	x	x	x	x
Cytl1	x	x	x	x	x	x	x	x	x	x	x	x	x
BC024814	x	x	x	x	x	x	x	x	x	x	x	x	x
Plekhh1	x	x	x	x	x	x	x	x	x	x	x	x	x
Apitd1	x	x	x	x	x	x	x	x	x	x	x	x	x
Cebpz	x	x	x	x	x	x	x	x	x	x	x	x	x
1110017D15Rik	x	x	x	x	x	x	x	x	x	x	x	x	x
Copb1	x	x	x	x	x	x		x	x	x	x	x	x
Unc5cl	x	x	x	x	x	x	x	x	x	x	x	x	x
Haus4	x	x	x	x	x	x	x	x	x	x	x	x	x

Fig. 8. Simplified context of gene expression in mouse tissues

The resulting concept lattice created by ConExp is shown on the left in Figure 9. The visualisation clearly shows that all twenty-four genes in the context are strongly expressed in nine of the tissues (the ones listed at the top of the lattice). The lattice on the right is highlighting the concept labeled by ‘pelvic girdle-strong’. It shows that twenty-three of the genes are expressed strongly in the pelvic girdle, but not gene Dnajc18 - in terms of Galois connections in the data, adding the tissue ‘pelvic girdle’ to the intent, removes ‘Dnajc18’ from the extent.

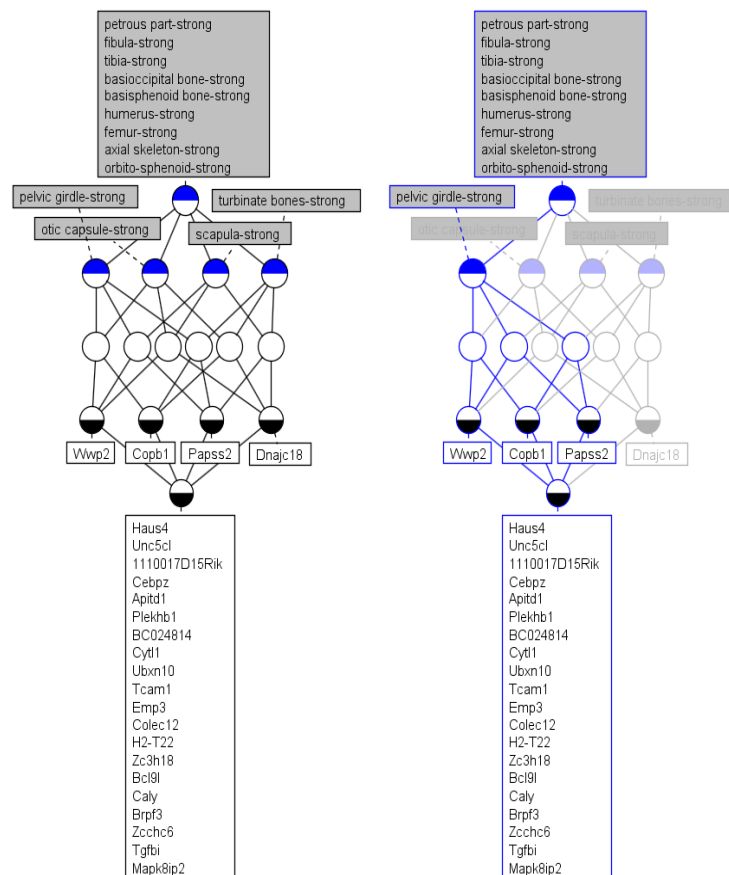


Fig. 9. The concept lattice of gene co-expression. The left version highlights the concept labeled with ‘pelvic girdle-strong’

It is therefore interesting to consider the empty cells in the context. Why is Dnajc18 ‘missing’ from ‘pelvic girdle-strong’? It is possible to investigate these

missing relationships in a traditional manner by querying the original data. For example, by querying the data for expressions involving ‘Dnajc18’, it transpires that this gene is expressed at a ‘moderate’ level in the pelvic girdle. However, the biological assays undertaken to obtain this data involve staining slices of mouse embryos for a particular gene and inspecting the results visually to make a judgment as to the level of gene expression. Clearly there is a subjective element to this assay that may even vary from one biologist to another, so for the purposes of this FCA it may be acceptable to add the relationship (pelvic girdle-strong, Dnajc18) to the context. It also transpires from the original data that an assay for gene ‘Wwp2’ has not yet been carried out in the turbinate bones. It may be that we can infer from the FCA that the likely result of such an assay will be a strong level of expression. This idea of ‘fault tolerance’ in FCA (Pensa and Boulicaut 2005) can be applied to a formal context, so that where there are small omissions in otherwise complete closures a sensible approximation to a complete closure can be made. The result here, for example, could be simplified to a single concept of gene co-expression; all twenty-four genes being strongly expressed in all thirteen tissues.

It was also interesting to discover what bones are missing from the co-expression and for what reason. There are bones in the skull for which there were no data recorded for particular genes, but when an image of the embryo from the relevant experiment was subsequently examined, it was clear that these bones *did* have expression in them. This corroborates the possible use, suggested above, for this FCA technique, in inferring the most likely level of expression for a gene-tissue pair when data is missing.

3.3 Mushroom Data

This dataset is taken from the UCI Machine Learning repository (Frank and Asuncion 2011) and contains descriptions of hypothetical samples of several species of gilled mushrooms in the Agaricus and Lepiota family. The dataset contains data of 8124 edible and poisonous mushrooms, with 23 many-valued attributes describing their properties such as edibility, stalk shape, cap colour and habitat. The mushroom guide from which the records were taken (Lincoff 1981) clearly states that there is no simple rule for determining the edibility of a mushroom; no rule like “leaflets three, let it be” for Poisonous Oak and Ivy. Nevertheless, it might be interesting to see what guidelines can be discovered from the data using FCA, by computing the largest formal concepts that exist for edible and poisonous mushrooms. A strength of FCA is that combinations of attributes are highlighted. Although no single attribute is a good indicator of edibility, perhaps a combination of several attributes is.

FcaBedrock was used to scale the data, automatically converting the 23 original many-valued attributes into 125 formal attributes. Attribute restriction was applied to divide the data set into two sub-contexts, one containing only edible mushrooms (see Figure 10) and one containing only poisonous ones. In the edible mushroom sub-context there were 4,208 mushrooms (objects) and 92,543 concepts and in the poisonous mushroom sub-context there were 3,916 mushrooms

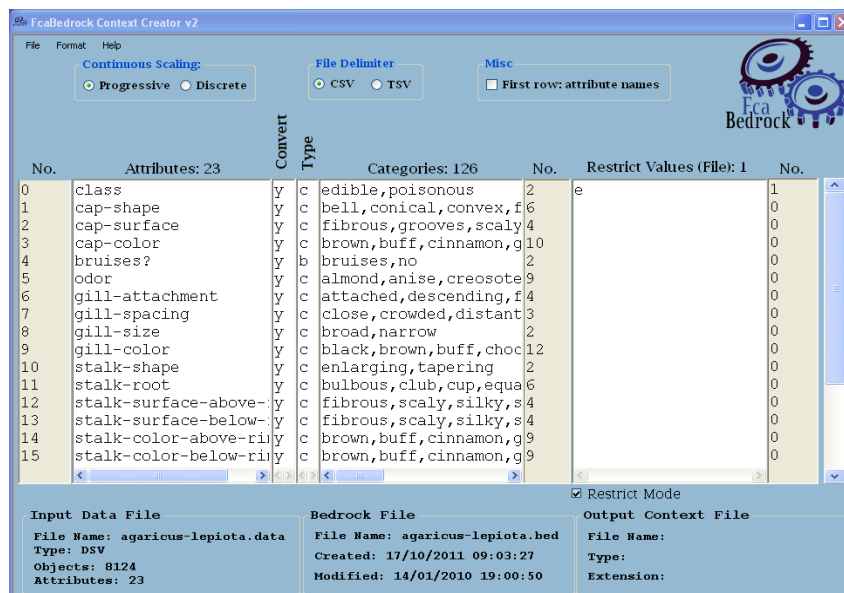


Fig. 10. Edible mushroom sub-context being created by FcaBedrock

and 86,198 concepts. By a process of trial and error, the minimum-support for mushroom attributes was set to ten using In-Close, to extract only relatively large concepts, that involve significant combinations of attributes. Thus, a small number of large co-factors of edibility and poisonousness were approximated from the data. This computation resulted in a simplified edible mushroom sub-context containing 2,848 mushrooms and 17 concepts, and a simplified poisonous mushroom sub-context containing 3,344 mushrooms in 14 concepts. It was apparent by comparing the corresponding lattices that a number of attributes were present in both, suggesting that these particular attributes do not have a strong bearing on the edibility of a mushroom. These common attributes were removed from each sub-context simply by unchecking them in a corresponding list in ConExp. The attributes remaining were thus particular to each sub-context and the corresponding lattices are shown in Figure 11.

The edible mushroom lattice suggests that the following are indicators of edible mushrooms: smooth stalk, no odor, pendant ring and having bruises. 62% of the 2848 edible mushrooms had all of these attributes. 87% had smooth stalks and bruises and all of them had a smooth stalk above the ring.

The poisonous mushroom lattice suggests that the following are indicators of poisonous mushrooms: silky stalk, foul odor, evanescent or large ring, white or chocolate coloured spore print, bulbous stalk root, narrow buff coloured gills and an enlarging stalk shape. 66% of the 3344 poisonous mushrooms had a silky

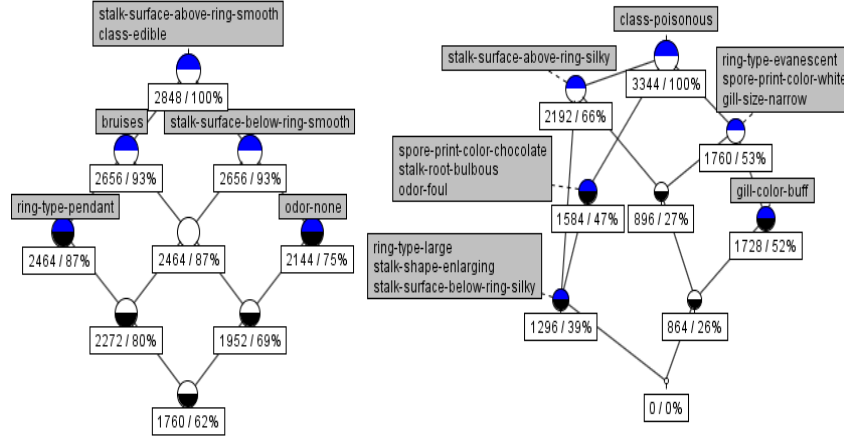


Fig. 11. Edible and poisonous mushroom concept lattices

stalk above the ring. Of all the attributes, the surface of the stalk seems to be the most significant, smooth indicating edible and silky indicating poisonous.

The poisonous mushroom lattice is also interesting in that it suggests two quite distinct types of poisonous mushroom, evidenced by two disjoint sets of attributes: *spore-print-color-chocolate*, *stalk-root-bulbous*, *odor-foul*, *ring-type-large*, *stalk-shape-enlarging* and *stalk-surface-below-ring-silky* on the left and *ring-type-evanescent*, *spore-print-color-white*, *gill-size-narrow* and *gill-color-buff* on the right.

Of course, these findings are approximations, having come from lattices derived using minimum support. Nevertheless, the notion of focusing on relatively large associations of objects and attributes seems to be a sensible one. And, once attributes of interest are indicated by these approximations, FCA and other techniques can be used to analyse these findings in more detail (see below).

4 Traditional DM methods using FCA

Traditional querying of data is evidenced in FCA by the lattices produced in section 3.2 concerning the market intelligence data. Using FcaBedrock to restrict the conversion of data into a formal concept by specifying particular attributes and attribute values is analogous to creating SQL queries, for example, that extract database information based on fields/tables and field/table values. Other methods of analysis such as association rule and implication rule mining also have counterparts in FCA. A key difference between the more traditional approaches and the FCA ones is that the results in FCA are naturally visualised in the Galois connections of the concept lattices.

4.1 Implications

Dependencies among attributes in a dataset can produce significant information. Facts of the form “attribute a exists for an object each time attribute b exists as well” may produce deep and hidden meaning about regularities of the domain which the data originate from (Valtchev, Missaoui and Godin 2004). Within this context, FCA offers *implication rules*, a “compact representation mode for this type of knowledge” (Valtchev, Missaoui and Godin 2004), following the same formalism as its logical counterpart, meaning that an implication is considered to be valid in a context if none of the objects violates the implication.

The scaling of ordinal numerical attributes in FCA gives some obvious examples of implication. In Figure 3, for example, the obvious implications:

$$age \geq 80 \Rightarrow age \geq 65$$

$$age < 18 \Rightarrow age < 40 \Rightarrow age \leq 65$$

are easily seen from the hierarchy of concepts.

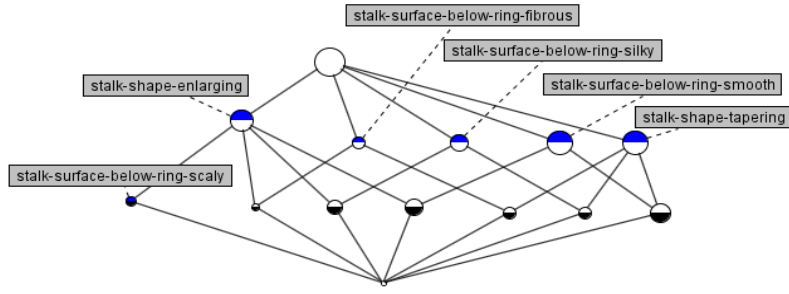


Fig. 12. A lattice showing dependencies between mushroom stalk shapes and surfaces

Figure 12 shows an example using the UCI Mushroom dataset, in which the conversion of the data was restricted to the *stalk-shape* and *stalk-surface-below-ring* attributes, to see if there were any dependencies between these attributes. It is quite clear from the lattice that:

$$stalk-surface-below-ring = scaly \Rightarrow stalk-shape = enlarging$$

In other word, all mushrooms (in the dataset) that have scaly stalks below their ring also have an enlarging stalk shape.

By considering the three concepts below the one labeled *stalk-shape-tapering* one can also discern that:

$$stalk-shape = tapering \Rightarrow stalk-surface-below-ring = smooth \vee$$

$$stalk-surface-below-ring = silky \vee stalk-surface-below-ring = fibrous$$

Implications from simplified formal contexts can also be made, but should be treated as *approximate implications*. The fact that a dependency exists in the simplified lattice does not preclude an object existing in the unsimplified context that violates that dependency. For example, in Figure 11:

$$\begin{aligned} ring-type = pendant &\Rightarrow bruises \Rightarrow \\ stalk-surface-above-ring = smooth \wedge class = edible \end{aligned}$$

However, it would be dangerous, for example, to say that having bruises implies that a mushroom is edible. The approximation suggest that bruises are significant, but a closer look is required to quantify this significance. In fact, a simple FCA data query using FcaBedrock determines that 82% of bruised mushrooms in the data set were edible, 18% poisonous and that bruised mushrooms comprised 41.6% of the sample.

4.2 Association rules

One of the benefits of FCA and particularly the concept lattice, is that the intuitive visualisation and the underlying conceptual hierarchy allow the generation of association rules directly from the lattice. The association between bruises and edibility in agaricus/lepiota mushrooms, mentioned in the previous section, is easy to see in a simple FCA lattice (Figure 13), and can be written:

$$bruises = [82\%] \Rightarrow class = edible$$

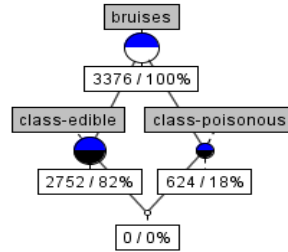


Fig. 13. Association between bruises and edibility

If all of the ‘edibility indicator’ attributes from the In-Close approximated lattice are combined, the association becomes 96% (Figure 14), a result that suggests the approximation was a good one; if an agaricus or lepiota mushroom has all of the indicators, then there is a very good chance that it is safe to eat.

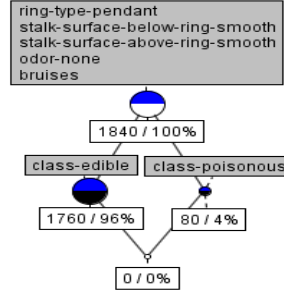


Fig. 14. Association between the combined ‘edibility indicators’ and edibility

More complex association rules can be calculated from the object counts of combinations of attributes: rules of the form “if object has attribute *a* and attribute *b*, it has an *X*% chance of also having attribute *c*”. For example, Figure 15 adds the attribute *stalk-root-bulbous* to the mushroom stalk lattice. The association between the attribute pair, *stalk-root-bulbous* and *stalk-shape-tapering* and the attribute *stalk-surface-below-ring-smooth*. The concept labelled *2112/26%* has the attributes *stalk-root-bulbous* and *stalk-shape-tapering*. The concept directly below, labelled *1968/24%*, adds *stalk-root-bulbous*. Thus, if a mushroom has a bulbous root and tapering stalk, there is a $1968/2112 = 93\%$ probability that it will also have a smooth stalk surface below its ring:

$$\text{stalk-root} = \text{bulbous} \wedge \text{stalk-shape} = \text{tapering} = [93\%] \Rightarrow \text{stalk-surface-below-ring} = \text{smooth}$$

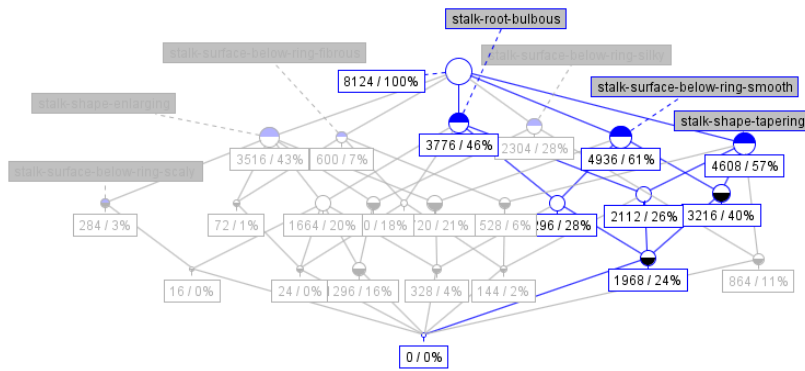


Fig. 15. An association between bulbous root and other stalk attributes

5 Conclusion and Further Work

This article has shown, through practical examples, how Formal Concept Analysis can be used as a knowledge discovery tool. It was shown that straightforward techniques and automated processes and tools allow FCA to be more widely and generally applied to data and data sets than has previously been the case. The intuitive concept lattice can be used to visualise meaningful approximations of complex results, discovering hidden knowledge that is not easily accessed by other means. This article has also shown how FCA can provide visual alternatives to traditional text-based queries and data mining methods.

Nevertheless, the tools and techniques presented here are prototypical and need further development and integration. In particular,

- An automated means of finding suitable levels of minimum support is required to replace the current trial and error process of context approximation.
- The current implementation of FcaBedrock is not ideal for large numbers of attributes. Beyond, say, 100 values for a many-valued attribute the current interface becomes rather unwieldy in navigating the values, and data sets with thousands of corresponding formal attributes begin to cause noticeable time lags in processing.
- Currently, only structured data is supported. In the case of unstructured or semi-structured data, such as the market intelligence data in section 3.1, bespoke pre-processing is required to extract structured data before scaling can commence.
- Although the tools described here are interoperable via the .cxt context file format, integration into a single tool would improve the availability and usability of the FCA techniques.

These issues are being addressed in a European Framework Seven project called CUBIST (Combining and Uniting Business Technologies with Semantic Technologies) (CUBIST 2011). In CUBIST, In-Close and Fcabedrock are being developed and integrated to sit on top of an RDF triple-store, along with a new concept lattice tool, to provide new FCA-based visual analytics. A Semantic Extract, Transform and Load (SETL) layer is being added to allow these analytics to be applied to structured and unstructured data. The SETL will provide the NLP required, for example, to process the market intelligence data described in section 3.1.

Once data are loaded as triples into the data store, an Fcabedrock interface will be used to query the triple-store using the SPARQL query language. From FcaBedrock the queried data will be scaled into a formal context, using an embedded In-Close function to approximate the context as, and if, required. FCA-based visualisations, such as those presented here, will then be used side-by-side with traditional BI and DM techniques to complement and corroborate each other in the analysis.

In the CUBIST vision, it is not a matter of which technology or method can produce the best results or discover the most knowledge, but rather how they

can best be combined and united to deliver a comprehensive, complementary and corroborative solution for data analysis.

Acknowledgments Significant parts of this work are part of the CUBIST project (“Combining and Uniting Business Intelligence with Semantic Technologies”), funded by the European Commission’s 7th Framework Programme of ICT, under topic 4.3: Intelligent Information Management.

Bibliography

- Andrews S (2011) In-Close2, a High Performance Formal Concept Miner. In: Andrews S, Polovina S, Hill R, Akhgar B. (eds) Proceedings of the 19th International Conference on Conceptual Structures (ICCS) 2011. LNAI 6828. Berlin: Springer-Verlag. pp. 50-62
- Andrews S (2009) Data Conversion and Interoperability for FCA. In: CS-TIW 2009, http://www.kde.cs.uni-kassel.de/ws/cs-tiw2009/proceedings_final_15July.pdf. pp. 42-49
- Andrews S (2009) In-Close, A Fast Algorithm for Computing Formal Concepts. In: Rudolph S, Dau F, Kuznetsov SO (eds) Supplementary Proceedings of the 17th International Conference on Conceptual Structures (ICCS) 2009. <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-483/paper1.pdf>
- Andrews S, McLeod K (2011) Gene Co-Expression in Mouse Embryo Tissues. In: Dau, F (ed) Proceedings of the 1st CUBIST (Combining and Uniting Business Intelligence with Semantic Technologies) Workshop, in conjunction with the 19th International Conference on Conceptual Structures (ICCS) 2011. CEUR Workshop Proceedings, Vol 753, ISSN 1613-0073
- Andrews S, Orphanides C (2010) FcaBedrock, a Formal Context Creator. In: Croitoru M, Ferre S, Lukose D (eds) Proceedings of the 18th International Conference on Conceptual Structures (ICCS) 2010. LNAI 6208. Berlin: Springer. pp. 181-184
- Andrews S, Orphanides C (2010) Analysis of Large Data Sets using Formal Concept Lattices. In: Kryszkiewicz M, Obiedkov S (eds) Proceedings of the 7th International Conference on Concept Lattices and Their Applications (CLA) 2010, ISBN 978-84614-4027-6. Seville: University of Seville. pp. 104-115
- Andrews S, Orphanides C, Polovina S (2011) Visualising Computational Intelligence through converting Data into Formal Concepts. In: Bessis N, Xhafa F (eds) Next Generation Data Technologies for Collective Computational Intelligence. Studies in Computational Intelligence (352). Berlin: Springer. pp. 139-166
- Becker P, Correia JH (2005) The ToscanaJ Suite for Implementing Conceptual Information Systems. In: Ganter B et al (eds) Formal Concept Analysis, LNCS (LNAI), vol. 3626, Springer-Verlag Berlin-Heidelberg, pp. 324-348
- Concept Explorer (2011). SourceForge, [<http://sourceforge.net/projects/conexp>]. Accessed December 2011
- CUBIST (2011) Combining and Uniting Business Intelligence with Semantic Technologies. EU FP7 project, <http://www.cubist-project.eu>. Accessed December 2011
- EMAGE gene expression database (2011) Edinburgh Mouse Atlas Project, <http://www.emouseatlas.org/emage/>. Accessed December 2011
- Eklund P, Wray T (2010) Social Tagging for Digital Libraries using Formal Concept Analysis. In: Kryszkiewicz M, Obiedkov S (eds) Proceedings of the 7th

- International Conference on Concept Lattices and Their Applications (CLA) 2010, ISBN 978-84614-4027-6. Seville: University of Seville. pp. 139-150
- Falk I, Gardent C, Lorenzo A (2010) Using Formal Concept Analysis to Acquire Knowledge about Verbs. In: Kryszkiewicz M, Obiedkov S (eds) Proceedings of the 7th International Conference on Concept Lattices and Their Applications (CLA) 2010, ISBN 978-84614-4027-6. Seville: University of Seville. pp. 151-162
- FcaBedrock Formal Context Creator (2011). SourceForge, [<http://sourceforge.net/projects/fcabedrock>]. Accessed December 2011
- Frank A, Asuncion A (2011) UCI Machine Learning Repository, [<http://archive.ics.uci.edu/ml>]. Irvine, CA: University of California, School of Information and Computer Science. Accessed December 2011
- Ganter B, Wille R (1989) Conceptual Scaling. In: Roberts F (ed) Applications of Combinatorics and Graph Theory to the Biological and Social Sciences. IMA, vol. 17. Berlin-Heidelberg-New York: Springer. pp. 139-168
- Ganter B, Wille R (1998) Applied lattice theory: formal concept analysis. In: Grätzer G (ed) General lattice theory. Birkhäuser Verlag
- In-Close Formal Concept Miner. SourceForge, [<http://sourceforge.net/projects/inclose>]. Accessed December 2011
- Poelmans J, Elzinga P, Viaene S, Dedene G (2010) Concept Discovery Innovations in Law Enforcement. In: Xhafa F, Demetriadis S, Caballe S, Abraham A (eds) Proceedings of the 2nd International Conference on Intelligent Networking and Collaborative Systems (INCoS) 2010. ISBN 978-0-7695-4278-2. IEEE Computer Society. pp. 473-478.
- Poelmans J, Elzinga P, Dedene G, Viaene S, Kuznetsov SO (2011) A Concept Discovery Approach for Fighting Human Trafficking and Forced Prostitution. In: Andrews S, Polovina S, Hill R, Akhgar B (eds) Proceedings of the 19th International Conference on Conceptual Structures (ICCS) 2011. LNAI 6828. Berlin: Springer-Verlag. pp. 50-62.
- Kaytoue-Uberall M, Duplessis S, Napoli A (2008) Using Formal Concept Analysis for the Extraction of Groups of Co-expressed Genes. In: Le Thi HA, Bouvry P, Pham Dinh T (eds) MCO 2008. CCIS vol. 14. Berlin/Heidelberg: Springer-Verlag. pp. 439-449.
- Krajca P, Outrata J, Vychodil V (2008) Parallel Recursive Algorithm for FCA. In: Belohlavek R, Kuznetsov SO (eds) Proceedings of the International Conference on Concept Lattices and their Applications (CLA) 2008. Palacky University, Olomouc. pp. 71-82.
- Lincoff GH (Pres) (1981) The Audubon Society Field Guide to North American Mushrooms, New York: Alfred A. Knopf.
- Orphanides C (2011) Exploring the Applicability of Formal Concept Analysis on Market Intelligence Data. In: Dau F (ed) Proceedings of the 1st CUBIST (Combining and Uniting Business Intelligence with Semantic Technologies) Workshop, in conjunction with the 19th International Conference on Conceptual Structures (ICCS) 2011. CEUR Workshop Proceedings, Vol 753, ISSN 1613-0073.
- Pensa RG, Boulicaut J-F (2005) Towards Fault-Tolerant Formal Concept Analysis. In: Advances in Artificial Intelligence, 9th Congress of the Italian Association for Artificial Intelligence.

- Priss U (2006) Formal Concept Analysis in Information Science. In: Cronin B (ed) Annual Review of Information Science and Technology, vol. 40. pp 521-543.
- Richardson L, Venkataraman S, Stevenson P, Yang Y, Burton N, Rao J, Fisher M, Baldock RA, Davidson DR, Christiansen JH. EMAGE mouse embryo spatial gene expression database: 2010 update. Nucleic Acids Research 2009; doi:10.1093/nar/gkp763
- Stumme G, Taouil R, Bastide Y, Lakhal L (2001) Conceptual Clustering with Iceberg Concept Lattices. In: Proceedings of GI-Fachgruppentreffen Maschinelles Lernen'01, Universitat Dortmund, vol. 763.
- Valtchev P, Missaoui R, Godin R (2004) Formal Concept Analysis for Knowledge Discovery and Data Mining: The New Challenges. In: Eklund P (ed) Concept Lattices, Proceedings of the Second International Conference on Formal Concept Analysis, ICFCA 2004 Sydney, Australia, February 23-26. Berlin: Springer.
- Wille R (2005) Formal Concept Analysis as Mathematical Theory of Concepts. In: Ganter B, Stumme G, Wille R (eds) Formal Concept Analysis: Foundations and Applications. Berlin: Springer. pp. 1-6.
- Wolff KE (1993) A First Course in Formal Concept Analysis. Available: http://www.fbmh.h-da.de/home/wolff/Publikationen/A_First_Course_in_Formal_Concept_Analysis.pdf. Last accessed 02 Sep 2011.
- Yevtushenko S (2000) System of data analysis "Concept Explorer" (In Russian). In: Proceedings of the 7th National Conference on Artificial Intelligence (KII) 2000, Russia. pp. 127-134.

Visual Analysis of a Large and Noisy Dataset

Honour Chika Nwagwu, Sheffield Hallam University, Sheffield, UK

Constantinos Orphanides, Sheffield Hallam University, Sheffield, UK

ABSTRACT

Visual analysis has witnessed a growing acceptance as a method of scientific inquiry in the research community. It is used in qualitative and mixed research methods. Even so, visual data analysis is likely to produce biased results when used in analysing a large and noisy dataset. This can be evident when a data analyst is not able to holistically explore, all the values associated with the objects of interest in a dataset. Consequently, the data analyst may assess inconsistent data as consistent when contradiction associated with the data is not visualised. This work identifies incomplete analysis as a challenge in the visual data analysis of a large and noisy dataset. It considers Formal Concept Analysis (FCA) tools and techniques and prescribes the mining and visualisation of Incomplete or Inconsistent Data (IID) when dealing with a large and noisy dataset. It presents an automated approach for transforming IID from a noisy context whose objects are associated with mutually exclusive many-valued attributes, to a formal context.

Keywords: *FcaBedRock, Formal Concept Analysis, Formal Context, Incomplete Data, Inconsistent Data, Large and Noisy Data Set, Research Method, Scaling Method, Visual Analysis*

1. INTRODUCTION

Visual data analysis provides a variety of techniques for scientific inquiry. These techniques include interactive exploration, graphical representation, and pictorial representation. But the use of visual data analysis as a research method is likely to produce biased results when a large and noisy dataset is investigated. This can be evident when a data analyst is not able to fully visualise the objects and associated attributes of interest.

Also, the inability of a visual apparatus to effectively depict all features of objects in a large and noisy dataset may lead to inaccurate analysis. For example, the percentage values associated with a thousand or more different attributes will not be effectively depicted in a pie chart or bar chart. More so, the inability of the data analyst to effectively visualise all the many-valued attributes in a large dataset, may lead to inaccurate analysis. For example, the data analyst may ascribe an inconsistent object as consistent or incomplete data as complete, where he did not visualise the inconsistency or incompleteness associated with the object. This work explores

DOI: 10.4018/IJCSSA.2015070102

the classical (conventional) and non-classical FCA approaches for dealing with IID in a large dataset. It examines how the classical and non-classical FCA approaches are applied on a dataset; notably, the dataset from the e-Mouse Atlas Gene Expression Database¹ (EMAGE).

EMAGE is a free online database which stores gene expressions information in the developing mouse embryo. It acquires the gene expression information from experimental results in journal publications, screening projects, and laboratory reports among others. Such gene expressions information is annotated with terms (standardised ontology) from Gene Ontology². The gene expressions information includes *detected* and *not detected* expressions information in tissues of the different Theiler Stages in the mouse. Detected gene expression information can be described as strong, moderate, or weak while not detected gene expression information can indicate that the experiment has not been performed, the experiment has been performed but the result is not clear, or the experiment has been performed but the result is not published by EMAGE.

EMAGE provides a real-life case study for illustrating how FCA approaches can be used to visually analyse IID. Its data contain objects which are associated with many-valued attributes. For example, a tissue (object) in an EMAGE database can be assigned gene expression information (attributes), such as detected or not detected, which is associated with many values. A gene detected in a tissue at a particular Theiler Stage may be associated with strong, moderate, or weak expression information. IID can be evident in EMAGE. For example, a tissue from a particular Theiler Stage can be detected as strong in one experiment and also as weak in another experiment. Such contradictory information is evident in the EMAGE database (McLeod and Burger 2011). Also, the information in EMAGE can be incomplete because not all the experiments about every gene in every tissue of every Theiler Stage have been performed. The possible causes of IID in EMAGE are described in (Nwagwu, 2013; McLeod and Burger, 2011).

This work does not only explore the classical and non-classical FCA approaches for dealing with IID, but also describes an automated method for the mining and visualising IID, as a means of dealing with the incomplete (partial) visual analysis of a large and noisy dataset. This approach enables the data analyst to adequately verify the soundness of the information from his investigated dataset as to avoid inaccurate conclusions. It is shown in this work, how FCA can be used to mine and visualise the IID from a noisy context whose objects are associated with mutually exclusive many-valued attributes.

Section 2 briefly introduces FCA. It explains FCA as consisting of visual analytical techniques and tools. It describes the classical FCA approach and it explains the challenges of visual analysis, notably incomplete visual analysis in the classical FCA approach. Section 3 describes the relevance of visual analysis by explaining the qualitative and mixed (qualitative and quantitative) FCA approaches for dealing with IID. It describes the non-classical FCA approaches notably fault tolerance, co-attribute expression, and semi-automated FcaBedrock approaches. Section 4 describes an improvement on the semi-automated FcaBedrock approach. This improvement involves automatically mining and transforming the IID from a large and noisy dataset into a formal context. A pseudocode depicting how the IID in objects that contain mutually exclusive many-valued attributes are mined and transformed to a formal context is presented. Dummy and real life examples are also presented. A conclusion is presented in section 5.

2. FORMAL CONCEPT ANALYSIS AS A VISUAL DATA ANALYSIS METHOD

Visual data analysis involves visual analytic tools and techniques and human analytical skills. Thomas and Cook (2006) explain that visual analytic tools and techniques enable the synthesis of

information and derivation of insight from massive, dynamic, ambiguous, and often conflicting data. This enables the detecting of the expected and discovering of the unexpected, providing timely, defensible, and understandable assessments, and communicating assessment effectively for action. This section explains how to visually analyse IID in classical FCA approach. It also describes the challenges associated with the approach.

2.1. The Classical FCA Approach for Visually Analysing IID

FCA was introduced by Rudolf Wille in (Wille, 1982). It is a mathematical method which uses the concept lattice as a formalism to explore correlations, similarities, refinements, anomalies, or even inconsistencies (Carpineto and Romano, 2004). It provides a data processing approach by which data is analysed by conceptually clustering objects with respect to a given set of attributes and visualising the set of clusters in a lattice structure. As a result, it enables visual analytical techniques.

FCA begins with a formal context. A formal context can be described as a single-valued context (G, M, I) . G represents a set of objects represented along the rows of a table, M is a set of attributes represented along the columns of the same table and I is a binary relation between G and M ($I \subseteq G \times M$) represented by a cross on the cell intersecting a particular object with its corresponding attribute. Table 1, is an example of a formal context. It describes some gene expression information in some tissues of the developing mouse embryo. The table describes a set of attributes (*Otx2_detected*, *Otx2_not_detected*, *Hoxb1_detected*, and *Hoxb1_not_detected*) and a set of objects (*node*, *mesoderm*, *organ system*, *neural ectoderm*, *future brain*, and *limb*). A cross in a cell is used to depict the presence of an experimental result showing that the gene is active (detected) or inactive (not detected) within the development of the corresponding mouse tissue. The absence of a cross in a cell indicates the experiment has not been performed, the experiment has been performed but the result is not clear, or the experiment has been performed but the result is not published by EMAGE. It is explained in (Wolff 1993; Burmeister and Holzer, 2005) that the absence of a cross in a formal context can also mean that it is not known whether the corresponding object has the corresponding attribute.

In FCA, if $x \in A$ and $y \in B$ (where A is a set of objects in a formal context K and B is a set of attributes in K) then xIy holds for the objects and attributes in K implying that the object x has the attribute y and the attribute y is a feature of the object x . For a subset of objects $A \subseteq G$, a derivation operator $'$ is defined to obtain the set of attributes common to the objects in A . Similarly, for a subset of attributes $B \subseteq M$, the derivation operator $'$ is defined to obtain the set of objects common to the attributes in B . A pair (A, B) is called formal concept if $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. The sets A and B are called the extent and the intent of the formal concept

Table 1. Formal context of gene expressions information in embryonic mouse tissues

	Otx2_detected	Otx2_not_detected	Hoxb1_detected	Hoxb1_not_detected
node	X			
mesoderm				X
organ system	X		X	
neural ectoderm	X	X		
future brain	X			
limb				

(A, B), respectively. The extent of a formal concept contains all objects that have the attributes of the intent while the intent contains all attributes shared by the objects of the extent.

Mathematically:

A formal context $K = (G, M, I)$

where:

G = a set of objects

M = a set of attributes

I = a binary relation between G and $M \rightarrow I \subseteq G \times M$

The extent 'A' (set of objects) and the intent 'B' (set of attributes) of each formal concept in a formal context K can be defined as follows:

(A, B) with $A \subseteq G$, $B \subseteq M$, $A = B'$, and $B = A'$

where:

$A' = \{y \in Y \mid \text{for each } x \in A \text{ then } (x, y) \in I\}$

$B' = \{x \in X \mid \text{for each } y \in B \text{ then } (x, y) \in I\}$

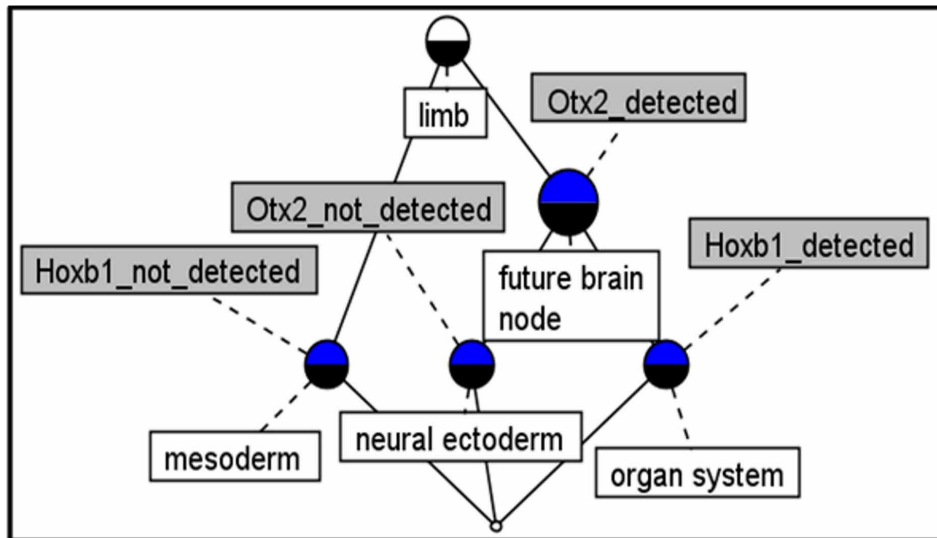
A and B are called the extent and the intent of the formal concept (A, B) and they define a formal concept in a formal context. Some of the formal concepts identified from Table 1 are as listed below:

$\{(mesoderm), (Hoxb1_not_detected)\}$
 $\{(neural\ ectoderm), (Otx2_not_detected, Otx2_detected)\}$
 $\{(organ\ system), (Otx2_detected, Hoxb1_detected)\}$
 $\{(node, organ\ system, future\ brain, neural\ ectoderm), (Otx2_not_detected)\}$

When a set of objects is described by a set of attributes, it becomes easy to identify which attribute is associated with which objects or which object is associated with which attributes. On this note, IID can easily be identified by noting the objects which are associated with contradictory attributes or the objects which are not associated with the set of attributes. In Table 1, *neural ectoderm* is associated with contradictory attributes- (*Otx2_not_detected*, *Otx2_detected*). The tissue '*limb*' is not associated with any attribute. The *neural ectoderm* is inconsistent while the *limb* is incomplete. However, visually identifying IID in a large formal context can be challenging. As explained in section 1, there is a need to visually identify all objects which are inconsistent or incomplete in a context. The identification of such instances from a large formal context can be easier when the concept lattice is built from the formal context.

A concept lattice is a structured diagram which is composed of one or many nodes. A concept lattice can be built from a formal context through the use of FCA tools such as ToscanaJ, Toscana, or ConExp. Building a concept lattice involves mining formal concepts in a formal context and displaying them hierarchically in a lattice structure. Figure 1, is a concept lattice built from Table 1 through the use of the ConExp³.

Figure 1. Concept lattice showing gene expressions information in tissues of a mouse



The concept lattice displays objects with similar attributes (concepts). A concept lattice is easy to read. The simple reading rule of a concept lattice as described in (Wolff 1993) is that an object *g* has an attribute *m* if and only if there is an upwards leading path from the circle named by “*g*” to the circle named by “*m*”. Consequently, it can easily be visualised from a concept lattice, objects which are not associated with any attribute or attributes which are not associated with any object (incomplete data). In Figure 1, *Otx2* is both *detected* and *not detected* in *neural ectoderm* hence *neural ectoderm* is as such inconsistent. Figure 1 provides an example of incomplete data displayed at the top of a lattice. The object *limb* is incomplete because it is not associated with any attribute.

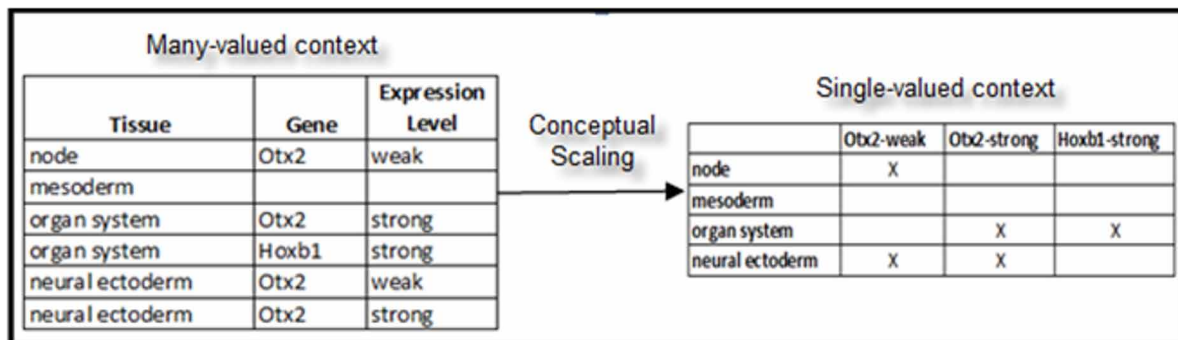
A concept lattice provides an easy means of identifying IID but there are challenges with its readability when visually analysing a large concept lattice. A description of such challenges is provided in the section below.

2.2. Challenges of the Classical FCA Approach in Dealing with IID

A data analyst may not fully visualise all the attribute values associated with an object of interest when dealing with a large and noisy dataset. Consequently, the data analyst may ascribe an inconsistent data as consistent where he did not visualise the contradictory attribute value. The data analyst may also ascribe an incomplete object (object with missing attribute value) as complete. Such lapses in data analysis can be described as incomplete analysis. Data analysis through classical FCA approach can lead to incomplete analysis. This is because classical FCA approach does not adequately enable the visualisation of the IID in a large and noisy dataset. Issues of incomplete analysis may therefore occur where the classical FCA approach is applied to visually identify or evaluate IID.

In classical FCA, a many-valued context is transformed to a single-valued context (formal context) through the use of a conceptual scaling method (Messai et al., 2008; Annoni and Brüggemann, 2008; Carpineto and Romano, 2004). This is often achieved by replacing every many-valued attribute in the record set by the corresponding attribute-value pairs, with each object being described by one attribute-value pair, per many valued attribute. An example of such a transformation is shown in Figure 2.

Figure 2. An example of a transformation from many-valued to a single-valued context



The conceptual scaling method, as applied in classical FCA provides a global view of the object-attribute-value relations of a many-valued context by replacing every many-valued attribute in the context with the corresponding attribute-value pairs. For example, the attribute *Otx2* is associated with the values *weak* and *strong* in the many-valued context of Figure 2. A transformation of this many-valued attribute to a single-valued attribute will result in the attribute-value pair '*Otx2-weak*' and '*Otx2-strong*'. Although the conceptual scaling has the advantage of presenting a global view of its represented data, its use in a large dataset can result in a very large formal context. It is noted in (Carpineto and Romano, 2004 p. 26; Dau, 2013a; Dau, 2013b) that an increase in the size of formal context could result in an exponential rise in the number of formal concepts. A concept lattice which has many formal concepts will also have many crossing edges. These factors (many formal concepts and many crossing edges) will lead to poor identification or visualisation of attributes of interest such as IID in a concept lattice, by the data analyst. Hence, visualising IID through the use of classical FCA approach is challenging when dealing with a large dataset. Section 3 explains the non-classical FCA approach to visually analysing IID.

3. NEW FCA APPROACHES FOR VISUALLY ANALYSING IID

IID in a large and noisy dataset can be visually identified by associating a description with the IID in the dataset. Melo et al. (2013) used distinct colours to identify the different types of inconsistent data from the eMouse Atlas Gene Expression Database (EMAGE) dataset. Also, the amount of IID in information from a noisy dataset can be evaluated. This is demonstrated by Nwagwu (2013) where it is explained how inconsistent data can be evaluated and depicted in a tabular structure. The works of (Hunter, 2002; Grant and Hunter, 2011; Mu et al., 2011; Bertossi, et al., 2005; Scheglmann, et al., 2013; Libkin, 2014) explain different statistical and logical means of dealing with IID. But all these approaches may not precisely identify the IID in a large and noisy dataset and they are not the focus of this write-up.

Visual analytical FCA tools and techniques are the focus of this work. They can be used by a data analyst to perceive the IID in a large and noisy dataset. The classical FCA approach does not adequately enable the visualisation of IID when dealing with a large and noisy dataset. Its use can result in an incomplete analysis of an investigated data. Nevertheless, FCA presents data analysis and visualisation techniques which have been shown in the CUBIST project⁴ to be useful and appropriate for the analysis of large dataset. Also, its tools and techniques enable the analysis of a large dataset. For example, Andrews and Orphanides (2010) demonstrate how some open-source FCA tools notably FcaBedrock⁵ and In-Close⁶ are used to analyse large formal

context. It is shown in (Dau, 2013a; Melo, et al. 2013; Nwagwu, 2014), how different FCA tools and techniques are used to identify and visualise IID existing in a dataset.

It is shown in (Nwagwu, 2014; Dau 2013a), how IID in a large and noisy dataset is identified by visualising the attributes of the IID in the dataset. Jiang et al. (2009) used a node without a label for its own object in a concept lattice to show an anonymous node. These works present examples of descriptive or qualitative approach to visually analysing IID. It is also shown in (Nwagwu, 2014), how the amount of IID existing in a dataset can be evaluated and also depicted in a concept lattice. This is achieved by measuring the IID existing in each object of a dataset and depicting the amount of IID, its corresponding attributes and the associated object in a concept lattice. This approach presents an example of a mixed approach to visually analysing IID.

Both the qualitative and the mixed FCA approaches to visually analysing IID are designed to describe the IID in a dataset. The data analyst also uses other data exploration techniques, as provided by the visual analytic tool such as interactive exploration and filter options to facilitate the qualitative or quantitative description and visualisation of IID. Even so, the identification and visualisation of IID through visual analysis may not always be accurate. This may be because of the inability of the data analyst to visually analyse many-dimensional attribute-values or the inability of the visual apparatus to clearly depict the IID as explained in the introduction of this work. More so, the use of descriptive attributes in visual analysis remains a concern. Sandelowski (2000) notes that “basic qualitative description is not highly interpretive in the sense that a researcher (data analyst) deliberately chooses to describe an event in terms of a conceptual, philosophical, or other highly abstract framework or system.” This entails that the researcher may not use the appropriate attributes to describe the investigated data. For example, the choice of a scale or descriptive attributes for ordinal data such as high, medium, and low by the data analyst may not always appropriately convey the actual context. Thomas and Cook (2006), note that the quality of data visualization is directly affected by the quality of its representation underlying the visualization. Dau (2013a) notes that the choice of the formal attributes of a scale in FCA is a question of the design of the scale. The formal attributes are meaningful attributes to describe the values and might be different entities or might even be the values of the property again. Consequently, visual analytical approaches may lead to incomplete analysis.

However, the visualisation of the values associated with attributes of the IID in the investigated dataset may ensure an unbiased identification and visualisation of the IID. This is because such a view provides an avenue by which the IID in the investigated dataset is clearly identified. This is achievable through the use of FCA tools and techniques. Section 3.1 below presents some non-classical FCA techniques for dealing with IID in a large and noisy dataset.

3.1. Non-Classical FCA Approaches for Visually Analysing IID

The non-classical FCA approaches for visually analysing IID are presented in works such as (Andrews and McLeod, 2011; Dau, 2013a; Dau, 2013b; Melo, et al., 2013; Nwagwu, 2014). They include fault tolerance, co-attribute expression, and semi-automated FcaBedrock approaches. These FCA approaches are briefly discussed as follows.

Fault tolerance is discussed in (Dau, 2013b; Andrews and McLeod, 2011). It is an approach that is used to explore and reason with incomplete data. It is implemented by substituting certain amounts of the missing data in a dataset with actual values. Andrews and McLeod (2011) explains that fault tolerance in FCA provides a medium by which a certain amount of missing information can be tolerated by adding a limited number of relations to complete a concept. This approach, when applied to a noisy dataset, produces interesting and manageable lattices as evident in (An-

draws and McLeod 2011; Melo et al. 2013). It enables a systematic reasoning with incomplete information but does not provide a means by which IID in the dataset can be visualised.

Dau (2013a) presents an FCA approach that enables the visualisation of IID in a dataset. The author built a prototype tool (named “SPARQL context creator”) that facilitates the retrieval of data from a triple store and the transformation of the retrieved data to a formal context. A concept lattice is built from the formal context and any IID in the concept lattice may be visualised by the data analyst. But, this approach does not provide a means of retrieving only the IID as to ensure the identification of such IID from a large concept lattice.

Another approach of visually analysing IID is the use of co-occurrence function. The co-occurrence function as explained in Melo et al. (2013) is used to identify objects that share any selected two attributes in an EMAGE dataset. The result of such exploration is displayed in a concept lattice and also in an associated chart as described in a CUBIST YouTube⁷ presentation. In doing so, contradictory attributes can be selected and objects sharing such attributes can be visualised. Also, different FCA techniques for reasoning with the incomplete or inconsistent EMAGE data are explained in (Melo, et al., 2013). Some of these reasoning methods include automatically fixing of flip-flops and fault tolerance techniques. However, these techniques may not be easily applied in other datasets without some corresponding scaling of the data. This may also require customising the CUBIST application to suit the investigated dataset. Melo et al. (2013) assert that “*Although most of the functionalities in CUBIST can be used with other data than EMAGE (with the corresponding scaling of data), as future work we will extend our experiments to other genes expression datasets like cancer and brain development.*”

The semi-automated FcaBedrock approach is presented in Nwagwu (2014). It is an approach for dealing with IID in a dataset. It is designed to identify IID in objects whose many valued-attributes are mutually exclusive. It works through the fact that mutually exclusive many-valued attribute has disjointed values. Consequently, in a noisy dataset which contains objects associated with mutually exclusive many-valued attributes, an attribute can be consistent, inconsistent, or incomplete. A consistent attribute of an object, is associated with only a value from a set of mutually exclusive values. For example, a gene can only be associated with ‘detected’ in a set of mutually exclusive attribute values - ‘detected’ and ‘not_detected’. An inconsistent attribute of an object, is associated with more than a value from a set of mutually exclusive attribute values. For example, a gene in a tissue of a Theiler Stage can be associated with both ‘strong’ and ‘weak’ gene expression information. An incomplete attribute of an object, will not be associated with any value from a set of mutually exclusive values. For example, a gene may not be associated with ‘strong’, ‘moderate’ or ‘weak’. The semi-automated approach uses the FcaBedrock to manually restrict single-valued attributes (consistent data), thereby transforming any incomplete or inconsistent attribute values in the investigated dataset to a formal context. The FcaBedrock is an FCA tool that is used to convert a many valued record set to a context file. It allows its users to optionally select which attributes to convert to a context file through a process of guided automation. A concept lattice is subsequently built from the formal context file. Also, the concept lattice is edited to enable the visualisation of only the objects whose attribute contain IID. The semi-automated FcaBedrock approach can be painstaking, especially when there are many attributes to be restricted in FcaBedrock.

4. OUR APPROACH: AUTOMATED FCABEDROCK APPROACH

A dataset can consist of objects associated with many-valued attributes. The many-valued attributes may include only mutually exclusive values. Such context can be transformed to a formal

context whose attribute values are mutually exclusive. There are many real life instances where objects are associated with many-valued attributes which are mutually exclusive. For example, students (objects) are associated with various departments (attributes) in an institution. The departments can offer its students, modules which are mutually exclusive. Another example is EMAGE. In the EMAGE, each gene from each tissue in each Theiler Stage can be associated with detected gene expression information. The detected gene expression information is a mutually exclusive many-valued attribute such that a gene can be associated with strong, moderate, or weak expression information. An object (such as a gene from a tissue in a Theiler Stage) that is associated with mutually exclusive attribute values (such as moderate and weak expression information) is inconsistent. Also objects in a mutually exclusive many-valued context which are not associated with any mutually exclusive attribute values are incomplete.

The approach presented in this section is the Automated FcaBedrock approach and the developed application is deposited in sourceforge⁸. It is an FCA approach which begins with a formal context consisting of many-valued attributes that are mutually exclusive. If $x \in A$, $y \in \{B_1, B_2, \dots, B_n\}$ and $|y| > 1$ (where A is a set of objects in a formal context K , B_1, B_2, \dots, B_n are different sets of mutually exclusive many-valued attributes in the formal context K , y is a feature of any of the mutually exclusive many-valued attributes and $|y|$ is the number of elements in a set of mutually exclusive many-valued attribute) then, xIy holds for the objects and many-valued attributes in K if the object x has more than one value in an associated many-valued attribute.

The automated FcaBedrock approach enables the identification and visualisation of objects whose attributes are associated with mutually exclusive values or objects that are not associated with investigated many-valued attributes. It automatically mines IID from a many-valued context and transforms the retrieved IID to a formal context. This is achieved through selecting the inconsistency mode in FcaBedrock when transforming a mutually exclusive many-valued context to a formal context. The formal context is subsequently built to a concept lattice through the use of ConExp. The pseudocode in Algorithm 1 describes how the inconsistency function in FcaBedrock was coded.

The automated FcaBedrock approach involves the following summarised steps:

- A noisy dataset provided in Comma Separated Values (CSV) format which consists of objects which are associated with mutually exclusive many-valued attributes is read by FcaBedrock;
- IID can be retrieved from the dataset and transformed to a formal context by selecting the Inconsistency mode in FcaBedrock;
- The output file (formal context) from FcaBedrock is subsequently visualised through a formal context builder such as ConExp.

The automated FcaBedrock approach resolves the lapses of the semi-automated FcaBedrock approach. As explained in (Nwagwu, 2014), the semi-automated FcaBedrock approach is tedious and does not always enable the holistic visualisation of the IID in a dataset. The approach may not mine only the IID existing in a large and noisy dataset. Figure 3b is the result of applying the automated FcaBedrock approach on an EMAGE dataset. The dataset is a result set retrieved through querying EMAGE database as explained in (Nwagwu, 2014). Unlike the semi-automated FcaBedrock approach, the automated FcaBedrock approach displays only the IID in a concept lattice. It can be seen that Figure 3b shows only the inconsistent tissues and their associated attribute-values. These tissues include *neural ectoderm*, *primitive streak*, and *mesoderm*. This is different from Figure 3a, which did not only display the IID in the concept

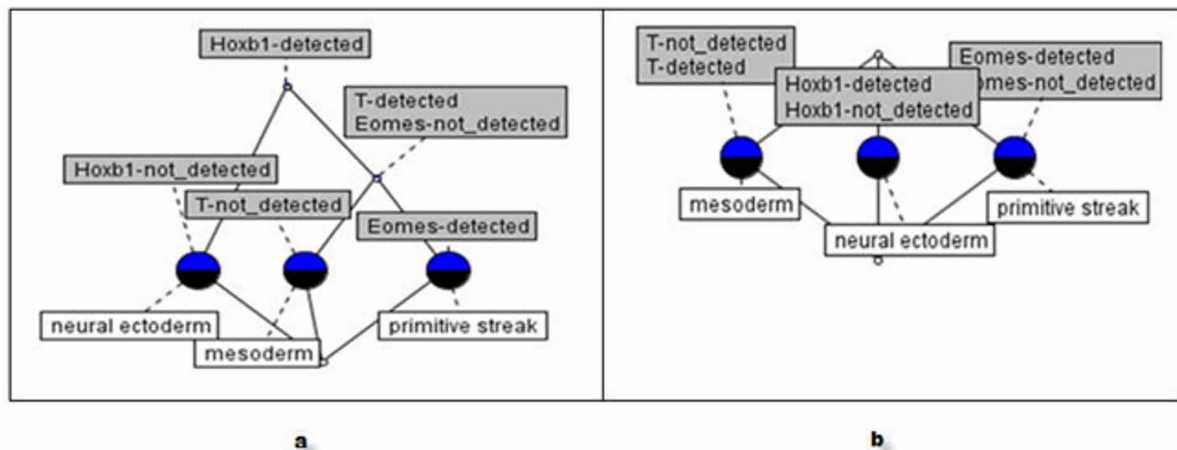
Algorithm 1. Pseudocode for the inconsistency mode in FcaBedrock

```

1: In a formal context K= (G, M, I)
2: Let G = all objects
3: Let M = all attributes
4: Let O(q,w) = empty list where q = inconsistent object index and w = inconsistent attribute values
5: Let counter = counter to count attribute-values //(i.e. all values of EACH attribute) for each object
6: For every object i in G
7:   counter = 0
8:   For every attribute j in M
9:     For every attribute-value x in M[j]
10:      If G[i] contains x then counter = counter + 1
11:    End For
12:  End For
13:  If counter > 1 then
14:    //inconsistency right here – object G[i] has more than one value
15:    //for attribute M[j], so save its index and the attribute which renders
16:    //this object inconsistent in O
17:    O.add(G[i], M[j])
18:  End If
19: End for
20: //So now O contains all objects along with the attributes which render each object inconsistent.

```

Figure 3. Concept lattices processed by the semi-automated FcaBedrock approach (a) and the automated FcaBedrock approach (b) as built from the EMAGE dataset



lattice. For example, in Figure 3a, *Hoxb1-detected* is associated with *mesoderm* which does not contradict any of its attribute-values.

5. CONCLUSION AND FUTURE WORK

This work has explained how incomplete analysis of information in a large and noisy dataset can cause bias to the visual analysis research method. The various FCA approaches of dealing with IID were examined. It is shown in this work, how IID in a dataset is automatically mined and transformed into a formal context file. The proposed FCA approach ‘automated FcaBedrock

approach' provides a means by which only the IID in a large and noisy dataset is visualised in a concept lattice, thereby avoiding inaccurate analysis of an investigated data in the dataset. The automated FcaBedrock approach is shown to be a more robust approach which enables the visualisation of IID in a large and noisy formal context than its contemporary semi-automated FcaBedrock approach.

This work has considered IID in objects whose attribute-values are mutually exclusive. There are other areas in which IID can thrive, such as Remote Sensing and Geographical Information System (GIS) as described in Devillers and Jeansoulin (2006 p. 17- 29, 184-207). The researchers hope to investigate how FCA may be applied to deal with IID in such areas. For example, similar pictures captured with a camera can be converted into digital form and stored in different datasets. Checks can be made with the stored data to identify distortions, inconsistencies or incompleteness. This can be achieved through visually analysing the associated concept lattices which display the IID in the datasets.

REFERENCES

- Andrews, S., & McLeod, K. (2011). Gene co-expression in mouse embryo tissues. In F. Dau (Ed.), *Proceedings of the 1st CUBIST (Combining and Uniting Business Intelligence with Semantic Technologies) Workshop 2011* (Vol. 753).
- Andrews, S., & Orphanides, C. (2010). Analysis of large data sets using formal concept lattices. In *Proceedings of the 7th International Conference on Concept Lattices and Their Applications*, Seville, University of Seville (pp. 104-115).
- Annoni, P., & Brüggemann, R. (2008). The dualistic approach of FCA: A further insight into Ontario Lake sediments. *Chemosphere*, 70(11), 2025–2031. doi:10.1016/j.chemosphere.2007.09.019 PMID:17961630
- Bertossi, L., Hunter, A., & Schaub, T. (2005). Introduction to inconsistency tolerance. In *Inconsistency Tolerance* (pp. 1–14). Springer Berlin Heidelberg. doi:10.1007/978-3-540-30597-2_1
- Burmeister, P., & Holzer, R. (2005). Treating incomplete knowledge in formal concept analysis. In *Formal Concept Analysis* (pp. 114–126). Springer Berlin Heidelberg.
- Carpineto, C., & Romano, G. (2004). *Concept data analysis: Theory and applications*. Wiley.
- Dau, F. (2013a). Towards Scalingless Generation of Formal Contexts from an Ontology in a Triple Store. *International Journal of Conceptual Structures and Smart Applications*, 1(1), 18–38. doi:10.4018/ijc-ssa.2013010102
- Dau, F. (2013b). An Implementation for Fault Tolerance and Experimental Results. In *CUBIST Workshop* (pp. 21-30). Retrieved from <http://ceur-ws.org/Vol-1040/paper3.pdf>
- Devillers, R., & Jeansoulin, R. (2006). *Fundamentals of spatial data quality* (p. 312). ISTE Publishing Company. doi:10.1002/9780470612156
- Grant, J., & Hunter, A. (2011, July). Measuring the good and the bad in inconsistent information. *Proceedings-International Joint Conference on Artificial Intelligence IJCAI*, 22(3), 2632.
- Hunter, A. (2002, July). Measuring inconsistency in knowledge via quasi-classical models. *Proceedings of AAAI/IAAI* (pp. 68-73).
- Jiang, G., Pathak, J., & Chute, C. G. (2009). Formalizing ICD coding rules using formal concept analysis. *Journal of Biomedical Informatics*, 42(3), 504–517. doi:10.1016/j.jbi.2009.02.005 PMID:19236957
- Libkin, L. (2014, June). Incomplete data: what went wrong, and how to fix it. *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems* (pp. 1-13). ACM. doi:10.1145/2594538.2594561

McLeod, K., & Burger, A. (2011). WP7 requirement document of CUBIST Consortium 2010-2013. Retrieved from http://www.cubist-project.eu/fileadmin/CUBIST/user_upload/Deliverable/CUBIST_D7.1.1_HWU_v1.0.pdf

Melo, C., Aufaure, M. A., Orphanides, C., Andrews, S., McLeod, K., & Burger, A. (2013, March). A conceptual approach to gene expression analysis enhanced by visual analytics. *Proceedings of the 28th Annual ACM Symposium on Applied Computing* (pp. 1314-1319). ACM. doi:10.1145/2480362.2480610

Messai, N., Devignes, M.D., & Napoli, A., & Smail-Tabbone, M. (2008, July). *Many-Valued Concept Lattices for Conceptual Clustering and Information Retrieval* (Vol. 178, pp. 127–131). ECAI.

Mu, K., Liu, W., & Jin, Z. (2011). A general framework for measuring inconsistency through minimal inconsistent sets. *Knowledge and Information Systems*, 27(1), 85–114. doi:10.1007/s10115-010-0295-y

Nwagwu, H. (2013). Evaluating and Analyzing Inconsistent RDF Data in a Semantic Dataset: EMAGE Dataset. *Proceedings of the 3rd CUBIST Workshop*. Retrieved from <http://ceur-ws.org/Vol-1040/paper5.pdf>

Nwagwu, H. C. (2014). Visualising Inconsistency and Incompleteness in RDF Gene Expression Data using FCA. *International Journal of Conceptual Structures and Smart Applications*, 2(1), 68–82. doi:10.4018/ijcssa.2014010105

Sandelowski, M. (2000). Focus on research methods-whatever happened to qualitative description? *Research in Nursing & Health*, 23(4), 334–340. doi:10.1002/1098-240X(200008)23:4<334::AID-NUR9>3.0.CO;2-G PMID:10940958

Scheglmann, S., Groener, G., Staab, S., & Lämmel, R. (2013, January). Incompleteness-aware programming with RDF data. *Proceedings of the 2013 workshop on Data driven functional programming* (pp. 11-14). ACM. doi:10.1145/2429376.2429380

Thomas, J. J., & Cook, K. A. (2006). A visual analytics agenda. *Computer Graphics and Applications, IEEE*, 26(1), 10–13. doi:10.1109/MCG.2006.5 PMID:16463473

Wille, R. (1982). Restructuring lattice theory: an approach based on hierarchies of concepts. In I. Rival (Ed.), *Ordered sets* (pp. 445–470). Dordrecht, Boston: Reidel. doi:10.1007/978-94-009-7798-3_15

Wolff, K. E. (1993). A first course in formal concept analysis. *SoftStat*, 93, 429–438.

ENDNOTES

1. <http://www.eMouseatlas.org/emage/>
2. <http://geneontology.org/>
3. <http://sourceforge.net/projects/conexp/>
4. www.cubist-project.eu
5. <http://sourceforge.net/projects/fcabledrock/>
6. <http://sourceforge.net/projects/inclose/>
7. https://www.youtube.com/watch?v=Kuu756nr1_I
8. <http://sourceforge.net/projects/fcabledrock/>

Honour Chika Nwagwu is a PhD student in Sheffield Hallam University. He holds an MSc (Distinction) in Database Professional from the same University. He has a BSc in Computer Science from Michael Okpara University of Agriculture, Umudike in Nigeria. Nwagwu is a member of the Editorial Review Board of International Journal of Conceptual Structures and Smart Applications (IJCSSA), a member of the Institution of Engineering and Technology (IET), and an Oracle Certified Professional (OCP). His interest is on knowledge representation and visual data analysis. He is also interested in how to deal with inconsistent and incomplete data. He has a number of publications among which include “Visualising Inconsistency and Incompleteness in RDF Gene Expression Data using FCA.” In his spare time, Nwagwu enjoys internet browsing and reading. He is a father of two children and lives in Sheffield, United Kingdom.

Discovering Knowledge in Online Drug Transactions Using Conceptual Graphs and Formal Concept Analysis

Constantinos Orphanides*, Babak Akhgar* and Petra Saskia Bayerl*†

*Centre of Excellence in Terrorism, Resilience, Intelligence & Organised Crime Research
Sheffield Hallam University, Sheffield, UK

†Rotterdam School of Management, Erasmus University, Rotterdam, NL

*{c.orphanides, b.akhgar}@shu.ac.uk †bayerl@rsm.nl

Abstract—In this short paper, we describe a conceptual approach in which Conceptual Graphs (CGs) and Formal Concept Analysis (FCA) are employed towards knowledge discovery in online drug transactions. The transactions are acquired by performing Named-Entity Recognition (NER) on documents crawled from online public sources such as Twitter and Instagram, and are structured based on a CG ontology created to model such transactions. The drug transactions are then visualized using FCA as the knowledge discovery method.

I. INTRODUCTION

The challenges of developing efficient and effective architectures for environment scanning for strategic early warning of Organised Crime (OC) [3], [4] have shown how, more often than not, approaches of conceptual nature are needed. In the pursuit of knowledge discovery and intuitive visualizations for identifying potential patterns in OC data, Formal Concept Analysis (FCA), with its knowledge discovery facilities already evidenced in the literature [2], [6], [9], has been shown to be an effective approach in the problem domain [1].

This short paper describes a conceptual approach towards discovering knowledge in online drug transactions. Section II provides a brief overview of the three technologies involved, namely Conceptual Graphs (CGs), Named-Entity Recognition (NER) and Formal Concept Analysis (FCA). Section III explains how these technologies are combined and used together to provide the desired results. An example is shown in section IV, to demonstrate the practical applicability of our approach, followed by evaluation in section V. Finally, concluding remarks are provided in section VI.

II. PRELIMINARIES

A. Conceptual Graphs

Conceptual Graphs (CGs) are conceptual structures, based on a field of mathematics called graph theory, introduced by John Sowa in his book “Conceptual Structures: Information Processing in Mind and Machine” [8]. They have been applied to numerous domains including natural language processing, information systems modeling, program specification and machine learning [8].



Fig. 1. CG of “Cocaine is extracted from Coca leaves”

A CG is a bipartite graph comprising of concepts and relations. Figure 1 shows a simple CG, comprising of two concepts and one relation. The direction of the arcs between concepts and relations assists the direction of the reading, distinguishing the source concept from the target concept. Therefore, *Coca_Leaf* (target concept) relates to *Cocaine* (source concept) by the relation *extract_of*, thus forming the CG “Cocaine is extracted from Coca leaves”.

B. Named-Entity Recognition

Named-Entity Recognition (NER) is a subtask of information extraction and Natural Language Processing (NLP) which aims to locate and classify elements found in free-text data into predefined categories such as names, persons, organizations, locations, and dates [5]. For example, using NER, the sentence “John Doe sells cocaine in Manchester” can be converted into an annotated block of text that highlights the identified entities:

Person → John Doe
Drug → Cocaine
Location → Manchester

Thus, NER provides the ability to produce structured representations of text and to put information in semantically precise forms, which allow further inferences to be made by computer algorithms, and analyses to be carried out by analysts.

C. Formal Concept Analysis

Formal Concept Analysis is a term that was introduced by Rudolf Wille in 1984 and builds on “applied lattice and order theory that was developed by Birkhoff and others in the 1930’s” [10]. FCA was initially developed as a subsection

of Applied Mathematics, based on the mathematisation of concepts and concepts hierarchies.

In FCA, *formal concepts* can be described by beginning with a set of objects G and a set of attributes M . A relation I is understood to be a subset of the cross product between the sets it relates, so $I \subseteq G \times M$. This is called the *formal context*, which can be represented as a cross-table [7].

Flights	Latin America	Europe	Canada	Asia Pacific	Middle East	Africa	Mexico	Caribbean	USA
Air Canada	×	×	×	×	×		×	×	×
Air New Zealand		×		×					×
Nippon Airways		×		×					×
Ansett Australia				×					
Austrian Airlines		×	×	×	×	×			×

The cross-table above is a formal context representing the destinations of five airlines, where the elements on the left are *formal objects* (airlines) and the elements at the top are *formal attributes* (destinations).

If an object has a specific attribute, it is indicated by placing a cross in the corresponding cell of the table. An empty cell indicates that the corresponding object does not have the corresponding attribute. In the airlines context, Air Canada flies to Latin America but does not fly to Africa. As such, if an object g has an attribute m , then $g \in G$ relates to m by I , so we write $(g, m) \in I$, or gIm . For a subset of objects $A \subseteq G$, a derivation operator $'$ is defined to obtain the set of attributes, common to the objects in A , as follows:

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

Similarly, for a subset of attributes $B \subseteq M$, the derivation operator $'$ is defined to obtain the set of objects, common to the attributes in B , as follows:

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

In a given formal context $\mathbb{K} = (G, M, I)$, a pair (A, B) is a *formal concept* iff $A \subseteq G$, $B \subseteq M$, $A' = B$ and $B' = A$. The set A is the *extent* of the concept and the set B is the *intent* of the concept. A formal concept is, therefore, a closed set of object/attribute relations, in that its extension contains all objects that have the attributes in its intension, and the intension contains all attributes shared by the objects in its extension [10].

In the airline formal context, it can be seen that Air Canada and Austrian Airlines fly to both USA and Europe. However, this does not constitute a formal concept because both airlines also fly to Asia Pacific, Canada and the Middle East. Adding these destinations completes (closes) the formal concept:

$$(\{\text{Air Canada, Austrian Airlines}\}, \{\text{Europe, USA, Asia Pacific, Canada, Middle East}\}).$$

A formal context can be visualized as a *concept lattice* (Figure 2), which is an intuitive way of discovering hitherto undiscovered information in data and portraying the natural

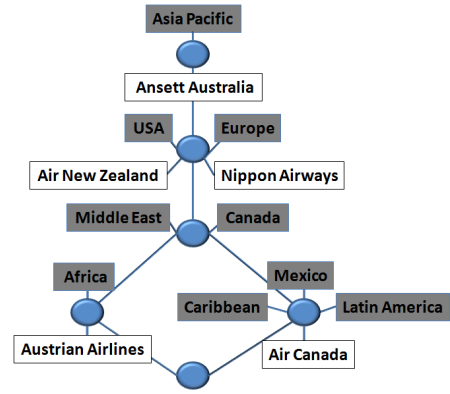


Fig. 2. A lattice corresponding to the airlines context

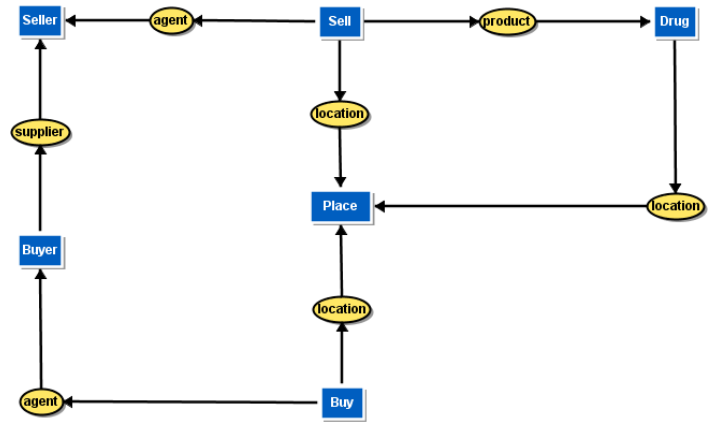


Fig. 3. A CG ontology of drug transactions

hierarchy of concepts that exist in a formal context. Each node in the lattice is a formal concept. The objects (airlines) are conventionally labeled below the nodes, while the attributes (destinations) are labeled above the nodes.

Extracting information from a lattice is straightforward. In order to see which attributes are featured by an object, one begins from the node where the object is located and moves upwards; all attributes one meets along the way are the attributes featured by that object. For example, if one heads upwards in the lattice from Air New Zealand (object), one will collect the attributes USA, Europe and Asia Pacific. This is interpreted as ‘Air New Zealand flies to USA, Europe and Asia Pacific’. Similarly, in order to see which objects have a particular attribute, one heads downwards in the lattice; any objects one meets along the way are objects which feature that attribute. For example, heading downwards from Canada (attribute), one will collect the objects Austrian Airlines and Air Canada. This is interpreted as ‘Canada is flown to by Austrian Airlines and Air Canada’.

III. THE APPROACH

Our approach begins by creating a CG ontology to capture the concepts and relations between entities in drug transactions

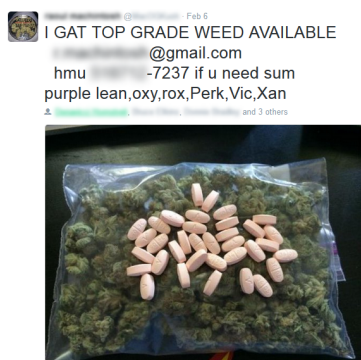


Fig. 4. An example of a Twitter post by an online drug seller

(Figure 3). We have kept the ontology simple and concise, only capturing the major entities involved such as the buyer, the seller and the location where the transaction took place. The ontology can of course be extended to capture more relations and involve more entities.

We then perform NER on a collection of documents crawled from online sources such as newspapers, OC-related websites and public social media posts. Our NER categorization and extraction rules were built using SAS Content Categorization Studio (CCS). The NER process includes:

- Industry-specific taxonomies, included in CSS, for identification of organizations and locations. These taxonomies are configurable and extendable.
- Bespoke taxonomies and rules, developed in-house, for identification of drug transactions. This is achieved in CCS by constructing Boolean rules and regular expressions that are applied to the crawled documents during the NER process. By combining multiple rules together, rules can be made more complex; for example, a rule could look for the combination of a drug and a location mentioned in the same sentence, and having a maximum distance of five words between the two.

With regards to the social media sources used, namely Twitter and Instagram, some assumptions are being made:

- When a user replies to a drug seller’s online post (such as the Twitter example in Figure 4) with a contextually related statement (e.g. “do you deliver in the Seattle area?” or “i’d like 2 grams, please message me”), we assume that this interaction has resulted in a finalized drug transaction.
- When locations are explicitly mentioned in a post, we infer these locations to be the ones where the transactions are taking place. If not, we attempt to extract the location from the user’s social media profile. Posts where no location is mentioned or inferred were discarded and excluded altogether from the analysis in section IV.

Finally, we employ the open-source FCA tool CGFCA¹ to create the formal attributes of the formal context based on the (*SourceConcept*, *Relation*, *TargetConcept*) triples

¹<http://sourceforge.net/projects/cgfca/>

in the CG ontology. CGFCA creates an FCA-equivalent of the actual CG, whereas our approach also *implements* the CG. For each triple, the formal object is a crawled document, rather than the *TargetConcept* value of the CG, and the value of *SourceConcept* comes from the corresponding value extracted from the crawled documents during the NER process. Once the process is complete, we use the open-source FCA tool FcaBedrock² to create the formal context. Lastly, we visualize the formal context as a lattice using ConExp³. The results are explained in section IV.

IV. RESULTS

Figure 5 shows the “drug transactions” lattice created by the formal context of the process described in section III. It is the result of a total of 119 crawled documents containing drug transactions, as identified by the NER process. There are a total of 20 formal concepts (nodes) in the lattice. For each formal concept, instead of listing each formal object (crawled document) individually, we instead display the object count (for example, the uppermost formal concept contains 119 documents, which is 100% of the total objects contained in the lattice). Formal attributes which were empty or contained no results (e.g. *Buy-agent*) were removed from the lattice to reduce visual clutter. It should be noted that the original document corpus is much larger in size and contains much more locations; due to the length limits imposed by the paper, we have restricted the documents to transactions which have occurred in two particular locations: Neverland and Alwaysland. (Please also note that for ethical purposes, all identified persons, locations and organizations have had their names changed to fictional ones. The anonymized dataset can be acquired by contacting the authors.)

Several information stand out in the lattice: 68% of all drug sales took place in Neverland, with the remaining 32% taking place in Alwaysland. John Doe and John Roe deal exclusively in Neverland. Both of them deal cocaine. John Doe is the only one selling marijuana, while John Roe is the only one selling crystal meth, which is, alongside cocaine, one of the most sought-after drugs in Neverland. Interestingly, there are no evident transactions of marijuana in Alwaysland.

Upon closer inspection of the lattice, it becomes evident that the most interesting seller is Jane Doe, which is the exclusive seller of cocaine and crystal meth in Alwaysland. However, while she sells exclusively in Alwaysland, she also plays the role of a buyer in Neverland. In particular, she has bought crystal meth 26 times from Neverland. What might that imply? For starters, we already know that John Roe is the only one selling crystal meth in Neverland (28 transactions). We can thus hypothesize that 92% (26 out of 28) of the time, John Roe sold crystal meth to Jane Doe. As such, we can imply that the supply of crystal meth in Alwaysland actually originates from Neverland, in particular by John Roe selling it to Jane Doe, who in turn sells it in Alwaysland. This is an interesting observation, which requires further investigation.

²<http://sourceforge.net/projects/fcabedrock/>

³<http://sourceforge.net/projects/conexp/>

